POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Procedural music generation for video games conditioned through video emotion recognition

Tesi di Laurea Magistrale in
Music and Acoustic Engineering

Author: **Francesco Zumerle**

Student ID: 962729
Advisor: Prof. Massimiliano Zanoni
Co-advisors: Luca Comanducci
Academic Year: 2022-23

Abstract

Video games have consistently become a predominant form of entertainment in recent years. As products encompassing diverse technological and artistic elements, including computer graphics, video and audio design, music composition, and more, they have attracted increasing research efforts across various scientific disciplines. Specifically, open-world video games, characterized by non-linear narratives and numerous gameplay scenarios, are currently one of the most popular genres. In such games, creating music that accommodates a vast array of events and variations poses a considerable challenge, as human composers find it exceptionally difficult to create music for every conceivable situation. Moreover, the recent success of a few indie games has proven the interest of both developers and gamers in artistic and emotional experiences, characterized by high musical and visual interactivity. Therefore, leveraging advancements in deep learning techniques, we introduce a new method to generate procedural music tailored for video games, with a particular focus on the open-world genre. Our approach involves extracting emotions, as modeled on the valence-arousal plane, from gameplay videos, assuming that these emotional values correspond to those experienced by the player. Subsequently, we employ this emotion-related data to condition a music transformer architecture, generating MIDI tracks that align with the emotional dynamics of the game. To demonstrate the effectiveness of our proposed technique, we conducted a perceptual experiment involving human players. This study not only evaluates the method's efficacy but also explores its applicability within the realm of video game music generation, providing useful insights for future researches in this field.

Keywords: Video game audio, procedural music generation, human-centered AI, affective computing, convolutional neural network, transformer

Abstract in lingua italiana

I videogiochi ad oggi costituiscono una delle forme di intrattenimento di maggior successo. Dietro alla loro realizzazione vi è spesso il lavoro minuzioso di un gran numero di artisti e programmatori, che si occupano di game design, narrazione, composizione musicale, computer grafica, e molto altro. Di conseguenza, la crescente popolarità del medium unita alla sua multidisciplinarietà sta suscitando sempre maggior interesse nella ricerca in vari ambiti scientifici. In particolare, tra i generi di maggior successo vi sono i giochi open-world, in cui ciascun giocatore è libero di esplorare vasti mondi, incontrando un gran numero di sfide e di eventi casuali. In tali giochi, la creazione di musiche che si adattino a questo enorme numero di possibili variazioni rappresenta una sfida considerevole, poiché un singolo compositore difficilmente è in grado di comporre una colonna sonora per ogni combinazione di situazioni. A tal proposito, negli ultimi anni alcuni sviluppatori indipendenti hanno iniziato a proporre brevi esperienze artistiche e coinvolgenti, caratterizzate da suoni e immagini che puntano a reagire costantemente alle diverse azioni del giocatore. Di conseguenza, sfruttando gli ultimi progressi nel deep learning, presentiamo un nuovo metodo per la generazione di musica procedurale per videogiochi, pensato in particolare per le esperienze open-world. Innanzitutto, il nostro approccio è composto da un primo modello che determina costantemente le emozioni suscitate dal video di gioco, modellandole secondo Valence e Arousal e assumendo che questi valori effettivamente rappresentino le emozioni del giocatore. Successivamente, i due valori ottenuti vengono utlizzati per condizionare un music transformer, un'architettura che genera tracce musicali MIDI, che comporrà quindi una colonna sonora coerente con l'impatto emotivo delle immagini di gioco. Per dimostrare l'efficacia della tecnica proposta, abbiamo condotto un test percettivo coinvolgendo dal vivo i diversi partecipanti. Questo lavoro non solo valuta l'efficacia del nostro metodo, ma esplora anche la sua effettiva applicabilità nell'ambito della generazione di musica per videogiochi, fornendo utili spunti per ricerche future in questo campo.

Parole chiave: audio nei videogiochi, musica procedurale, human-centered AI, affective computing, rete neurale convoluzionale, transformer

Contents

Abstract **i**

Abstract in lingua italiana **iii**

Contents **v**

1 Introduction **1**

2 Theoretical Background **5**

2.1 Video games 5

2.1.1 Definition and Audio component 5

2.1.2 Video game Genres and Categories 6

2.2 Emotion Modeling 9

2.2.1 The Circumplex Model of Affect 10

2.3 Convolutional Neural Networks 11

2.3.1 From Artificial Neural Networks to CNNs 11

2.3.2 The CNN Architecture 14

2.4 Transformer 18

2.4.1 Transformer: an Introduction 18

2.4.2 Core Architecture 20

2.4.3 Attention Mechanism 21

2.4.4 Positional Encoding 24

2.5 Music Transformer 25

2.5.1 Music Domain and Data Representation 25

2.5.2 Relative Attention 26

2.5.3 Musical Motif and Primer Conditioning 28

2.6 Conclusive Remarks 29

3 State of the Art **31**

3.1 Generative music in Video games . . . 31
3.1.1 Role and Motivations . . . 31
3.1.2 Implementation and Dimensions . . . 32
3.1.3 iMuse (1991) and Horizontal Arrangement: from Monkey Island 2 until Doom (2016) . . . 33
3.1.4 No Man's Sky (2016): the power of Vertical Arrangement . . . 35
3.1.5 MetaCompose (2017): focusing on the Generative Dimension . . . 36
3.2 Video game Emotion Detection . . . 37
3.2.1 In-game features . . . 38
3.2.2 Visual features . . . 39
3.2.3 Expanding Horizons: Emotion Detection applied to Movies . . . 42
3.2.4 Affective Datasets of audiovisual content . . . 43
3.3 Deep Learning and Conditional Music Generation . . . 45
3.3.1 Generative models and Controllability: an introduction . . . 45
3.3.2 Affective music generation models . . . 46
3.3.3 Affective Datasets of symbolic music . . . 48
3.4 Conclusive Remarks . . . 49

4 Proposed Approach 51
4.1 Video emotion Detection . . . 51
4.1.1 Design Choices . . . 52
4.1.2 Dataset Pre-processing . . . 53
4.1.3 Architecture . . . 54
4.2 Music generation conditioned on emotions . . . 59
4.2.1 Design Choices . . . 59
4.2.2 Architecture and Emotion Conditioning . . . 60
4.2.3 Primer conditioning . . . 62
4.3 Final model . . . 63
4.3.1 Overall Architecture . . . 63
4.3.2 Proposed Architecture as a Video game Music System . . . 65
4.4 Conclusive Remarks . . . 66

5 Experimental Setup and Evaluation 67
5.1 Emotion Estimation Model . . . 67
5.1.1 Dataset and Model Implementation . . . 67
5.1.2 Results . . . 69
5.1.3 Discussion . . . 73
5.2 Subjective Evaluation . . . 73

5.2.1 Experimental Setup 73
5.2.2 Results 80
5.2.3 Discussion 87
5.3 Conclusive Remarks 89

6 Conclusions and Future Developments **91**
6.1 Main Contributions and Results 91
6.2 Future Developments 92
6.3 Conclusive Remarks 93

Bibliography **95**

A Appendix A **105**

List of Figures **109**

List of Tables **111**

Ringraziamenti **113**

1 | Introduction

Music is a fundamental component of video games. Not only its role has rapidly gained value and interest as the medium's history unfolded, but it has always been strongly linked to technology and computer science's evolution. In 1978, for the first time ever, a game named *Space Invaders* [2] included a continuous background soundtrack, consisting of four descending bass notes repeating in a loop: incidentally, that simple theme was also an early example of adaptive music, since during gameplay the tempo gradually increased as the enemies moved closer to the player (video example). At that time, music development was slow because its programming on early machines was difficult and time-consuming [30]: still, developers managed to compensate those limits with their creativity, foreshadowing an approach that later would be named *generative music*. Nowadays, the industry situation seems the opposite: with the widespread diffusion of music production technologies, even small teams are able to create beautiful games with memorable soundtracks (e.g. *Undertale*, *Crypt of the NecroDancer* [7, 10]). Moreover, recent breakthroughs in AI generative systems, such as ChatGPT and Midjourney [41], have raised new ethical questions concerning the future of art. For instance, in 2022, during an annual art competition, a piece explicitly created by the software Midjourney claimed the first prize, igniting debates and dissatisfaction among participants and critics [81]. These cases have prompted artists and creators to discuss the extent to which artistic work can be delegated to a machine, without compromising its perceived value. From our perspective, the key to interpret technology's relationship with art and creativity lies in fostering innovation rather than merely imitating existing tasks.

In the context of video game music, we specifically examined the *open-world* genre, that has recently become extremely popular. In these games, the player has the freedom to explore worlds full of activities and events, resulting in nonlinear experiences that provide a multitude of gameplay scenarios. As a consequence, music must account for numerous variables in order to coherently respond to the player's actions. This has led to the increasing application of procedural and interactive music [74], which involves creating music through automated procedures, typically controlled by specific parameters extracted from gameplay. However, in most commercial games the use of these techniques is still limited,

since interactivity often extends only to major gameplay changes, while the generative approach is primarily employed for recombining pre-composed pieces. Interestingly, certain recent video games, such as *Journey* and *Abzû* [8, 11], are proposing artistically and emotionally engaging experiences in which music and visuals strive to reflect the player's actions throughout the game, although within a more linear context compared to open-world games. Nevertheless, such cases are rare, as it remains extremely challenging for a human to individually compose music for every possible in-game situation, even with current generative music techniques.

As a consequence, after reviewing recent advances in the field of video emotion detection [60, 61, 95] and affective music conditioning [92, 102], we designed a system that constantly analyzes a game's video stream, predicts the emotions perceived by the player and continuously generates music that aligns with those emotions (an article based on this thesis has been accepted to a scientific conference [104]). To the best of our knowledge, this is the first research proposing a complete framework that links emotion analysis solely based on visual information and conditioned music composition. We believe our study can provide valuable insights into the current strengths and limitations of all the techniques employed, questioning and evaluating their future applicability in a real-world context. Since this type of assessment is often lacking in deep learning research [91], it could provide essential guidance for future research directions.

Predicting emotions from videos is a challenging task, firstly because defining an objective mapping between a visual stimulus and a specific emotion contrasts with the subjective nature of affect. Secondly, even though deep learning models already showed promising results in predicting emotions from movies, thanks to the availability of adequate datasets [64], the same results have not yet been reached for video games. This can be explained by the lack of equivalent datasets for this medium, but also by the fact that until now other approaches where preferred (e.g manually defining associations between game events and emotional response [18, 76]). Conscious of these limitations, we trained a 3D Convolutional neural network with the LIRIS-ACCEDE dataset, composed of short movie excerpts associated with Valence and Arousal values, which together represent a wide range of human emotions [21]. Then, we employed a pre-trained Music Transformer able to generate symbolic music conditioned on the same two continuous values [92], customizing its inference algorithm so that it generates a musical continuation starting from any input MIDI file. Next, combining these two blocks we built a final architecture that performs procedural music generation conditioned through video emotion recognition, our initial goal. Finally, we implemented and evaluated our model with a real video game, analyzing its performance by conducting a subjective experimental evaluation, mainly

consisting of an emotion annotation task and a questionnaire. Specifically, we recorded gameplay videos of *No Man's Sky* [12] and automatically generated music for each one of them, simulating real-time inference. We additionally generated baseline videos containing unconditioned music from the same model, original soundtrack and no-music, for a total of 4 video categories.

We collected Valence and Arousal continuous annotations for all types of videos and from each participant, globally 272 time series, observing that our proposed procedure significantly outperforms the original soundtrack in terms of emotion conditioning, suggesting the effectiveness of our approach in eliciting the desired sentiment. On the contrary, annotations of videos without music showed that user's subjectivity still constitutes a challenge in this kind of tasks. From the questionnaire responses we draw even more promising conclusions, observing that the majority of participants recognized our proposed approach as the most coherent with both gameplay events and emotions, with respect to its counterpart generated without conditioning. Multiple challenges remain, such as the lack of affective music datasets covering different genres, which according to our evaluation could be a huge limit to the player's immersion if gameplay and music's style are not coherent. Lastly, a real-time implementation of our system remains a future work, even though we believe that a proper low-level optimization and the power of cloud gaming [87] already provide satisfying conditions for easily achieving this goal.

The contents of this thesis are organized as follows.
In *Chapter 2* we provide an essential theoretical background of the main concepts and techniques at the basis of this work. After presenting some basic notions related to video games and emotion modeling, two deep learning architectures are analyzed: the convolutional neural network and the music transformer.
In *Chapter 3* we illustrate current state-of-the-art works regarding different aspects of our research. We begin by presenting a broad perspective on generative music techniques employed in game industry, highlighting their strengths and current challenges. Subsequently, we explore the topic of emotion recognition, applied to both video game and movie domains. Lastly, we present latest advances in the field of affective music generation, focusing on deep learning techniques based on valence-arousal features.
Chapter 4 contains the complete exposition of our proposed method, conceptually divided in two distinct blocks: video emotion detection and conditioned music generation.
In *Chapter 5* we initially evaluate our proposed CNN for the emotion detection task, briefly discussing its performance. Then, we extensively present our experimental setup for the subjective evaluation, analyzing and discussing all the obtained results.
Lastly, *Chapter 6* is devoted to conclusions, proposing suggestions for future developments.

2 | Theoretical Background

In this chapter we present the theoretical background at the basis of this thesis. First, section 2.1 introduces the reader to the video game medium, defining terms and categories that will be used in future discussions. Followingly, the circumplex model of affect, the emotion modeling adopted for this work, is described (section 2.2). The remaining sections provide the mathematical and theoretical groundings of the neural networks employed in next chapters. Specifically, section 2.3 is devoted to Convolutional neural networks, the architecture we employed for predicting emotions from video frames, and contains also an introduction with the bases of deep learning. Then, sections 2.4 and 2.5 respectively discuss the Transformer and Music Transformer models, the latter being an extension of the former and the actual architecture employed for generating music in this work.

2.1. Video games

This section begins by defining what is a video game and providing a description of its different audio components (subsection 2.1.1). Then, a selected subset of terms commonly used in this field is presented and explained, in order to facilitate the fruition of the next chapters (subsection 2.1.2).

2.1.1. Definition and Audio component

Nowadays, the video game industry is among the highest grossing markets on the planet, with the most relevant games selling millions of copies worldwide. To give an idea, *Doom* [62], released in 2016, in one year sold 2 million copies only considering the PC version, while the most commercially successful titles go even beyond, reaching tens of millions of copies sold [101]. Nonetheless, in this section we provide a few basic concepts in order to help a reader who may not be familiar with this medium.

Starting from a definition, a video game is a system based on the interaction between one or more users and a digital device (e.g. a computer, a console, etc...) by means of an input device (e.g. mouse and keyboard, controller, ...). The player usually receives visual

feedback through a TV or Monitor and audio feedback with speakers or headphones. While the therm "video game" may suggest a predominance of the visual component in the interaction, the audio plays a dominant role in many aspects.

The audio component of a game is usually divided in 3 categories [74]:

- **speech**, that encompasses all the dialogues between characters inside a game. This component is usually recorded by voice actors, though some games have also used other techniques to communicate with the player. For instance, in the Platform genre it's common to play sound effects while visualizing a text to reproduce dialogue lines (*Celeste* [14] is an excellent example of this case, as showed in this gameplay excerpt);
- **sound effects**, which include all audio elements that can be defined aperiodic and nonmusical. They may be realistic or abstract, recorded or synthesized. A sound effect can have different purposes, like giving feedback to a player about the result of a task (e.g. your attack was successful or failed). They can also serve as a warning for some events that may happen, like an enemy that could see the protagonist or a treasure chest that may be close;
- **music**, which is the category that we will focus on inside this thesis. Generally it has a certain continuity over time, it is pitched and it has a regular time division, even though these features are just indicative and they are not to be taken as absolute.

2.1.2. Video game Genres and Categories

When a video game is presented to an audience, its name is usually accompanied by a few key-words providing the reader an overall idea of its main characteristics. Currently a wide amount of terms is used by specialized websites, reviewers and among gamers, many of which are neologisms specifically created for this medium and they cannot be compared to any definition used in other fields. In fact, developers are continuously experimenting new ideas, for example by combining peculiarities of multiple "traditional" genres into a single game, and sometimes the results gain so much success and relevance that a specific nomenclature is required.

As discussed in [28], this situation poses many issues and prevents researchers to easily define a fixed list of video game categories. Nonetheless, in this subsection we provide a representative subset of genres and classification terms, so that the reader can more easily understand the future discussions involving video game examples.

First of all, video games can be classified focusing on their *Gameplay*, which can be

defined as "the overall nature of the experience defined by a pattern of interactions and game rules" [56]. Consequently, we report 10 resulting genres proposed by Lee et al. [56] based on gameplay mechanics:

- **Action**: Games with a heavy emphasis on a series of actions performed by the player in order to meet a certain set of objectives. In most action games, players perform each intended action by pressing the right buttons with a precise timing. Conversely, in some other genres players simply choose the desired action, with the computer assuming responsibility for executing the action's performance [20];
- **Action/Adventure**: Games which are set in a world for the player to explore and complete a certain set of objectives through a series of actions;
- **Driving/Racing**: Games involving driving various types of vehicles as the main action, sometimes with an objective of winning a race against an opponent;
- **Fighting**: Games involving the player to control a game character to engage in a combat against an opponent;
- **Puzzle**: Games with an objective of figuring out the solution by solving enigmas, navigating, and manipulating and reconfiguring objects;
- **Role Playing Game - RPG**: Games with an emphasis on the player's character development and narrative components;
- **Shooter**: Games involving shooting at, and often destroying, a series of opponents or objects;
- **Simulation**: Games intending to recreate an experience of a real world activity in the game world;
- **Sports**: Games featuring a simulation of particular sports in the game world;
- **Strategy**: Games characterized by players' strategic decisions and interventions to bring the desired outcome.

This list is far from being complete and flawless, as pointed out by the authors. Although mutual exclusivity was one of the key objectives in developing this classification, certain categories exhibit vague demarcations and conceptually overlap. Among the others, *Action* is clearly a general term (in some ways comparable to the musical *Pop* genre) and could be seen as a wider set including many of the other terms like *Fighting* or *Shooter*.

Based on the specific games discussed in this thesis, below we present an additional list of terms that better specify some gameplay components of a game:

- **Platform** (or Platformer): it can be seen as a sub-genre of either *Action* or *Puzzle* video games, depending on the actual gameplay components [22]. The primary goal is to navigate the game character between different locations within a given environment, consisting of uneven terrain and suspended platforms of varying height that require jumps, climbs and even more actions in order to be traversed.
- **Survival**: games belonging to this sub-genre usually require the player to manage resources, craft tools, build shelters or collect items in order to survive, while completing the main objectives.
- **Open-World**: this term describes a design philosophy of the game world and a set of aligned game mechanics. "An open-world is a nonlinear virtual world in which the player has the agency to roam freely and tackle objectives in the order they choose. Obstacles can typically be overcome in multiple ways in order to facilitate player freedom and to achieve a heightened sense of player agency." [98]

Another common classification criteria is the game's *Presentation* - "the manner or style of game display" [56]. Generally, a first subdivision can be done between **2D** and **3D**, indicating the number of dimensions used to represent the game's space. Additionally, more specific terms are commonly used to further define the visual perspective employed in a game : [1]

- **2D**, the space is represented in two dimensions:
  - **top-down**: A 2D game that provides an overhead or bird's-eye view of the action;
  - **side-scrolling**: A 2D game where the character moves along one dimension (left-right or up-down) and the screen scrolls following them. The perspective is usually set from the side;
  - **isometric**: the point of view is from above the game world (bird's-eye view). Isometric games employ solely 2D *sprites* (2D graphical elements or images that are part of the game's visual composition), while obtaining a 3D appearance by appropriately setting the camera angle. This visual technique reveals aspects of the environment that would remain hidden if observed solely from a purely two-dimensional perspective.
- **3D**, the space is represented in three dimensions:

[1] The proposed classification combines information provided by [56] and the Unity documentation page Game perspectives for 2D games

- **1st person**: the player sees the game world from the point of view of the game character he is controlling;
- **3rd person**: the point of view is behind the game character controlled;
- **2.5D**: this technique is analogous to the isometric perspective, but 3D geometry is used for the environment and characters, even though the gameplay is still restricted to two dimensions.

A visual comparison between some of the listed perspectives is provided in Figure 2.1.

Figure 2.1: A comparison between 4 common video game perspectives: (a) first person - *Halo Reach*, (b) third person - *Red Dead Redemption 2*, (c) side-scrolling - *Hollow Knight*, (d) isometric - *Hades* [4, 13, 15, 16].

2.2. Emotion Modeling

Before going into the details of emotion modeling, it's important to spend a few lines understanding what is an emotion. According to Keltner and Gross, emotions are "episodic, relatively short-term, biologically based patterns of perception, experience, physiology, action, and communication that occur in response to specific physical and social challenges and opportunities [53]. This intricate definition presents emotions as responses to some kind of event or thoughts. Another article states that "Emotions are the tools by

which we appraise experience and prepare to act on situations" [29], emphasizing that they are also functional in helping us react to a stimuli.

Several researchers and psychologists have studied and proposed computational models aimed at representing human emotions [67]. In this thesis, we will investigate techniques based on the circumplex model of affect.

2.2.1. The Circumplex Model of Affect

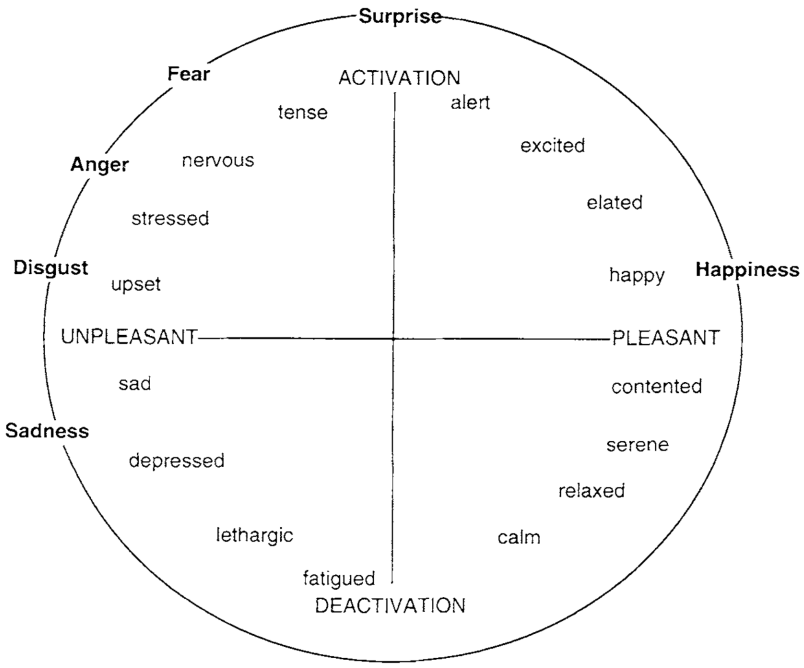


Figure 2.2: The affective circumplex, depicting each emotion along continuous dimensions of arousal (y-axis) and valence (x-axis). The inner circle shows a schematic map of core affect, while the outer circle indicates where several prototypical emotional episodes typically fall [84].

Following a dissertation by Posner et al. [77], for a considerable time clinicians and researchers have observed individuals struggling to precisely identify their own emotions according to discrete categories. This difficulty implies that emotions are often felt as ambiguous and overlapping experiences, rather than distinct entities. Just like the spectrum of colors, emotions seem to lack well-defined boundaries that would clearly set them apart from one another. More concretely, rarely do individuals describe experiencing a single positive emotion without simultaneously acknowledging other positive feelings. These connections, often obscured in experimental designs focusing on discrete emotions, are directly tackled by dimensional models of emotion, which perceive affective experiences

as a continuous range of closely interconnected and frequently unclear states.

After an extensive analysis of emotional connections using different statistical techniques, researchers have consistently produced two-dimensional (2-D) models of emotional experiences [77]. The most commonly used dimensions are Valence and Arousal, as proposed by Russel in 1980 [83]. Valence describes how positive or negative an event is, whereas arousal models the activation of an event, ranging from calming/soothing to exciting/agitating. Inside this thesis we will sometimes reference them with the abbreviations *valence-arousal* and *V-A*.

Every emotional experience is described by a linear combination of these two independent values, which is then understood as indicative of a specific emotion. For instance, according to circumplex theorists *fear* is understood as a neurophysiological state arising with the combination of negative valence and high arousal (this and more examples are depicted in Figure 2.2).

2.3. Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a type of neural network architecture that in recent years has allowed researchers to obtain remarkable results in many computer vision and machine learning problems. After an introduction providing some basic concepts of machine learning and deep learning (subsection 2.3.1), the core structure of a CNN is defined, along with its fundamental elements (subsection 2.3.2).

2.3.1. From Artificial Neural Networks to CNNs

An Artificial Neural Network (ANN) is a computational system inspired by the functioning of biological nervous systems, such as that of the human brain. The building block of ANNs is the artificial neuron, a mathematical model which performs three distinct operations: weighting, summation and activation.

First, the inputs received by the neuron are weighted, i.e every single value is multiplied with an individual weight. Compared to the human brain, negative weights indicate inhibitory connections, while positive values reflect excitatory connections. Second, a summation function adds up all the weighted inputs and the bias term. Third, the result of the previous operations is passed trough an activation function.

Formally, let $\mathbf{x} = [x_0, x_1, ..., x_n]$ be an input vector of length n that is received by a neuron j. Let also $\mathbf{w}_j$ be the neuron's weight vector, b_j its bias parameter and $\mathcal{F}_j$ the activation

function. Therefore, the output y of the overall mathematical model can be described as

$$y = \mathcal{F}_j \left(\sum_{i=0}^{n} x_i w_{ij} + b_j \right), \tag{2.1}$$

where $x_i \in \mathbf{x}$ indicates an input received by the neuron and w_{ij} is its correspondent weight [55]. In general, $\mathcal{F}$ could be any mathematical function and it's used to introduce non-linearities in the system, hence allowing the resolution of more complex problems. A few examples of popular activation functions are: Rectified Linear Unit (ReLU), Sigmoid, Exponential Linear Unit (ELU), Logistic.

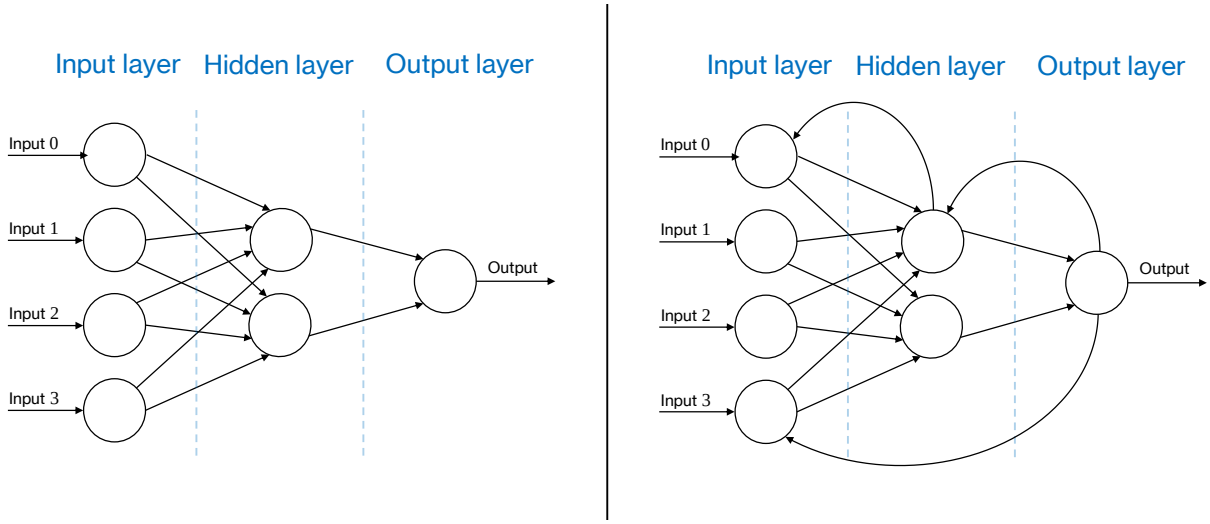


Figure 2.3: Example of the two main Artificial neural network topologies: Feed-Forward Neural Networks (left) and Recurrent Neural Networks (right). Feed-Forward Neural Networks (FFNN) require the information to flow only in one direction (acyclic graph), while Recurrent Neural Networks contain one or more cycles in their structure (cyclic graph). The depicted structures, comprising an input layer, one hidden layer and an output layer, can be expanded with any number of nodes and connections and constitute the basis of most common ANN architectures.

While an artificial neuron alone has almost no real usefulness, the combination of two or more units determines an artificial neural network, which depending on its topology (how single neurons are connected and arranged inside a graph, see Figure 2.3) finds application in a wide variety of fields. However, once the final ANN architecture has been designed it is not yet ready to be used: like happens in its biological counterpart, the network must be trained in order to learn the desired behaviour. Commonly, there are three learning paradigms:

1. **Supervised learning**: For each training example provided, the network receives a set of input values and the desired output. The goal of this procedure is to minimize the overall prediction error of the model by iteratively determining the output value of each example and comparing it to its ground truth value. After each iteration, the network's weights are adjusted in order to improve the performance;
2. **Unsupervised learning**: Differently from before, no ground truth labels are provided. In this paradigm the network is expected to uncover statistically significant characteristics within the input dataset. In other words, since no desired output is specified, the system needs to construct its own interpretation of the input stimuli;
3. **Reinforcement learning**: this type of learning significantly differentiates from the previous two since it does not rely on a specific set of training data. Instead, the ANN iteratively performs some actions inside a dynamic environment, constantly receiving a feedback. After each iteration, its parameters are adjusted according to a reward or cost function. Overall, the system's goal is to maximize the total amount of reward received (or minimize the total amount of cost).

For our purposes we will consider the first approach in next analyses.

The learning process of the network occurs in repeated cycles, known as *epochs*. At the end of each epoch, the obtained result is compared with the ground truth value through a *loss function*, and depending on the obtained result an *optimizer* adjusts the weights of the network accordingly, so that the loss is minimized. The choice of the loss function depends on the type of problem addressed, which usually belongs to one of the following categories:

- **Classification**: the goal in these problems is to correctly assign a class, chosen from a discrete pre-determined set, to each input received. A simple example of real-life problem is the "email spam detection", where the network must learn to correctly recognize spam emails, i.e. assign the correct class $c \in \{\text{spam}, \text{not-spam}\}$ to any mail analyzed. Loss functions used in this case include: *Cross Entropy*, *Binary Cross-Entropy*;
- **Regression**: differently from before, in these problems we want to predict continuous values. For instance, predicting the price of a car given a set of features (engine type, mileage, condition, ...) belongs to this category. *Mean Squared Error*, *Mean Average Error* and *Root Mean Squared Error* are a few relevant examples of loss functions in this case.

Once implemented, the loss function enables us to measure the effectiveness of the current

set of weights of our network. Then, the objective of the optimization process is to update those weights leading to the least possible value of the loss function. Currently, the most popular optimizers include: *Gradient Descent*, *Stochastic Gradient Descent* (SGD), *Adaptive Moment Estimation* (Adam). A common hyper-parameter named *learning rate* is used to determine the size of the steps taken to reach a local minimum in this minimization process. Its choice is crucial, since a too small learning rate could lead to slow convergence, while excessively large values may impede convergence, leading the loss function to oscillate around the minimum or even to diverge [82].

A significant drawback of conventional ANNs is their difficulty in handling the computational demands required for processing image data [70]. To clarify, let's assume to be training a network with a dataset of RGB images (3 channels) with resolution 64×64: a single neuron would have to manage 13 056 weights. Moreover, image-based problems require larger network topologies compared to the simple examples provided in Figure 2.3. This constitutes two huge problems. First, the computational power to train these networks is not unlimited and the current approach is clearly not optimal, since images are still being handled as vectors (see equation 2.1). Second, dealing with a larger number of parameters increases the risks of $Overfitting$, a condition arising when the network is not able to generalize the information provided by the training set and instead focuses on less relevant information, resulting in poor performance when tested on new data.

To overcome these issues and effectively handle images we must introduce a new deep learning architecture: Convolutional Neural Networks.

2.3.2. The CNN Architecture

Convolutional neural networks are a specific instance of artificial neural networks and the two share many common features, such as their structural organization comprising a sequence of layers. Despite that, a significant difference of CNNs is that their neurons are specifically designed for image processing, which allows for the embedding of specific characteristics within the architecture. For example, a fundamental property is *translation invariance*, meaning that a CNN layer is able to capture relevant information regardless of its position inside the image [94].

In order to understand the reason behind the "Convolutional" term, a brief mathematical background is provided. In general, a convolution $(x * k)(a)$ of two functions x and k over a common variable a is defined as

$$(x * k)(a) = \int_{-\infty}^{\infty} x(\tau)\, k(a - \tau)\, d\tau, \tag{2.2}$$

where $a \in \mathbb{R}^n$, $\forall n > 0$. In CNNs, x corresponds to the input, k is named *filter* or *kernel*, and the output is often referred to as activation, or *feature map*. However, while equation 2.2 is defined on continuous values, images are modeled as discrete objects, hence we replace the continuous τ term with a discrete variable i, used for a summation, resulting in the *discrete-time convolution*:

$$(x * k)(a) = \sum_{i=-\infty}^{\infty} x(i)\, k(a - i). \tag{2.3}$$

Finally, since images are 2-D objects, equation 2.3 can be easily extended for convolving across 2 dimensions, obtaining

$$(x * k)(a, b) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} x(i, j)\, k(a - i, b - j), \tag{2.4}$$

and the same can be done for videos by adding an ulterior dimension.

In practice, x is implemented as a 2-D matrix representing a single frame, while the filter k is also a matrix and it's computed for each acceptable value of a and b. This implies that we can implement and visualize equation 2.4 as the sum of a finite number of multiplications, i.e. the dot product between two matrices, performed for each i and j.

Convolutional neural networks are characterized by three fundamental building blocks: convolution layer, pooling layer, and fully connected layer. Respectively, these layers are responsible for feature extraction, dimensionality reduction and classification. Below we provide an in depth analysis of each component:

1. **Convolutional layer**. It can be visualized as the dot product between an input matrix and a moving filter that shifts across each input value. Figure 2.4 shows an instance of the described process.

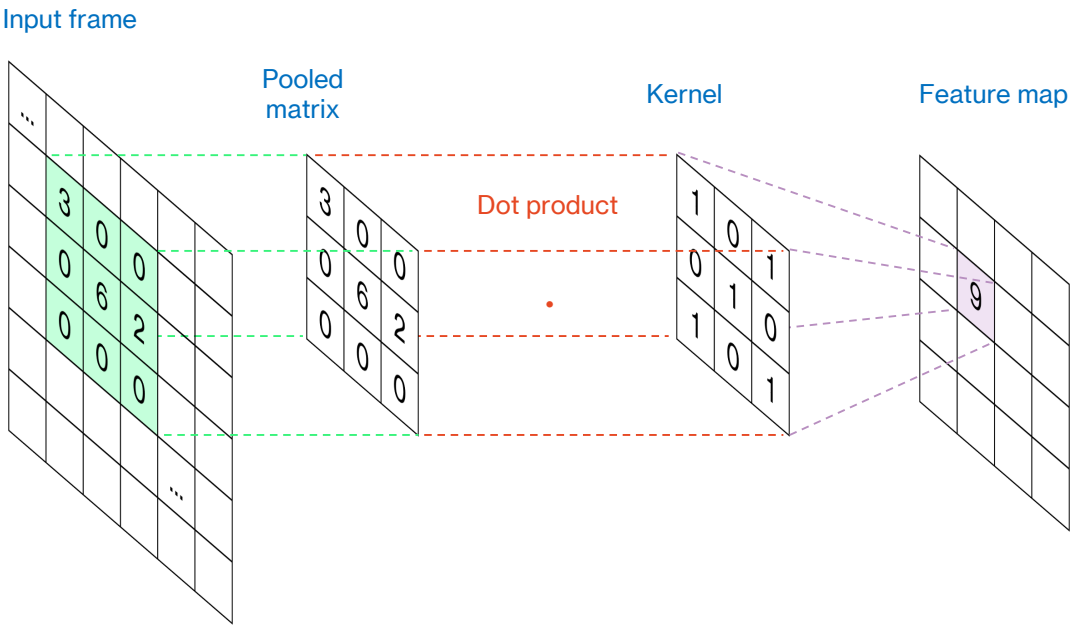


Figure 2.4: Scheme of the convolution operation performed by the convolutional layer, including a numerical example.

This layer can be adjusted with multiple hyperparameters.

- The **depth** refers to the number of learnable filters employed in the convolution: each filter performs a convolution operation on the input image, and the result is a single output feature map, hence the depth parameter directly determines the number feature maps. Intuitively, more filters allow detecting more features within the input data, but increase computational cost.
- **Kernel size** trivially refers to the dimensions of the filter employed.
- **Stride** defines the step size at which the filter moves across the input data during the convolution. For instance, a stride = 2 means that the filter will move of two matrix cells after each multiplication.
- Finally, **Padding** consists of adding extra pixels to the input data's edges before applying the convolution operation. This process allows to control the output feature map's size and can prevent information loss at the borders of the input frames.

As with traditional ANNs, an activation function is applied at the end of the computations. A common choice after a convolutional layer is ReLU function, defined

as

$$\mathrm{ReLU}(x) = \max(0, x), \quad x \in \mathbb{R}, \tag{2.5}$$

since it generally leads to faster training convergence, compared to other nonlinearities.

2. **Pooling layer**. The objective of a pooling layer is to generate a condensed statistical representation from its input, hence reducing the dimensions of the feature map without sacrificing essential information.

   Compared to convolution, here we still have a pooling window sliding along the input feature map, but instead of the dot product with a kernel other operations are performed, depending on the specified behaviour. Figure 2.5 illustrates Max-pooling, that computes the max operation with a 2×2 window. Other common pooling operations are *average pooling* and *stochastic pooling*.

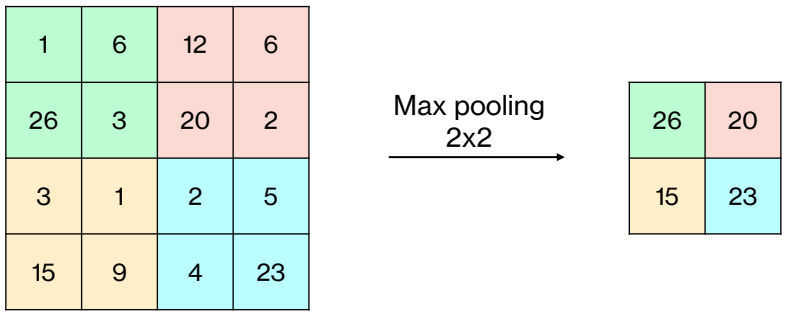


Figure 2.5: Example of max-pooling operation. In this case the feature map's dimensionality is effectively reduced by a factor of 2

   Also in this case, the hyperparameters **pool size** (or kernel size), **stride** and **padding**, already defined before, can be used.

3. **Fully-connected layer** (or Dense layer): it is composed of neurons that establish direct connections with the two adjacent layers, analogously to how neurons are linked in classical ANNs. A common practice is to apply a fully-connected layer at the end of the CNN architecture, reducing the overall number of neurons and obtaining a compressed representation of the features learned by the network.

After multiple iterations of the layers described above, usually a last layer is used to perform the desired behaviour with the learned knowledge. For classification problems, the *softmax* activation function is the most common choice, since it provides a vector of values that can be interpreted as probabilities. Each value indicates the model's estimated

probability that the input belongs to the corresponding class. The class with the highest probability is considered the final prediction for the current input (see Figure 2.6).

On the other hand, when dealing with regression tasks the simplest approach consists of reducing the output dimensionality to the number of values we want to predict, which can be easily done by means of a *dense layer* with a linear activation function.

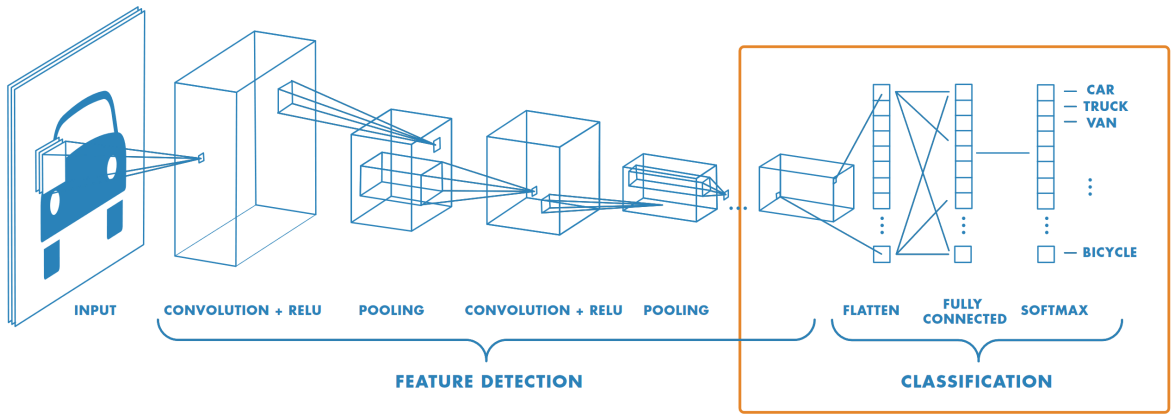


Figure 2.6: Example of CNN trained for object classification, from Matworks deep learning ebook.

2.4. Transformer

In order to understand the Music Transformer, the model employed for this thesis, a solid background on its original architecture is needed. Consequently, this section begins with an overview of the Transformer model (subsection 2.4.1). Then, its architecture (subsection 2.4.2), the attention mechanism (subsection 2.4.3) and the positional encoding (subsection 2.4.4) are explained.

2.4.1. Transformer: an Introduction

The Transformer is a deep learning model first introduced in 2017 by the famous paper "Attention is all you need" [97] and it follows the structure of an Encoder-Decoder, represented in Figure 2.7.

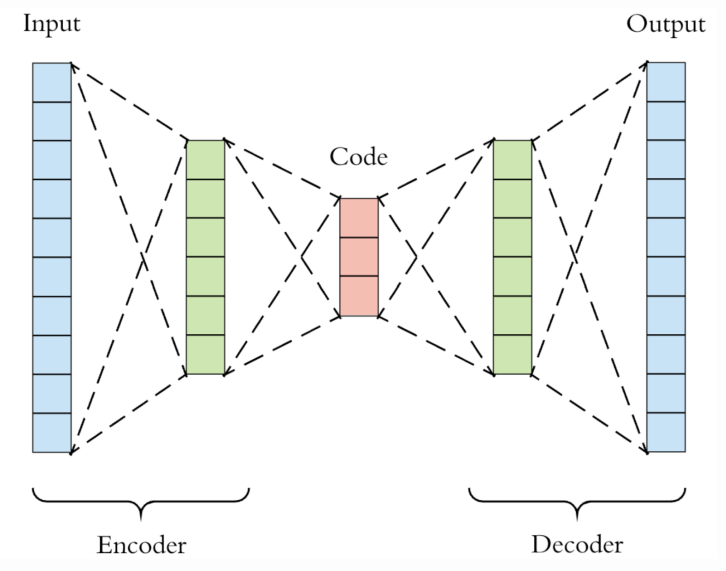


Figure 2.7: Encoder-Decoder basic structure. An *Encoder* is responsible for taking an input sequence and converting it into a fixed-length vector, while a *Decoder* receives that vector and converts it into an output sequence.

Dealing with natural language processing tasks, researchers had already proposed a large amount encoder-decoder models, mostly employing LSTM [47] and other Recurrent neural networks, including the attention mechanism. Then, the central idea presented in the Transformers paper, as suggested by its title, was that attention could serve as the sole mechanism for establishing connections between input and output dependencies.

In order to understand the Transformer architecture, let's first define some basic concepts:

- **Token**: in natural language processing, tokens are often words or subwords in which an input sentence is split. For instance, considering the sentence "Hello World", using a word-based tokenization the tokens might be: ["Hello", "world"];
- **Embedding**: it refers to a numerical representation of discrete elements, such as words, subwords, or characters, that are used as input to the model. Embeddings produced from a layer aim at capturing various semantic and contextual aspects of the original elements. Assuming a simple word embedding space where each word is represented by a 1-D vector of fixed length 3, the previously obtained tokens could become: "Hello" $= [0.2, 0.4, 0.1]$, "world" $= [0.5, 0.7, 0.6]$.

A Transformer's input consists of a sequence of tokens, that is received by the encoder and converted to a fixed-dimensional representation for each of the tokens, along with an additional embedding for the entire sequence. Then, the decoder receives the encoder's

output as input, producing a sequence of tokens as its output.
Formally, the encoder learns to map an input sequence of symbols $(x_0, x_1, ..., x_n)$ to a sequence of continuous representations $\mathbf{z} = (z_0, z_1, ..., z_n)$, while the decoder produces an output sequence $(y_0, y_1, ..., y_m)$, one element at a time. Additionally, the generative process of the decoder is autoregressive, because at each step the previously generated symbols are fed as additional input.

2.4.2. Core Architecture

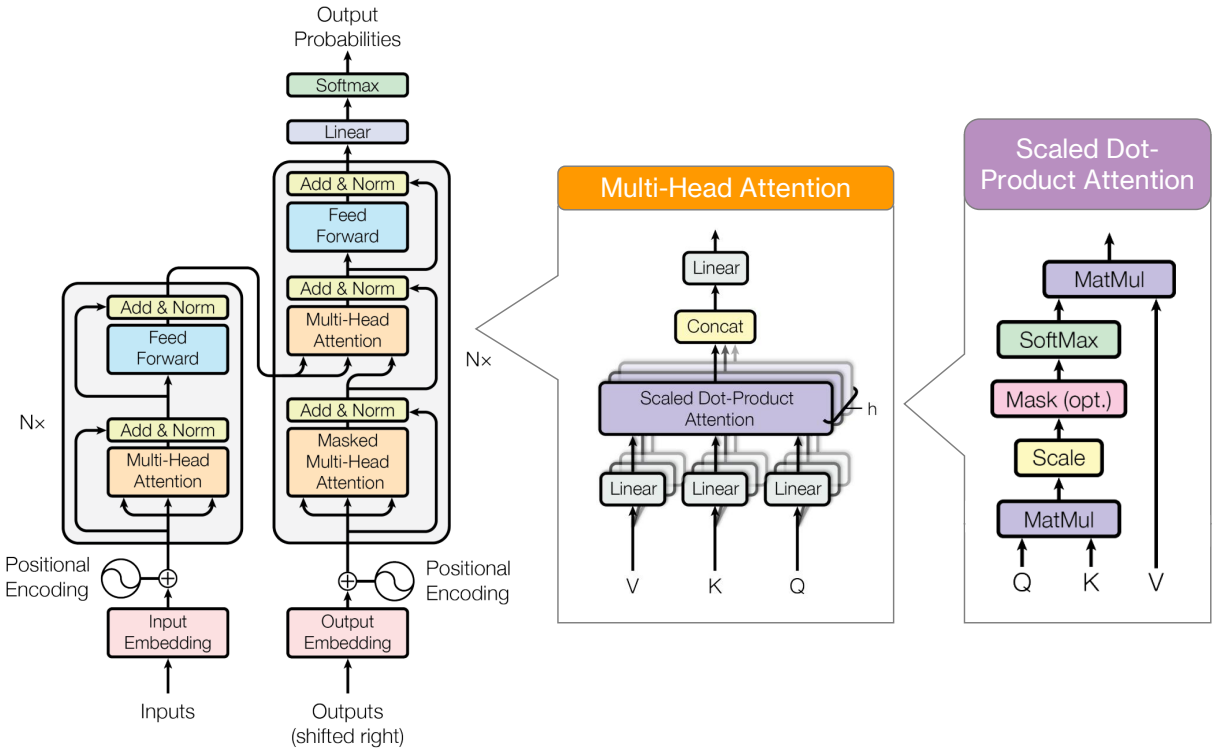


Figure 2.8: Scheme of Transformer's encoder-decoder architecture (left), with additional representation of the Multi-Head Attention block (center) and the Scaled Dot-Product Attention (right), the base function employed. Pictures from the original paper [97].

The Transformer's **encoder** is structured in 6 layer, each one composed of:

- a multi-head self-attention mechanism, combined with a residual connection and followed by a normalization layer.
- a position-wise fully connected feed-forward network, also combined with a residual connection and followed by layer normalization.

The output dimensionality of all layers, including embedding layers, is fixed at $d_{model} =$

512, so that residual connections are facilitated during training.

Subsequently, the **decoder** is also composed of 6 layers and each one of them includes:

- a masked multi-head attention layer, receiving the output of the encoder stack. The "masking" process consists of filtering out all tokens located to the right of the token for which the representation is being calculated. This ensures that the decoder's attention is limited to tokens preceding the one it's predicting. Also this block is combined with a residual connection and followed by a normalization layer.
- a multi-head self-attention layer + residual connection + normalization, identical to the encoder
- a position-wise fully connected feed-forward network + residual connection + normalization, identical to the encoder.

The complete architecture is depicted in Figure 2.8 (left image). Let's now discuss the attention mechanism, which is the building block for most layers listed above.

2.4.3. Attention Mechanism

Attention can be defined as the process of mapping a *query* and a set of *key-value* pairs in order to generate an output, that is obtained by computing a weighted sum of the values. Each value's weight is determined through a *compatibility function*, calculated on the query and the corresponding key [97].

Before describing the specific implementation of this approach inside the Transformer, we describe the concept without formulas, clarifying the advantages that its introduction provides.

Overview

The underlying idea of attention is to mimic how human attention works. Biologically, when we look at something we only focus on significant elements and forget/ignore irrelevant information [80]. Equivalently, in computer vision a model employing an attention mechanism will only focus on specific regions of images instead of the entire picture, therefore reducing computational cost and improving effectiveness.

The same concept has been successfully applied to natural language processing, the field of application where it became most popular. For instance, let's assume to be training a model on a dataset of online reviews: our goal is to predict an integer number $s \in [1, 5]$, the correspondent number of stars rating, from any input text.

Employing an attention mechanism in this case, as described by Lin et al. [58], means that the model, depending on the target output (i.e. number of stars), will learn to assign higher weight only to the most relevant words inside the input text. Figure 2.9 illustrates how the model learned to concentrate on specific words/sentences for 5-star ratings, mostly focusing on excerpts expressing positive emotions.

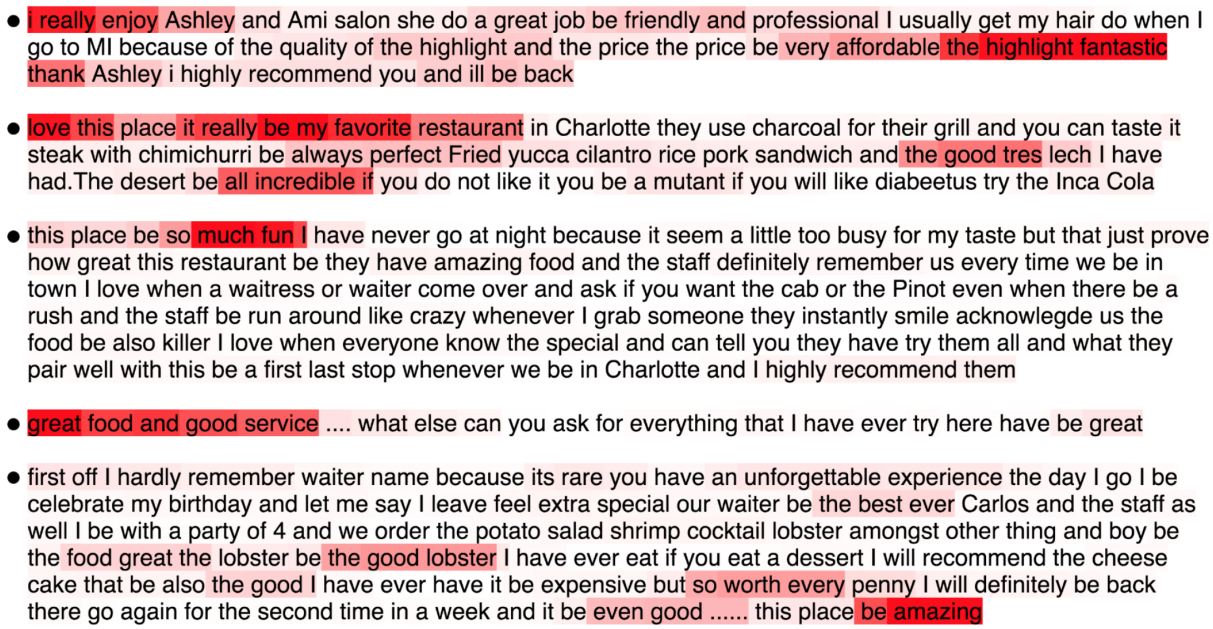
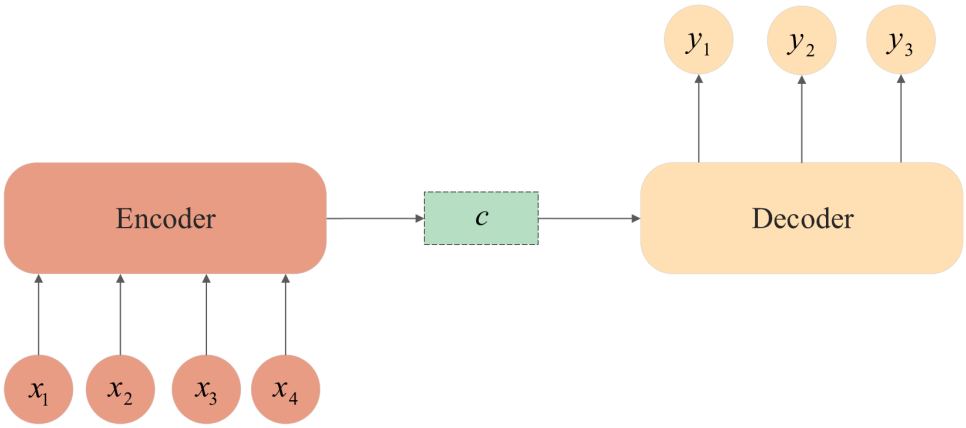


Figure 2.9: Heatmap of some reviews with 5-star score, from [58]. Red intensity indicates what the model mostly focuses on.

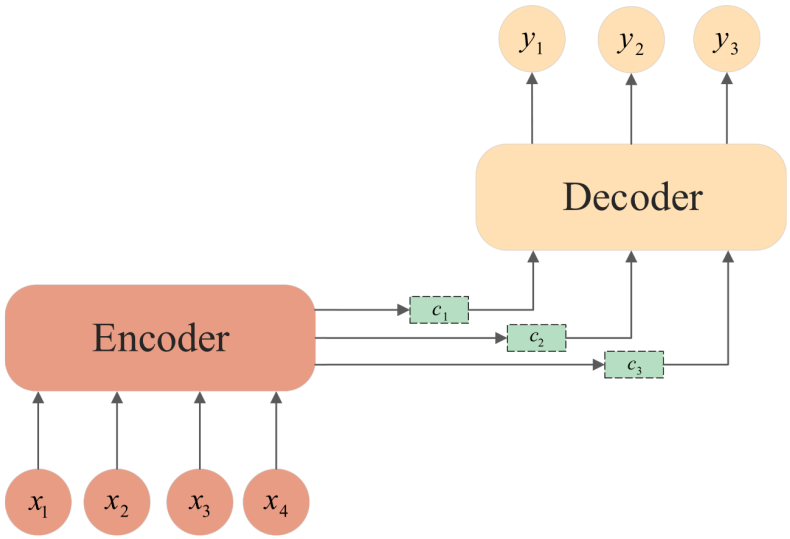
Even though the actual meaning of query, key and value may vary depending on the problem addressed, generally they can be seen as:

- **Query**: represents the context the model is trying to generate in the output (e.g a specific review rating);
- **Key**: Each key vector represents the context of an input token (e.g. position of a word inside a text, surrounding words, etc);
- **Value**: it holds the actual information associated with each input token (e.g. the actual word inside an input text).

To sum up, introducing the attention mechanism guarantees that the contributions of elements in the input sequence will vary depending on the target element being decoded. Consequently, a model employing attention will learn different representations of the input tokens depending on the query considered (Figure 2.10).



(a) Encoder-Decoder without Attention.



(b) Encoder-Decoder with Attention.

Figure 2.10: Comparison of two encoder-decoder architectures, focusing on the attention mechanism, from [68]. While in Figure 2.10a each input token has the same impact regardless of the output, in Figure 2.10b the attention approach ensures that each contribution is different depending on the decoded element.

Implementation

The Transformer employs a specific type of the attention mechanism named **Scaled Dot-Product Attention** (Figure 2.8, right image).

Let's define the dimensionality of input queries and keys d_k, and the length of values d_v, considering all the mentioned elements as vectors. The Scaled Dot-Product Attention is defined as the dot product of a query with all the keys, divided by a factor $\frac{1}{\sqrt{d_k}}$, which is then applied to a softmax function that returns a weight for each value.

In practice, the attention function is computed each time on a set of queries simultaneously, hence we define Q as a matrix containing the current query vectors, while K and V are the matrices with respectively all keys and values combined. Consequently, the resulting formula can be written as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{2.6}$$

where $K^\top$ indicates that the matrix with all the keys is transposed, so that the compatibility between a query and each key is computed.

Furthermore, as depicted in Figure 2.8 (center), the Transformer performs **Multi-Head Attention**, meaning that equation 2.6 is performed in parallel h times (specifically, $h = 8$), corresponding to the number of *heads*.

In practice, instead of performing a single calculation with the complete Q, K, V matrices, each head learns a linear projection for each of them, leading to different representation subspaces and allowing the model to jointly attend to information from each of them at different positions.

Formally, we define a single head_i, with $i < h$, as

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \tag{2.7}$$

where the learned projections are the parameter matrices $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$.

Subsequently, once each head is computed we perform a final concatenation $Concat$ and apply a linear projection into the original subspace with W^O, obtaining the final formula

$$\text{MultiHead}(Q, K, V) = Concat(\text{head}_0, ..., \text{head}_h)W^O, \tag{2.8}$$

with $W^O \in \mathbb{R}^{hd_v \times d_{model}}$.

2.4.4. Positional Encoding

Natural language processing tasks are sensitive to the ordering of input words. However, until now no recurrent mechanism was introduced, so we still need a tool to capture the relative or absolute position of each token inside the sequence. This can be done by using *Positional Encodings*, that are summed to the embeddings of both encoder and decoder, before any other calculation (Figure 2.8 on the left).

Positional encodings can either be learned or computed according to a fixed formula, and should have the same dimensionality d_{model} as the embeddings so that the sum can be computed.

After evaluating multiple options, the authors of the Transformer model opted for sine and cosine functions, with their frequency depending on current element's position pos and computing a different sinusoid for each index i of the embedding of dimension d_{model}, resulting in the formulas:

$$\mathrm{PE}_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}), \tag{2.9a}$$

$$\mathrm{PE}_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}). \tag{2.9b}$$

By adding positional encodings, the Transformer can finally differentiate between tokens based on their positions, understanding and capturing their sequential order within the sequence.

2.5. Music Transformer

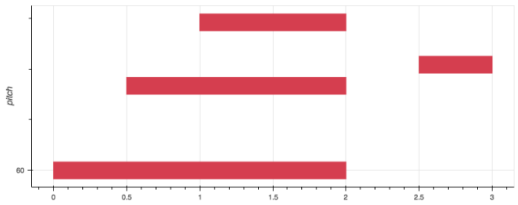
The Music Transformer is an architecture proposed in 2018 by Huang et al. [48], built upon the original Transformer method and adapted for generating symbolic music.

In this section we illustrate its main differences with respect to the base model presented until now. We initially describe how symbolic music is modeled, addressing its different peculiarities compared to natural language (subsection 2.5.1). Then, we define the specific attention mechanism employed inside this architecture, motivating its choice over the previous method (subsection 2.5.2). Finally, we define the concept of musical motif and illustrate the procedure named primer conditioning, part of the proposed approach of this thesis (subsection 2.5.3).

2.5.1. Music Domain and Data Representation

Compared to a normal text, music compositions often contain recurring elements at multiple levels. To clarify, a musical phrase, composed of a few notes performed with a specific timing, is usually repeated inside a section. At the same time, sections composing a piece are usually repeated more than once, although with some variations introduced: for instance, most pop-music pieces follow a common structure like *verse-chorus-verse-chorus-bridge-chorus*, while classical compositions also rely on simple schemes like ABA.

Before deciding how to capture these stratified repetitions, it's crucial to first choose a mathematical representation that allows to model a music piece without losing fundamental information. The authors of the Music Transformer chose to follow the modeling proposed by Oore et al. [69], which starting from a MIDI piece converts all the midi note events into a sequence of tokens, capturing both the time and pitch component of each note (Figure 2.11 shows an example of this encoding). Depending on the dataset employed to train the model, the vocabulary of tokens is defined accordingly.



```
SET_VELOCITY<80>, NOTE_ON<60>
TIME_SHIFT<500>, NOTE_ON<64>
TIME_SHIFT<500>, NOTE_ON<67>
TIME_SHIFT<1000>, NOTE_OFF<60>, NOTE_OFF<64>,
NOTE_OFF<67>
TIME_SHIFT<500>, SET_VELOCITY<100>, NOTE_ON<65>
TIME_SHIFT<500>, NOTE_OFF<65>
```

Figure 2.11: Picture from [48] where a segment of piano performance is represented as a pianoroll (left) aside with its correspondent performance events (right) ordered from left to right and then descending the rows. The pianoroll depicts an arpeggiated and sunstained C Major chord, followed by an F note. The encoding employed captures all the performance details, including the time progression (`TIME_SHIFT`), the different sunstains (`NOTE_ON`, `NOTE_OFF`) and different velocities (`SET_VELOCITY`).

2.5.2. Relative Attention

To effectively model the repetitive nature of music, the authors decided to replace the Scaled Dot-Product Attention of the original Transformer (equation 2.6) with a more suited function.

The problems of applying attention to a musical piece can be summarized in two instances:

1. With music, the **relative distance** between elements is more relevant than their absolute position inside a sequence;
2. Both **time** and **pitch** must be taken into account when measuring the relative distance between notes, as the former determines features like rhythm and the latter captures musical intervals between notes.

Consequently, the authors implemented a relation-aware version of self-attention, proposed by Shaw et al. [86] and specifically deigned for music, that captures the relative distance between two positions. Moreover, they extended the above mentioned so that both relative timing and pitch are considered in the learned representations.

Concretely, starting from equation 2.6 let's additionally define a relative position embedding E^r of shape (H, L, d_k), with H = number of heads, L = sequence length and d_k = query and key length. For each possible value of r, this component models how far two elements are apart in a sequence, therefore representing the pairwise distance $r = j_k - i_q$ between a query q and key k, where i and j indicate the position of each element respectively. The resulting embeddings are sorted in ascending order, from $r = -L+1$ to $r = 0$, and each head learns them separately.

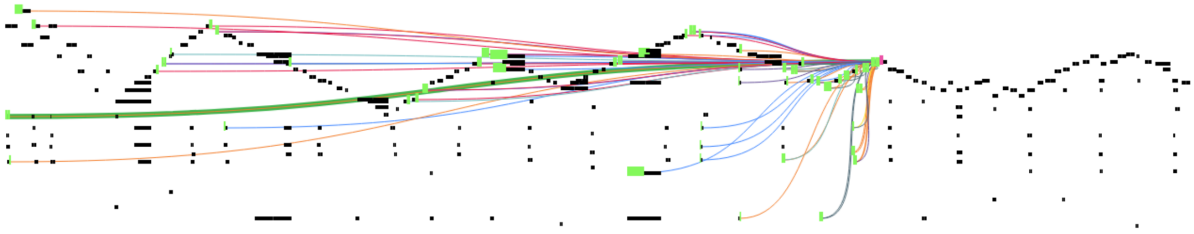
Then, these relative embeddings are multiplied by the query matrix Q (already defined in equation 2.6) resulting in $S^{rel} = QE^{r\top}$, where each entry (i_q, r) should contain the dot product between the query in position i_q and the embedding of relative distance r. Specifically, researchers needed to perform a additional "skewing" process in order to effectively obtain the result described: since its detailed discussion goes beyond our scopes, we refer the interested reader to the original paper [48].

Finally, the resulting S^{rel} consisting of an $L \times L$ dimensional logits matrix can be combined with the original attention formula (equation 2.6) obtaining: [2]

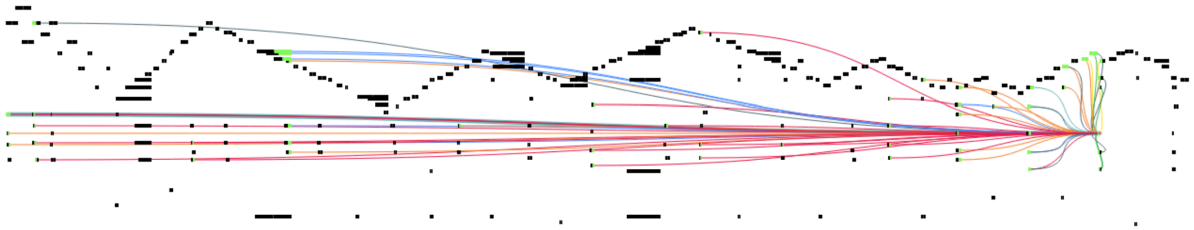
$$\text{RelativeAttention} = \text{Softmax}\left(\frac{QK^\top + S^{rel}}{\sqrt{d_k}}\right)V, \tag{2.10}$$

where the relative distance information is effectively employed for the attention mechanism. Figure 2.12 illustrates how in practice the described formula allows the Music Transformer to capture musical structure and recurrences.

[2] Although Huang et al. [48] used a slightly different notation, for simplicity we maintain equivalent symbols already defined in previous Attention formulation (equation 2.6).



(a) This first piano piece shows a recurring triangular contour. The current query is at one of the last peaks and the arcs with highest weight are linked to previous peaks until the beginning of the song.



(b) Here, the current query is a note performed with the left hand (i.e. the accompaniment part, shown in the lower half of the pianoroll). While attending to all the closest previous notes, it's also clear how the attention is mainly directed to all the previous left hand chords, proving the Music Transformer's ability to discern between accompaniment and melody.

Figure 2.12: To visualize the attention mechanism, Figures 2.12a and 2.12b show the last layer of attention weights at two different points of the generation process. The arcs starting from the current note (i.e. query) show which notes in the past are informing the future generation, with their color indicating the different heads and their width the resulting softmax probability. Notes highlighted in green are receiving the highest softmax probabilities. Pictures from [48].

2.5.3. Musical Motif and Primer Conditioning

In generative music models, **primer continuation** refers to the practice of generating or extending a musical piece starting from an input musical segment. In order to understand the usefulness of this procedure, let's first define the concept of **motif**, that Anders et al. summarized as:

Proposition 2.1. *A short musical idea, melodic, harmonic, rhythmic, or any combination of these three. A motif may be of any size, and is most commonly regarded as the shortest subdivision of a theme or phrase that still maintains its identity as an idea [19].*

In classical music, a common approach to composition consists of starting with a small unit of one or two bars, i.e. a motif, and developing it to obtain a melody or music phrase [45]. Therefore, imitating this procedure with deep learning can lead to realistic

and elaborate results, enabling the use of generative models as a creative tool in multiple applications.

In particular, the Music Transformer has proven to be capable of capturing and elaborating an initial brief motif, creating musical phrases with distinct contours that are subsequently reiterated and varied over time. To visualize this described procedure, we present one of the experiments performed by the authors.

Specifically, Figure 2.13 depicts the initial motif from Chopin's Étude Op. 10, No. 5 (on the left) and a novel composition obtained with the Music Transformer conditioned on that musical excerpt (on the right). Analyzing multiple inferences, the authors observed how the Music Transformer is able to compose coherent and pleasant music even in long pieces, while still considering and expanding the initial primer during generation. [3]

Figure 2.13: Primer continuation performed with the Music Transformer.

2.6. Conclusive Remarks

In this chapter we've established the theoretical foundation for our thesis. First, we introduced key concepts regarding video games, clarifying the role of audio in this medium and defining the most common classification terms, related to genre and visual perspective. Then, we mentioned the circumplex model of affect, which constitutes the basis upon which we will analyze and discuss emotions in the subsequent chapters. After that we explained the bases of deep learning, defining procedures, terms and concepts that are common among all the neural networks we employed. Finally, we defined all the architectures that will be part of our proposed approach and experiments, i.e. the Convolutional neural network and the Music Transformer. Specifically, we focused on providing insight on fundamental mathematical concepts, such as convolution for CNNs and attention for Transformer models.

[3] For more examples and audio samples, see `https://magenta.tensorflow.org/music-transformer`

3 | State of the Art

For this chapter, we have organized the contents in a logical order, following a top-down approach. Initially, section 3.1 will be devoted to present the topic of generative music inside video games, first focusing on why procedural and interactive music can be useful if applied to a video game, and then describing some real-world applications in order to give an idea of the impact of generative music systems. Then, the second and third sections will cover the two main components of our proposed work: the video emotion analysis (section 3.2) and the affective music generation (section 3.3).

3.1. Generative music in Video games

This section is organized in 5 subsections. First, subsection 3.1.1 consists of a brief introduction, showing the main findings regarding music's contribution to the gaming experience. Next, generative music for video games is examined in subsection 3.1.2 according to two different dimensions: adaptivity and generativity. Finally, in the following subsections (3.1.3, 3.1.4, 3.1.5) some relevant generative music systems are presented, highlighting their advantages and limitations.

3.1.1. Role and Motivations

First of all, music is nowadays a core part of video games, and it's almost impossible to find a modern video game sold without a decent music component. Even if we look at indie games (short for independent video games, they are typically created by individuals or smaller development teams without the financial and technical support of a large game publisher [100]) the composer is often a figure involved. Moreover, the video game industry has become incredibly profitable in recent years, ranking among the highest grossing forms of media in the market [26], so developers have started to invest more resources in order to improve each of its component, including music generation and interaction. But what are the advantages of a good musical component inside a video game? Only recently some researchers started to investigate this topic, and lately a few publications have shown interesting results.

Klimmt et al [54] in 2019 performed an experiment using the famous Triple-A video game *Assassin's Creed: Black Flag* [9], and measured the contribution of the original soundtrack to different components of the overall experience. The study, which involved 68 young male individuals with previous playing experience, showed that players experienced greater enjoyment with the presence of the soundtrack, compared to playing the game without music.
In the same year, Plut et al. [73], focusing on adaptive music (i.e. music that is modified in real time during gameplay, reacting to in-game events), demonstrated that when musical tension adapts to game tension, the player experiences it with higher intensity. Additionally, they showed that users are able to perceive whether the soundtrack is coherent with the current gameplay, and concluded that the adaptivity feature of music, applied to tension, contributes to the gaming experience.

3.1.2. Implementation and Dimensions

The most basic way to use music inside a video game is by playing a music piece linearly and continuously, starting again the playback when the end is reached: basically, the audio file is constantly played in a loop. The choice of the audio track may depend only on the current level, like happened in older games (e.g. *Super mario bros* [65]), when each time a new level is loaded the correspondent audio track is reproduced in background. In this first situation, music does not react to any gameplay change; nonetheless, this approach cannot be applied to many genres, such as open-world games, where there is no real distinction between levels. In this second case the approach must be different, and the strategy adopted usually consists of the music reacting to a change of game state, for example when the current state moves from "combat" to "non-combat", or vice versa: both these states have an associated music piece, and in the event of a transition the current music track can either be ended abruptly or there can be a fading between the current and next piece. It must be noted that in all these cases the music is played just as it was provided by the composer, without any kind of novelty. According to [74], all techniques aimed at extending a linear composed piece can be categorized in two ways:

1. **adaptive music**, or interactive music, concerns all the techniques that make the music react to game states. To be precise,

   Proposition 3.1. *we can talk about adaptive music when the generation process maps game variables to musical features [74].*

   For instance, A game variable can be the player's current health or the number of enemies in the surrounding area, and a variation of one of these parameters can

determine the removal of an instrumental layer from the current music, or a change of tempo. The adaptivity of the game's soundtrack can be seen as a dimension: music with low adaptivity is limited to adjusting to a small number of in-game factors, whereas music with high adaptivity can dynamically respond to numerous in-game variables, ranging from tens to even hundreds;

2. The second approach, **generative music** or procedural music, is related to the actual creation of musical content:

   Proposition 3.2. *music can be considered generative within a video game if it is produced by a systemic automation that is partially or completely independent of the gameplay [74].*

   So, according to this definition even if we use a music generation system that is completely independent from the video game, the result can be still defined generative. Implementing this approach inside a music system can be useful since it can substantially increase the amount of music available. Moreover, some games like Journey [8] aim at proposing a unique musical experience at each gaming session, and as a consequence generative methods can be a powerful approach for granting this result.

It's important for developers to have clearly in mind all the main features of their game, in order to choose the right music system with the focus on the most important characteristics. In order to better explain this topic, which will be crucial for understanding our proposed method, we will briefly present one of the most influential music systems in this field, iMuse [90], describing how its architecture has been successfully implemented for the first time in 1991 and giving an few examples of how its heritage has still a great impact on modern games.

3.1.3. iMuse (1991) and Horizontal Arrangement: from Monkey Island 2 until Doom (2016)

Despite being created long ago in 1991, this generative system has had the largest influence on current state of the art methods, since most of them are still largely based on this framework [72]. Its workflow starts by reproducing a piece of music stored symbolically. Then, if there are no in-game changes, the piece will just be reproduced linearly. Otherwise, depending on the new game state, the system will gradually move from the current piece to the next one, creating the effect of a seamless transition. It must be noted that iMuse does not compose new music, instead it behaves as an arrangement system, mean-

ing that it recombines existing musical elements in new ways in order to obtain the effect of a smooth transition. This was made possible by introducing many constraints on the composed music, since all the pieces should be written with compatible keys and tempo. Moreover, each song had to be be annotated with all the possible time instants where a transition could start. A short demonstration of iMuse can be seen in the following video example, where it was implemented for the first time inside the famous video game *Monkey Island 2: LeChuck's Revenge*, a released in 1991 [39]. With each change of scene, the music immediately starts to adapt, adding or removing instruments or modifying the main melody. This approach strongly focuses on the adaptivity dimension, and it can be categorized as *horizontal arrangement* [74], since the music evolves horizontally by concatenating musical themes or phrases over time. The final result was so powerful and effective that the same, improved strategy for generating reactive soundtracks is still used nowadays.

Figure 3.1: A screenshot of Monkey island 2 [39] and Doom [62], discussed in this subsection

In 2016, 25 years later, the software house *id Software* published a new video game named *Doom* [62], presented as a reboot of their homonymous game originally released in 1993 for MS-DOS, keeping the core gameplay idea and improving each component, including music. In this case, the soundtrack written by the composer Mick Gordon was first subdivided into musical phrases. Then, these phrases were grouped in different categories, based on the different sections of a standard song structure: intro, verse, chorus, bridge, outro. Each category had more or less 30 possible phrases, and during gameplay they were selected based on current game state, in order to react coherently to each situation. This kind of approach has a peculiar characteristic, being that during each gaming session the soundtrack will appear as a long unique music piece, with no noticeable interruptions and a high reactivity to the player's actions. Furthermore, since the game belongs to the First

Person Shooter (FPS) genre these characteristics were fundamental in order to produce and immersive and adrenaline-filled experience, which is one of the foundations of the series success. On the other hand, since Doom allows any musical phrase to transition to any other phrase (which every time is selected randomly among phrases of the same category), in order to make this process work the composer had many constraints, like writing all the music for each level with same key and bpm. This limit is common among games based on this kind of systems, and may be overcome with the introduction of a generative approach.

3.1.4. No Man's Sky (2016): the power of Vertical Arrangement

In the same year as Doom, another game named *No Man's Sky* [12] was released, gaining a lot of popularity due to its promises of infinite exploration inside a procedurally generated universe. In fact, this Adventure-Survival game offered each player an almost infinite number of planets, each one with its unique flora, fauna and environment [93]. While keeping most of the properties seen until now, the music system of No Man's Sky makes one more step forward *generativity* by also performing what is defined as *vertical arrangement* [74]. In practice, unlike in previous games where each musical piece or phrase was played as a unique music track, with this approach the current piece can be altered in many ways during its playback. For instance, the arrangement can happen at instrument level, meaning that while a specific level is being played the soundtrack evolves by adding or removing new instruments, without moving to a different music piece. *Halo: Combat Evolved* [3] and its sequels used this approach for their own music system, adding or removing groups of instruments in real-time, making the player hear a single piece of music with many different arrangements, reducing the fatigue from repetition [74].
The corpus of music for No Man's Sky was initially composed by a band named *65daysofstatic* and then categorized according to five possible game states: "Wanted", "Space", "Planet", "Map", and "AmbientMode". Then, each original piece was divided into small clips in order to be used by the generative system. With this process and some more restrictions, like having a fixed key and tempo within each composition, all resulting clips can be assumed to fit well with the others obtained from the same song. Once the music system starts to work, each time the game moves to a new state a new soundtrack is generated by pseudo-randomly combining music clips together. The choice of instruments used for the resulting music is updated depending both on the player's location and what he is looking at. More details regarding the rules and restrictions that govern the combinations of musical elements were not made accessible.

3.1.5. MetaCompose (2017): focusing on the Generative Dimension

Metacompose [85] is a recent music system proposed in 2017. The authors describe it as a "compositional, extensible framework for affective music composition". To elaborate further, this system has the ability to generate music in real-time for any interactive application, while also conditioning it on a desired Valence value. In order to describe its workflow, we will define it as a combination of two systems

- The first one, which accomplishes the **composition** task, works with a combination of a rule based algorithm and a genetic algorithm. The workflow starts with the first algorithm choosing a chord progression by random walking through a directed graph, like the one depicted in Figure 3.2, which represents the possible chord transitions. After this process is complete and the chord progression is defined, a genetic algorithm generates the melodic line, using a specific fitness function based on counterpoint composition. Once the melody is complete, the chord progression must be converted into an actual accompaniment, and that is possible with two additional algorithms: the first one generates an euclidean rhythm, which has the characteristic to maintain even and repeating divisions of time, while the second one chooses an arpeggio line among a set of pre-composed patterns for chord playback over time.
- The second one focuses on **emotional playback**, since it starts from the generated music and alters it according to provided values of Valence and Arousal. In particular, arousal is directly linked to volume, while valence determines the brightness of the timbre. In addition, the lower the valence, the more the notes will be altered, creating dissonance in order to obtain tension. Finally, a drum track is chosen by the system according to both features, which will determine its velocity and regularity.

Differently from before, this system was proposed by researchers inside a scientific paper and, to the best of our knowledge, it has not been applied to a commercial game.

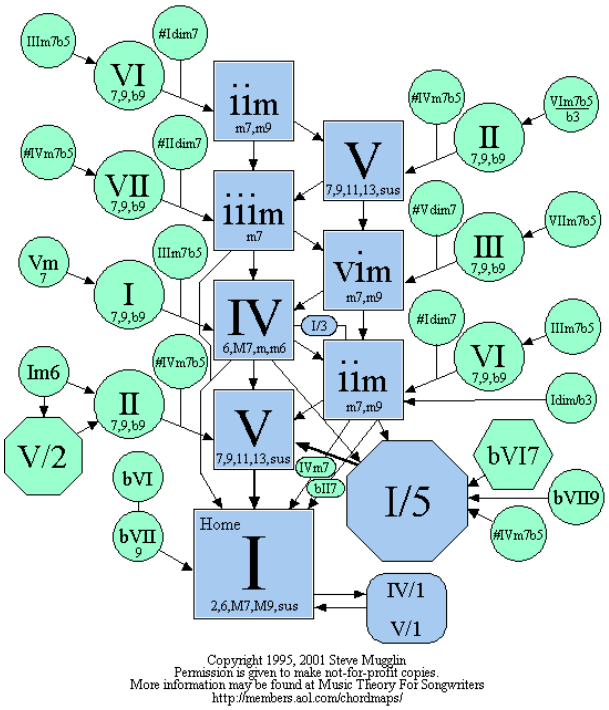


Figure 3.2: Common chord progression map for major scales, from [1]

3.2. Video game Emotion Detection

When people play video games they experience emotions, and understanding how to predict or elicit these emotions is currently an open field of study. In fact, there are many challenges around this topic: which kind of features should be taken into account in order achieve the most accurate predictions? Which technique is best suited for this task? In subsection 3.2.1 we will cover articles that propose affective music systems that determine player's emotional state by analyzing in-game parameters (values returned by the game engine, such as current health points or number of enemies nearby). Then, subsection 3.2.2 will present approaches which determine emotions by analyzing the game video stream, while in subsection 3.2.3 we will discuss models that apply the same approach to movies.

We will not cover methods involving the use of physiological features (a rich overview of the current state of the art techniques has been written by Granato et al. [42]) since this approach would require specific instrumentation, posing many limits both for game developers and players.

3.2.1. In-game features

An interesting method was proposed by Hutchings et al. in 2020 [50], which discussed that currently the development of adaptive music systems (AMS) for video games faces many difficulties, one of them being the lack of solid strategies for effectively modeling player actions, game-world context and emotions. Their proposed AMS was conditioned by a spreading activation model of game context: it was implemented as a weighted undirected graph $G = (V, E)$. Each vertex V in the graph belongs to one of three categories(affect, objects and environments) and its weight represents the activation value ranging from 0 to 100. The association strength between vertices is captured by edge E weights: a higher weight facilitates the propagation of activation. Whenever a new game begins, a new graph is initiated, and every 30ms it receives messages sent from the game engine, which are used to update it by adding new vertices, new edges, or modifying weights. As depicted in Figure 3.3, the result of this process is a contant real-time representation of the following six affect categories: sadness, happiness, threat, anger, tenderness, and excitement.

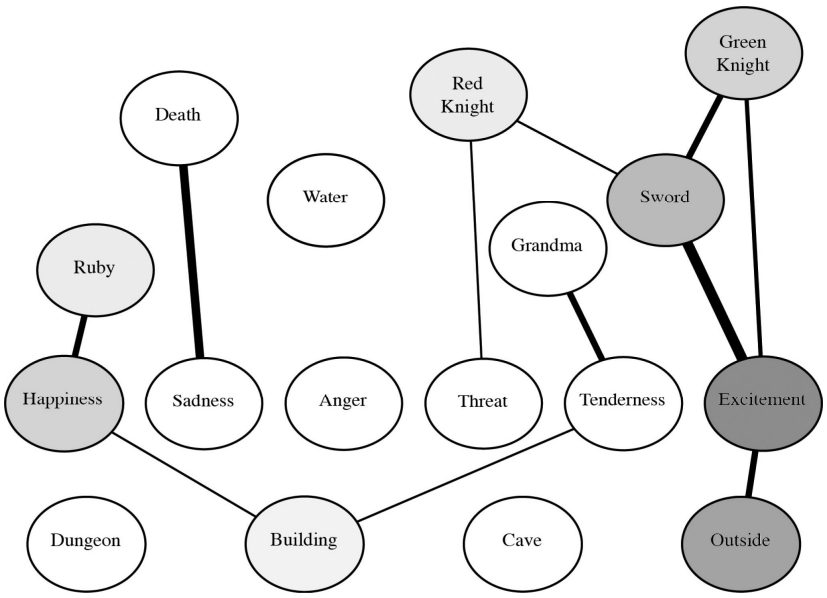


Figure 3.3: Visualization of the spreading activation model proposed in [50]. Node activation level is represented by gray-scale shading, while edge weights are indicated by edge width.

Their proposed system has been integrated in two games belonging to different genres (*Zelda: Mystery of Solarus* [6] and *Starcraft II: Wings of liberty* [5]) and in their subjective

evaluation gamers have reported for both games a higher overall sense of immersion and a stronger correlation between the music and the game-world events, when compared to the original game soundtracks.

In 2022 Plut et al. [75, 76] developed an application for affective adaptive music generation, implemented in the context of a single-player action-RPG video game. For the affective analysis, the researchers proposed a "Predictive Gameplay Layered Affect Model" (PreGLAM), which they defined as a "cognitive agent that models a spectator with a provided bias". In their demonstration, the authors biased the model in order to root for the player, so that their application will generate music that coherently comments successes and failures during each game. In particular, PreGLAM begins with a default mood value, which is then constantly updated depending on player's actions and other in-game events (e.g. an enemy has damaged the player, or the player has used a new ability, more details in Figure 3.4). Also Amaral et al. [18] in the same year proposed an interesting pipeline for the same goal, but when referring to the task of emotion detection and the different in-game features that could be taken into account, they stated "How to aggregate these various informations is still an open issue for future work".

Event	Valence	Arousal	Tension	Modifiers
P. complete atk combo	1	1	1	Missing O. shield
P. heavy atk	1	1	1	Missing O. health
O. atk combo	1	1	1	Missing P. shield
O. heavy atk	-2	1	2	Missing P. health, Parry active
P. shields down	-2	1	2	Missing P. health
O. shields down	2	1	2	Missing O. health
P. exploit O. disable	3	1	2	Missing O. health
P. death	-3	1	3	P. shield recharge time
O. death	3	1	3	O. shield recharge time
P. heal	2	1	2	Missing P. health
O. heal	-2	1	2	Missing O. health

Figure 3.4: Emotionally evocative events in-game, defined in [75]. P. indicates actions or events related to the player, while O. is related to the opponent controlled by the game. Each row indicates the resulting emotional changes determined by PreGLAM

3.2.2. Visual features

In 2010, Joosten et al. [52] conducted an experiment in order to assess whether the use of four different colors elicited specific emotions to gamers. The experimental setup consisted of asking the participants to play a specifically built video game, where background colors were manipulated during playtime. After that, each of the 60 participants was asked to

fill the Self-Assessment Manikin questionnaire [23] in order to measure his/her emotional response. As a result of the study, the authors found out that the color red was associated with high arousal and negative valence (Anger), while yellow elicited high values both of valence and arousal (Joy).

Following a similar approach, Geslin et al. [37] in 2016 suggested in their work that brightness, saturation and color choice are determinant factors when developers try conveying emotions to the players. As part of their experiments, 85 participants were asked to observe 24 randomly selected video game frames and, between each observation, they had to fill a semantic subjective questionnaire. After collecting all the results, the authors measured a strong correlation between the feeling of joy or sadness and some visual features, including: brigthness, value, chroma, lightness. Following these results, the authors proposed a "Circumplex model for emotions induction in video games and virtual environments" (Figure 3.5), presenting it as a tool to be used by developers in order to better understand and use color design for this task.

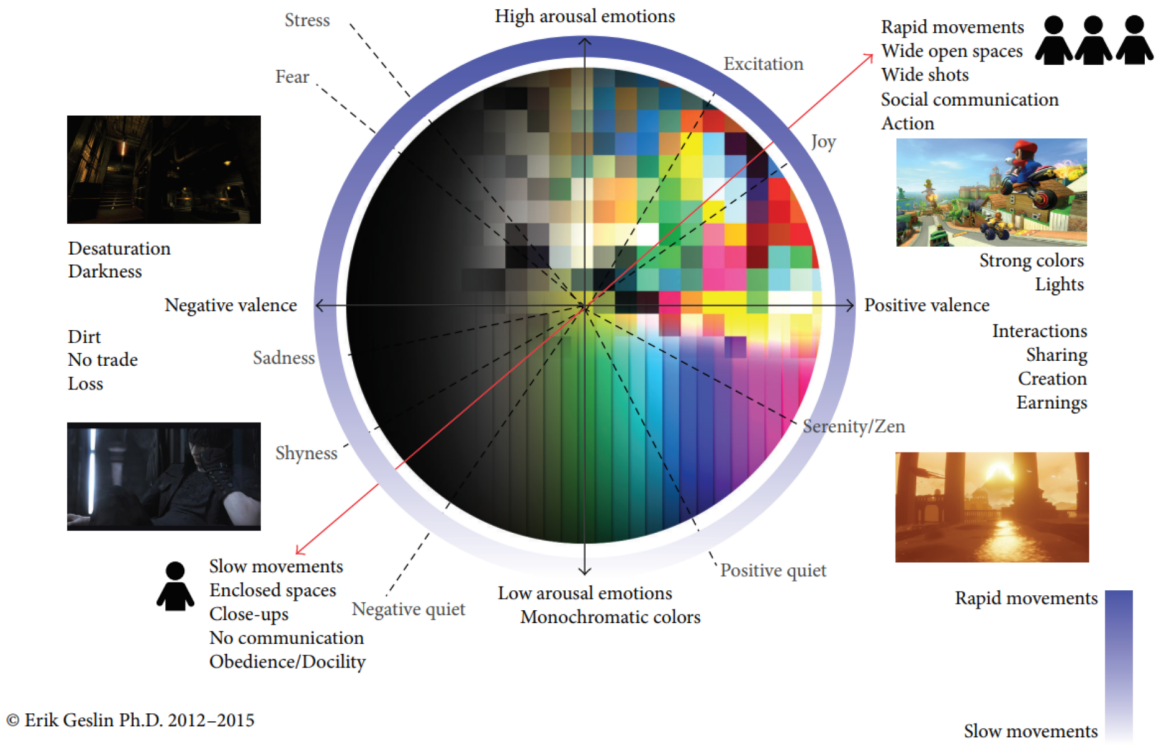


Figure 3.5: A Circumplex model for color scripting in video games, proposed by [37].

Changing perspective, Makantasis et al. [60] in 2019 tried to answer the following question: "Is it possible to predict the affect of a user just by observing her behavioral interaction through a video?". In other words, the goal of the authors was to find out if the arousal

perceived by a player could be predicted solely from the gameplay video, without gaining other information from the user or from the game: this is exactly the same objective of this thesis, even though we will try to predict both Valence and Arousal. The authors proposed three different Convolutional Neural Networks (see Figure 3.6) and trained them on a small dataset of gameplay videos coupled with user's arousal annotations, meaning that each video had an associated time-series representing the evolution of arousal over time. An annotation task involves different volunteers that, using a specific interface like Ranktrace [59] and PAGAN [63], are asked to provide their real-time perception of a specific emotion, e.g. Arousal. All networks showed a similar performance, and were able to correctly classify high vs low arousal level with 78% accuracy on average.

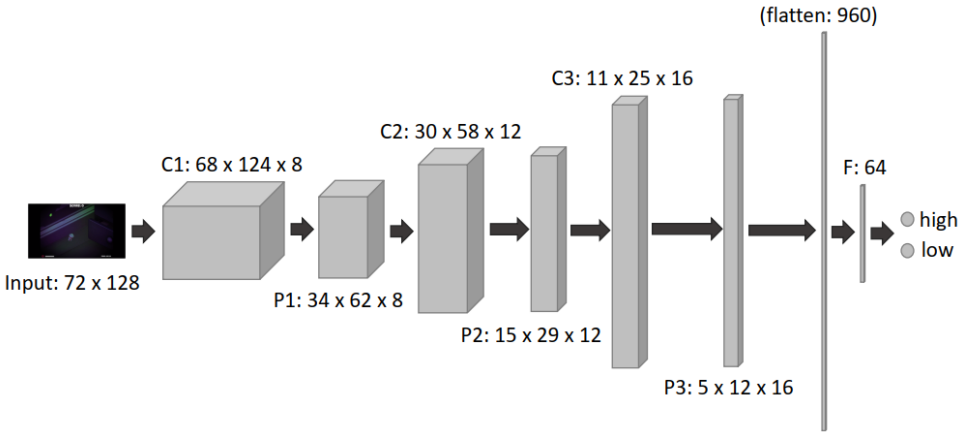


Figure 3.6: One of the CNN architectures proposed by Makantasis et al. "C": Convolutional layers, "P": Max Pooling layers, "F": Fully Connected layers

Furthermore, the same authors two years later expanded their research by proposing a similar model including also audio features [61]. Their work was evaluated on four different video games, and among their findings they highlighted that the arousal prediction accuracy depends on whether the audiovisual feedback of the game represents appropriately the gameplay context. To further explain this concept, researchers showed how the approach to sound design differs between the four games involved in the study: one of them was a horror game named *Sonancia* and its only real sound effect consisted of a low-volume growl, reproduced each time a monster saw the player. In addition to that, there was a constant background audio which changed randomly depending on the current room the player was exploring. As a consequence, the authors observed a poor performance of their model with that specific game and they emphasized how the second audio component could have acted as a confound, worsening the model predictions. To our knowledge, these are the only articles that take this approach for emotion detection applied to video

games.

3.2.3. Expanding Horizons: Emotion Detection applied to Movies

Since the literature considering valence-arousal estimation from gameplay videos is limited to the works presented in [60, 61], we also considered literature concerning the estimation of V-A on videos from movies, which are the most similar media compared to video games.

In 2020, Thao et al. [95] proposed multiple adaptations of a self-attention based network for forecasting emotions in movies. In their approach they extract both video and audio information using multiple pre-trained convolutional neural networks. These architectures include the ResNet-50 [44] and RGB-stream I3D network [25] for appearance feature extraction, FlowNet Simple [33] for motion features, and the VGGish neural network [46] and the OpenSMILE toolkit [34] for audio feature extraction. After training and evaluating two variants of their model on two different movie datasets (COGNIMUSE [103] and Mediaeval 2016 [31]), the authors observed that their performance metrics were better when using only audio audio information, compared to using video features alone (see Figure 3.7). Consequently, they suggested that audio exerts a greater influence on the elicited emotion compared to video.

	Arousal		Valence	
Models	MSE	PCC	MSE	PCC
Feature AAN (only video)	0.933	0.350	0.764	0.342
Feature ANN (only audio)	1.111	0.397	0.209	0.327
Feature ANN (video and audio)	**0.742**	**0.503**	**0.185**	**0.467**
Temporal ANN (only video)	1.182	0.151	0.256	0.190
Temporal ANN (only audio)	1.159	0.185	0.225	0.285
Temporal ANN (video and audio)	**0.854**	**0.210**	**0.218**	**0.415**
Liu et al. [56]	1.182	0.212	0.236	0.379
Chen et al. [55]	1.479	0.467	0.201	0.419
Yi et al. [22]	1.173	0.446	0.198	0.399
Yi et al. [41]	**0.542**	**0.522**	**0.193**	**0.468**

Figure 3.7: Accuracy of models proposed in [95], tested with three variants: video features only, audio features only, all features.

This is not the only approach based on attention networks, in fact Ou et al. [71] in the same year proposed a multimodal local-global attention network, named "MMLGAN", for the same application. Also in this case multiple datasets were tested, including LIRIS-ACCCEDE [21], Mediaeval 2015 [89] and Mediaeval 2016 [31]. As a last example, Wang, et al. [99] in 2022 investigated how to improve the fusion of audio and video representations,

proposing a novel framework and validating it on the LIRIS-ACCEDE dataset.

3.2.4. Affective Datasets of audiovisual content

The choice of training dataset is always a crucial point in the implementation of Deep Neural Networks. In fact, these models often require a large amount of ground truth data in order effectively learn the desired behaviour (supervised learning). In the context of affective computing, more challenges arise since human emotions are highly subjective, making it difficult to collect large and reliable amounts of affective annotations that can be adequate for being used as ground truth [21]. Additionally, the distribution of existing annotated datasets is sometimes restricted due to copyright issues related to video clips.

A recent article published in 2022 by Melhart et al. [64] presented a review of all existing affective datasets of video and video game content, which is summed up in Figure 3.8. In the same paper, the authors proposed a new dataset named "Arousal Video Game AnnotatIoN" (AGAIN), an extensive collection of affective data that includes more than 1100 gameplay clips, along with accompanying gameplay data. These videos were sourced from nine distinct games and have been annotated by 124 participants to indicate Arousal levels in a continuous manner. Each game was developed specifically for the annotation experiment, focusing on creating a pleasant aesthetic and an intuitive gameplay. Also, the authors decided to represent in their games three of the most popular game genres: Racing, Shooters, Platformers. As a result, after concluding the annotation experiments the researchers collected 1116 videos, each one approximately 2 minutes long, resulting in 37 hours of game footage. To our knowledge this is the only currently existing video game dataset with this amount of footage and emotion annotations, although it lacks other affective dimensions such as Valence.

Shifting our focus to movies, a relevant dataset named LIRIS-ACCEDE was proposed in 2015 by Baveye et al. [21], consisting of 9800 short video excerpts extracted from 160 movies. Each video is associated with Valence and Arousal global annotations, while its duration ranges from 8 to 12 seconds, with an overall dataset footage of 27 hours. Affective annotations were collected from a large group of people from different cultures using a crowdsourcing platform. The annotations were made by comparing pairs of videos, which ensured a high level of consistency that was also confirmed by an high inter-annotator agreement. The movies used to create the database are all available under Creative Commons licenses, meaning that they can be freely used, shared, and adapted for research and other purposes. Moreover, the movies in the dataset are classified into nine representative movie genres: comedy, animation, action, adventure, thriller, documentary, romance,

drama, and horror.

	Elicitation				Participants			Annotation			
Database	Interactive	Type	Items	Video	Number	Modalities	Perspective	Type	Labels	Annotators	Tasks
MAHNOB-HCI [38]	No	Video	20 videos	20 hours	30	EEG, ECG, EDA, temp., resp., face and body video, gaze, audio	First-person	Discrete (9-step)	Arousal, valence, dominance, emotional keywords, predictability	self-report	20
DEAP [39]	No	Video	40 videos	40 mins	32	EEG, BVP, EDA, EMG, temp., resp., face video	First-person	Discrete (5-step)	Arousal, valence, dominance, liking, familiarity	self-report	40
LIRIS-ACCEDE [40]	No	Video	9,800 videos	27 hours	N/A	N/A	First-person	Pairwise	Arousal, valence	1517 (arousal) 2442 (valence)	UNK
Aff-Wild [41]	No	Video	298 videos	30 hours	200	N/A	Third-person	Continuous bounded	Arousal, valence	6-8	298
AffectNet [42]	No	Image	450,000 images	N/A	N/A	N/A	Third-person	Continuous bounded, categorical	Arousal, valence, 8 emotion categories	12	137,500
Sonancia [18]	No	Audio	1280 sounds	N/A	N/A	N/A	First-person	Pairwise	Arousal, valence, tension	UNK	10
SEWA DB [43]	Yes	Video	4 videos 1 task	27 hours 17 hours	398	Facial landmarks, FAU, hand and head gestures	Third-person	Continuous bounded	Arousal, valence (dis) liking intensity, agreement, mimicry	5	90
RELOCA [44]	Yes	Video	1 task	4 hours	46	ECG, EDA, face video, audio	Third-person	Continuous bounded	Arousal, valence	6	23
GAME-ON [45]	Yes	Social game	1 game	11.5 hours	51	Video, audio, and motion capture data	First-person	Discrete (5–9-step)	Emotions, cohesion, warmth, competence, competitivity, leadership, and motivation	self-report	5
MUMBAI [46]	Yes	Board-game	6 games	46 hours	58	Gameplay, facial video, and facial action units	First-person and Third-person	Discrete labels	Valence, attention, gameplay experience, personality	56 (Third-person) 58 (First-person)	6
MazeBall [47]	Yes	Videogame	1 game	N/A	36	BVP(HRV), EDA, game telemetry	First-person	Pairwise	Fun, challenge, frustration, anxiety, boredom, excitement, relaxation	self-report	1
PED [48]	Yes	Videogame	1 game	6 hours	58	Gaze, head position, game telemetry	First-person	Discrete (5-step), pairwise	Engagement, frustration, challenge	self-report	1
FUNii [49]	Yes	Videogame	2 games	N/A	190	ECG, EDA, gaze and head position, controller input	First-person	Continuous, discrete	Fun (cont.), fun, difficulty, workload, immersion, UX	self-report	2
AGAIN	**Yes**	**Videogame**	**9 games**	**37 hours**	**124**	**Game video, game telemetry**	**First-person**	**Continuous unbounded**	**Arousal**	**self-report**	**9**

Figure 3.8: A Survey of Affective Datasets of Audiovisual Content, from [64]

.

In the following years, the LIRIS-ACCEDE dataset was further expanded with other collections, like the "Continuous LIRIS-ACCEDE collection" presented by Li et al. [57], which consists of 30 movies with continuous Valence and Arousal annotations. Later works presented under the key name ”MediaEval" focused on different tasks, like violent scenes detection [89].

Another interesting work was presented in 2017 by Zlatintsi et al. [103] who introduced COGNIMUSE, a multimodal video database with different kinds of annotations, includ-

ing Saliency (i.e. video elements where the viewer focuses instantly or over time), Audio Events, Visual Actions, Cross-media semantics, Emotions. Specifically, the emotion annotation task was conducted on seven Hollywood movies, generating as a result continuous Valence and arousal annotations for each movie.

3.3. Deep Learning and Conditional Music Generation

This section begins with a presentation of the reasons behind the growing interest towards generative models (subsection 3.3.1). Then, the most relevant publications are listed and discussed, with a specific focus on affective music conditioning (subsection 3.3.2). Finally, the currently available affective datasets of symbolic music are reviewed (subsection 3.3.3).

3.3.1. Generative models and Controllability: an introduction

Deep learning, which traditionally has been mainly used for classification, prediction and translation problems, recently has started to be applied also to generative tasks [24]. To make a few examples, text-to-image models like *DALL·E 2* can create authentic and realistic images based on a textual description. Then, regarding music a notable text-to-audio model is *Jukebox*, which creates songs with lyrics in the raw audio domain, based on given prompts [41]. One reason for the rising popularity of these applications is that nowadays powerful GPUs have become more affordable and widespread, allowing a wider number of researchers to train and test their models. Secondly and most importantly, these architectures have demonstrated to be able to autonomously learn complex information from any dataset provided, and then to use that knowledge to generate the desired responses with astonishing results. These characteristics are allowing researchers to have a data-driven approach to their goals, obtaining more versatile architectures that can be applied to a wider range of situations. On the contrary, Deep learning models still have many open issues regarding control and interactivity, for example there are no direct ways to control the generation process, even though there are many promising approaches aimed at overcoming these issues.

Focusing on symbolic music generation models, a common technique for conditioning the generation process consists of taking a user-specified music sequence, feeding it to the DL model as priming sequence and making the model generate a continuation (this technique is discussed in chapter 2, subsection 2.5.3). However, this approach alone cannot guarantee that the desired peculiarities of the priming sequence will be maintained as the generation

progresses [88].

Since this thesis specifically involves music generation controlled through emotions, the next subsection will present current state-of-the-art methods for affective music conditioning, i.e. generative models able to generate music that elicits a desired emotion.

3.3.2. Affective music generation models

In recent years the Music Transformer architecture has been used by many researchers in order to autonomously generate music with different goals and constraints. Sulun et al. [92] in 2022 conditioned a music transformer in order to generate midi multi-instrument music with controllable valence-arousal values. The training was performed on 174 270 MIDI files, obtained from the Lakh MIDI dataset (LMD)[79], which were associated to valence and arousal labels, obtained with the Spotify for Developers - Web API. The authors proposed four variants of the same model and evaluated each of them, observing that the one named *continuous-concatenated* outperformed current state of the art methods. The pre-trained models, as well as the code and datasets have been made publicly available for future research. A more in-depth examination of this article is provided in chapter 4 at section 4.2.

In the same year, Zheng et al. proposed EmotionBox [102], a music generation system that, using a Recurrent Neural Network, generates music conditioned on a desired emotion. The peculiarity of this approach is that it does not need a training dataset with ground-truth affective labels, since the emotion conditioning was implemented according to music psychology notions. In particular, the authors automatically extract from each training MIDI file the pitch histogram and the note density, which are used by the model for learning the desired conditioning. In fact, from this information it is possible to inference the musical mode (major or minor) and tempo, that according to multiple studies can be effectively used to elicit different emotions depending on their configuration. As a result, the researchers implemented and evaluated a model able to generate music belonging to each of the four quadrants of valence-arousal plane (depicted in Figure 3.9), observing notably superior performance for low-arousal high-valence compositions, with respect to an analog emotion-label-based method used as ground truth.

Figure 3.9: Circumplex model of affect limited to discrete values (high-low) for both dimensions. As a consequence, emotion modeling is limited to four quadrants (Q1: high valence - high arousal, Q2: low valence - high arousal, etc...).

Ferreira et al. published multiple articles regarding this topic [35, 36]: one of their latest works describes the use Monte Carlo Tree Search (MCTS) as a selection mechanism for maintaining a desired emotion during symbolic music generation [36]. Precisely, the authors generate symbolic music by training a Neural language model (LM), and condition the generation process by additionally training an Emotion Classifier and a Music Discriminator, following a similar approach to Generative Adversarial Networks [40]. The Emotion Classifier was trained by fine-tuning a linear Transformer with the labelled pieces of the VGMIDI dataset [35], obtaining a model able to map any symbolic music piece to one of the four quadrants of the Circumplex model of affect (Figure 3.9). On the other hand, the Music Discriminator aims at increasing the generated music quality and is trained in order to discrimitate between original pieces and LM's compositions. Finally, their proposed MCTS algorithm, which guides the final music generation, uses a value function that jointly employs the Emotion Classifier and the Music Discriminator, hence improving the probability distribution over symbols given by the LM while accounting for the target emotion and leading to pieces that are more realistic.

This is the first approach employing an heuristic search algorithm for the currently discussed application, and according to the authors' experimental evaluation their model outperforms similar baseline techniques. However, researchers highlighed that their choice

to limit the emotion conditioning to only four quadrants was forced by currently available datasets. On top of that, since most of the discussed models require emotionally labelled pieces, a review of affective datasets of symbolic music will be crucial in order to guide our research.

3.3.3. Affective Datasets of symbolic music

In a 2019 article by Ferreira et al. [35] the authors observed an absence of emotionally labeled symbolic music datasets, facing a significant limit for their research. As a consequence, they created and publicly shared a new dataset named VGMIDI, composed of music pieces sourced from video game soundtracks in MIDI format. The choice of video game music was motivated by its peculiar application, since generally the composer aims specifically at keeping the player in a certain affective state, therefore the resulting pieces are less subjective in terms of emotions.

Currently, it comprises 200 annotated pieces, each one with valence and arousal values gathered from 30 human subjects according to the Circumplex model of emotion using a custom web tool (Figure 3.10), and 3850 unlabeled pieces.

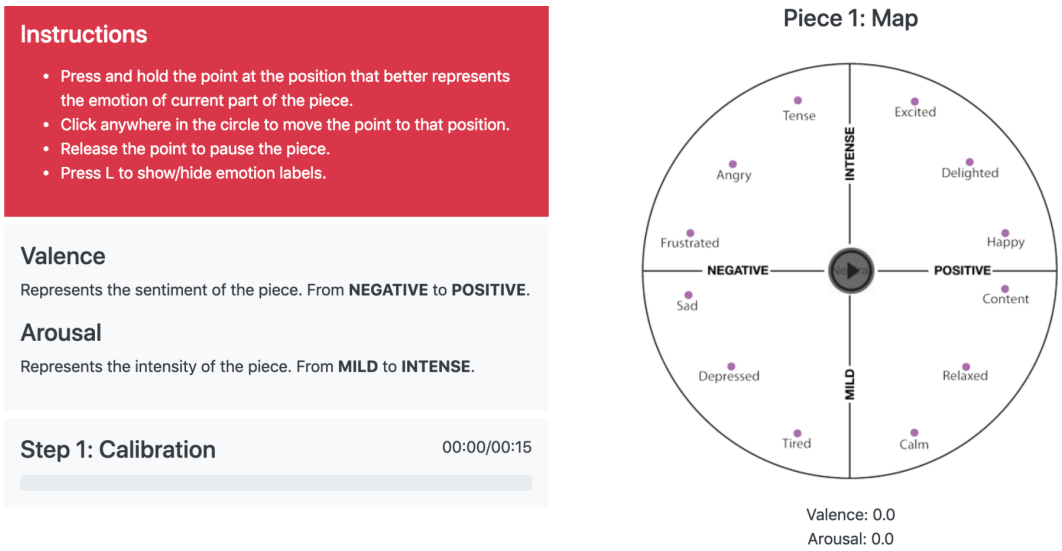


Figure 3.10: Emotion annotation task conducted for creating VGMIDI dataset. This image depicts the instructions provided to each participant (left) and the actual interface used to annotate each piece (right).

Followingly, in 2021 Hung et al. proposed EMOPIA [49], a multi-modal database of both audio and MIDI music consisting of 1087 music clips from 387 songs and clip-level emotion labels. Differently from VGMIDI, this dataset only provides discrete annotations

(high-low) for both dimensions, as previously shown in Figure 3.9.

Name	Label type	Genre (or data source)	Size	Modality
Jamendo Moods [24]	adjectives	multiple genres	18,486	Audio
DEAM [25]	VA values	(from FMA [26], Jamendo, MedleyDB [27])	1,802	Audio
EMO-Soundscapes [28]	VA values	(from FMA)	1,213	Audio
CCMED-WCMED [29]	VA values	classical (both Western & Chinese)	800	Audio
emoMusic [30]	VA values	pop, rock, classical, electronic	744	Audio
EMusic [31]	VA values	experimental, 8 others	140	Audio
MOODetector [6]	adjectives	multiple genres (AllMusic)	193	Audio+MIDI
VGMIDI [10]	valence	video game	95	MIDI
EMOPIA (**ours**)	Russell's 4Q	pop (piano covers)	1,078	Audio+MIDI

Figure 3.11: Table comparing some existing emotion-labeled music datasets contained in [49]. Note that VGMIDI was expanded after this publication, so the details provided here are not up to date.

A recent advance in terms of dataset dimension was made in 2022 by Sulun et al. [92], who proposed the Lakh-Spotify dataset. As described inside their article, a subset of the Lakh MIDI dataset [78] was successfully coupled with valence-arousal continuous values by using the Spotify for Developers API. Overall, a total 34 791 songs with an average duration of 223 seconds compose the largest affective dataset of symbolic music currently available, to the best of our knowledge.

3.4. Conclusive Remarks

In this chapter, we initially investigated the role of generative music for video games, defining its properties and describing the main techniques employed over the years. Then, we delved into the core of our work, analyzing state-of-the-art models for video emotion analysis and affective music generation. Specifically, we summarized strengths and limitations of each mentioned technique, hence clarifying the key elements that motivate our research. Additionally, we briefly reviewed the currently available datasets of both symbolic music and videos annotated with valence-arousal emotions, as they constitute a crucial element inside our proposed approach.

4 | Proposed Approach

In the previous chapters, we have outlined the limitations of current music systems for video games. In fact, many commercial games that adopt a generative music systems can be more accurately described as arrangement systems, since instead of generating entirely new music they recombine pieces or fragments produced by a composer. On one hand, this approach has been successful for the past three decades, and the popularity of many games employing it attests to its quality. On the other hand, there are certain drawbacks to this method. First, hiring a composer can often be cost-prohibitive for small companies, especially when additional professionals like musicians and audio engineers are required. Second, while this approach mitigates music repetitiveness, which can be a concern in certain genres, it's still very limited and does not exploit the variety of situations offered by open-world games. On top of that, research has demonstrated that players highly value music that coherently aligns with the game they are playing. They are even capable of perceiving whether the soundtrack matches the events they are experiencing [73]."

Following these considerations, we propose a novel approach to music generation for video games, which consists of a music system that generates original music coherent with the emotions elicited by the video stream of the game. More specifically, through a 3D CNN we extract valence-arousal information from the gameplay video, which is then used to condition how the music is composed. In section 4.1 we will present the Neural Network used to perform the emotion detection task, while in section 4.2 we will describe the architecture chosen for the music generation task. Lastly, section 4.3 will illustrate the overall architecture and outline its working pipeline.

4.1. Video emotion Detection

This section will be devoted to analyze in depth the first main component of our work, i.e. the video emotion detection. After discussing the design choices in subsection 4.1.1, in subsection 4.1.2 we will describe the complete pre-processing pipeline that is applied to the training dataset. Then, subsection 4.1.3 will present a detailed examination of our proposed model for emotion-from-video estimation, including a visual scheme of the

overall architecture.

4.1.1. Design Choices

Focusing on emotion detection in video games, we have seen different approaches proposed in recent years. Publications involving the measurement of physiological features of the player, like galvanic skin response or heartbeat rate, go beyond our desired application, since their implementation would require the use of specific devices, increasing costs and preventing us to make our strategy more accessible to players and developers. Other approaches like [18, 50, 75] work by mapping game events to an emotional response, but usually these mappings are arbitrarily determined and are not easily transferable to games with different design or belonging to other genres. Finally, we have highlighted the lack of models able to predict both Valence and Arousal solely from video stream of a game. The closest results were achieved by Makantasis et al. in [60, 61], but as a design choice they only aimed at predicting the Arousal affective dimension, which alone cannot fully represent Circumplex Model of emotions [83]. Nevertheless, we believe that building a Deep Neural Network with a similar strategy will make our proposed approach versatile and less influenced by arbitrary assumptions, since the network itself will learn from a large amount of data how to assign emotions to video frames. In addition, we chose valence-arousal modeling since its employment in recent generative music architectures (e.g. [92, 102]) has provided impressive results for conditioned generation tasks. A scheme of the approach proposed for this first section is illustrated in Figure 4.1.

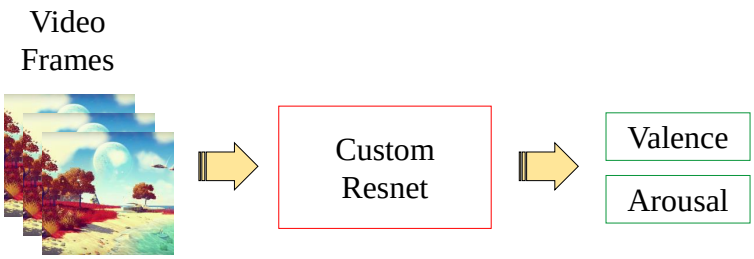


Figure 4.1: Scheme of the proposed approach for video emotion detection.

Formally, we define a video frame as

$$f = \left[\begin{bmatrix} a_{1,1,R} & \dots & a_{1,w,R} \\ \dots & \ddots & \dots \\ a_{h,1,R} & \dots & a_{h,w,R} \end{bmatrix} \begin{bmatrix} a_{1,1,G} & \dots & a_{1,w,G} \\ \dots & \ddots & \dots \\ a_{h,1,G} & \dots & a_{h,w,G} \end{bmatrix} \begin{bmatrix} a_{1,1,B} & \dots & a_{1,w,B} \\ \dots & \ddots & \dots \\ a_{h,1,B} & \dots & a_{h,w,B} \end{bmatrix} \right], \tag{4.1}$$

where $h, w \in \mathbb{N}$ represent the height and width of the frame, R, G, B are the three channels of RGB color model, and $a \in [0, 1] \subset \mathbb{R}$ indicates the value of each pixel for each color channel.

Hence, we can describe our proposed model as a function $\mathcal{S}_1$ that, given as input a sequence of video frames $\mathbf{f} = [f_1, \dots, f_N]$ returns a sequence of two continuous values V and A, representing respectively the predicted Valence and Arousal. Consequently, our system can be modeled as

$$[V, A] = \mathcal{S}_1(\mathbf{f}), \tag{4.2}$$

where $V \in [-1, 1] \subset \mathbb{R}$ and $A \in [-1, 1] \subset \mathbb{R}$.

4.1.2. Dataset Pre-processing

For training our model we used a Dataset composed by short video excerpts of movies associated with Valence and Arousal global annotations for each clip.

First of all, the annotation values of each feature were normalized in order to lie in the desired interval $[-1, 1]$. We performed the conversion by first applying a min-max normalization, which by definition rescales input values in a range $[0, 1]$, and then performing a linear conversion, which finally achieves our desired result.

Formally, given $\mathbf{x} = [x_1, ..., x_N]$ the sequence of length N containing all annotations of a single affective dimension, we extract the global $min(\mathbf{x})$ and the global $max(\mathbf{x})$, that are respectively the minimum and maximum value across all the annotations. As a result, $\forall i \leq n$ we apply the "min-max" normalization formula

$$Norm(x_i) = \frac{x_i - min(\mathbf{x})}{max(\mathbf{x}) - min(\mathbf{x})}, \tag{4.3}$$

and then we scale the normalized values from $[0, 1]$ to $[-1, 1]$ applying the following formula,

$$Scaled(x_i) = (x_i - Min_{\text{old}}) * \frac{Range_{\text{new}}}{Range_{\text{old}}} + Min_{\text{new}}, \tag{4.4}$$

where in our case $Min_{\text{old}} = 0$, $Max_{\text{old}} = 1$, $Min_{\text{new}} = -1$, $Max_{\text{new}} = 1$ and both ranges are determined as $Range = Max - Min$.

Moving to the videos composing the training dataset, for each file we import a subset of its frames and resize them to a desired resolution. Resizing the video frames is crucial in order to reduce the overall dataset size and to allow a higher number of videos to be processed without consuming all the virtual memory.

4.1.3. Architecture

Following the approach proposed by Makantasis et al. [60], we implemented a 3D Convolutional Neural Network for the task of video emotion prediction. We based our model on the architecture proposed in [96], where the authors conducted a comparative analysis of multiple variations of 3D ResNets, proposing that 3D convolutions could be effectively approximated by a 2D convolution followed by a 1D convolution, separating spatial and temporal modeling into two distinct stages. Consequently, we implemented their proposed block in our architecture with a layer named **Conv2Plus1D** (see Figure 4.2).
This (2+1)D decomposition, when compared to full 3D convolution, offers two distinct advantages. Firstly, without altering the number of parameters, it effectively doubles the number of nonlinearities in the network by introducing an additional ReLU activation layer between the 2D and 1D convolutions. This increase in nonlinearities enhances the network's capability to represent complex functions. Secondly, by separating the 3D convolution into spatial and temporal components, it simplifies the optimization process. As a result, with this approach a model reaches lower training error compared to 3D convolutional networks of equivalent capacity.

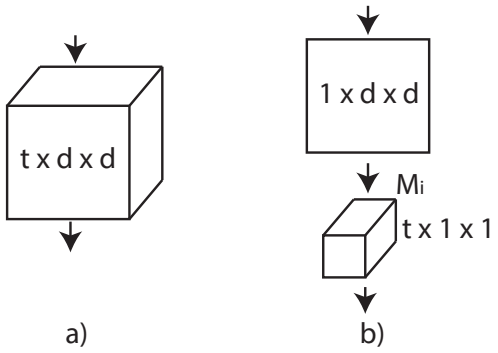


Figure 4.2: **(2+1)D vs 3D convolution**. This picture from [96] illustrates a simplified scenario where the input is a sequence of frames containing only one color channel. In this comparison, (a) represents a full 3D convolution performed using a filter of dimensions t × d × d, where t represents the number of frames and d corresponds to the spatial width and height. On the other hand, (b) depicts the (2+1)D convolutional block, where the same computation is split into a spatial 2D convolution followed by a temporal 1D convolution.

As a consequence, we based our proposed emotion estimation model on a ResNet18 [44], where each convolutional layer is replaced by a Conv2Plus1D convolution. A ResNet model is constructed by combining a series of residual blocks. Each residual block consists of two branches: the main branch, responsible for performing calculations, and the residual branch, which bypasses the main calculation and adds the input to the output of the main branch. While the main computations are performed in the main branch, the residual branch offers an easier path for gradients to propagate. As a result, a direct connection from the loss function to any main branch within a residual block is established, effectively mitigating the vanishing gradient problem. An illustration of the residual block is provided in Figure 4.3

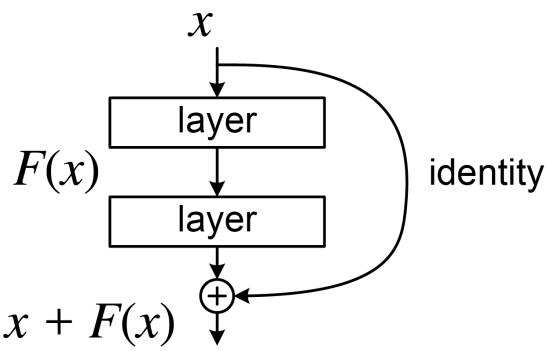


Figure 4.3: A residual block from [44].

Our proposed Convolutional Neural Network is composed of the following layers:

- **Conv2Plus1D**: A 3D convolutional layer that can be modeled according to the following formula

$$x_o = Conv2plus1D(x_i, Filter) + Bias, \tag{4.5}$$

with x_i being the input tensor and x_o the output tensor. *Filter* refers to the learnable convolutional filters, *Bias* represents the bias term associated with each filter.

- **Residual Block**: Defining x_i and x_o respectively the input and output tensor of the layer, the residual block can be formalized as

$$x_o = x_i + \mathcal{F}(x_i; \theta), \tag{4.6}$$

where $\mathcal{F}(;\theta)$ performs the composition of two convolutions parameterized by weights θ and the application of the ReLU functions, as depicted in Figure 4.3.

- **ResizeVideo**: This layer resizes the input video according to a desired output dimensionality. Formally, it can be defined as

$$x_o = Resize(x_i, [height_o, width_o]), \tag{4.7}$$

where $height_o$ and $width_o$ represent the target output dimensions of the layer. Downsampling the video frames enables the model to focus on specific regions within the frames, enabling the detection of specific patterns. Moreover, this process leads to dimensionality reduction, resulting in faster processing within the model.

- **BatchNormalization**: this layer performs a transformation that keeps the output's

mean near 0 and its standard deviation close to 1. When the network is being trained, the layer performs the following operation

$$batch = \gamma * \frac{batch - mean(batch)}{\sqrt{var(batch) + \epsilon}} + \beta, \tag{4.8}$$

where ϵ is a small constant, β and γ are learned parameters named respectively scaling factor and offset factor, while $var()$ computes the variance of current batch and $mean()$ trivially returns its mean.

- **ReLU**: Rectified Linear Unit, it's an activation function that receives as input the previous layer and returns element-wise $max(x, 0)$.
- **Global Average Pooling 3D**: a Pooling layer down-samples feature maps received as input, so that deeper layers effectively integrate larger extents of data. The Global Average Pooling 3D layer is designed for receiving a 3D input and performs average operation in order to reduce it to a single dimension output. In other words, starting from input feature maps of dimensions $f \times h \times w \times n$ it returns the global average across the first three dimensions, obtaining a sequence of n values (i.e. only last dimension is preserved).
- **Dense layer**: also named Fully Connected layer, it establishes dense connections with its preceding layer, meaning that each neuron in the layer is connected to every neuron in its preceding layer. Formally, it implements the operation

$$x_o = activation(dot(x_i, kernel) + bias), \tag{4.9}$$

where $dot()$ refers to the dot-product between two matrices, $activation()$ is the element-wise activation function passed as argument (linear by default), $kernel$ represents a matrix of weights generated by the layer, and $bias$ is a bias vector also created by the layer.

- **Dropout**: during training, at each iteration the Dropout layer randomly assigns some input units to 0, with a frequency defined by the $rate$ parameter. The remaining non-zero inputs are scaled up by a factor of $1/(1 - rate)$ to maintain the overall sum of inputs unchanged. As a result, the network becomes less sensitive to the precise configuration of individual neurons and it is more likely to generalize well to new data, making this layer a very effective choice for preventing over-fitting.

Since we will perform regression, we set an output Dense layer with 2 units, returning a continuous value for each affective dimension we want to predict (Valence and Arousal).

Lastly, for training the described network we employed the MSE formula as loss function. Formally, let $\hat{\mathbf{y}} = [\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N]$ be the sequence of N predictions performed at each epoch by the network, where $\hat{\mathbf{y}}_i = [\hat{V}_i, \hat{A}_i]$ contains the continuous values for Valence and Arousal, $\forall i \in [1, \ldots, N]$. Hence, we can write the MSE function as

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2, \tag{4.10}$$

where $\mathbf{y}_i = [V_i, A_i]$ indicates the ground truth value of both affective dimensions, compared at each epoch with the predictions $\hat{\mathbf{y}}_i$.

The scheme of our proposed neural network is depicted in Figure 4.4, while the overall training pipeline is represented in Figure 4.5. Finally, a more detailed representation of our model's implementation can be found in the Appendix A.

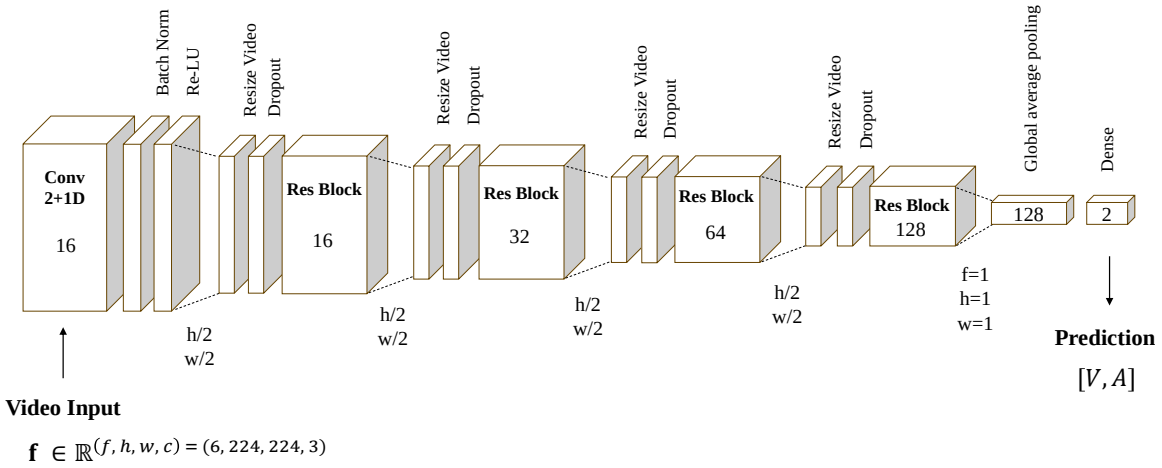


Figure 4.4: Scheme of the custom ResNet proposed for the emotion detection task. This CNN receives an input of dimensionality $f \times h \times w \times c$, where each letter respectively indicates the number of frames, height, width, number of channels. The last dimension, initially representing the RGB channels, indicates the number of feature maps obtained after the convolutional layers, and it is reported inside the layers to illustrate its evolution along the the network. Although the input is $4D$, in this representation we displayed $3D$ blocks, considering that the first dimension f remains unchanged until global average pooling.

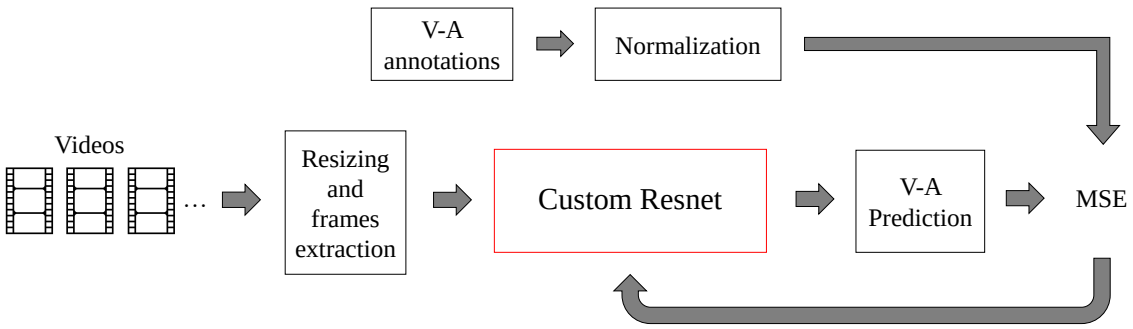


Figure 4.5: Scheme of the overall training pipeline presented in this section.

4.2. Music generation conditioned on emotions

In this section we will concentrate on the second main component of the proposed method, specifically the music generation model. While subsection 4.2.1 will clarify the main design choices that guided our work, in subsection 4.2.2 we will present our chosen model and how the music generation was conditioned on Valence and Arousal values. Finally, in subsection 4.2.3 we will describe how we additionally conditioned our model to generate music starting from a desired input piece (motif continuation).

4.2.1. Design Choices

For the music generation task we relied on the Music Transformer [48], which in recent years proved to be a solid architecture for generating high quality music compositions for a vast amount of genres and styles. We specifically used a pre-trained model proposed in an article by Sulun et al. [92], which was trained on a dataset of pop songs with valence-arousal annotations. Additionally, due to the lack of controllability of transformer models, we modified the previous architecture in order compose new music as a continuation of an input MIDI file. As a result, we obtain the scheme depicted in Figure 4.6.

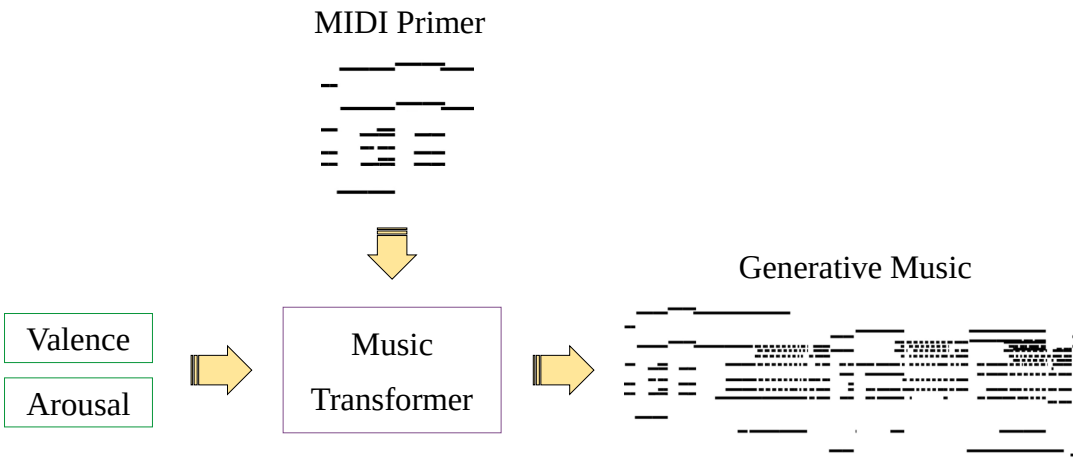


Figure 4.6: Scheme of the proposed approach for conditioned music generation.

Let $\mathbf{m}$ be a sequence of music tokens $\mathbf{m} = [T_1, ..., T_N]$, where each music token T is a tuple of two elements $T = <i, v>$, with i being an index corresponding to a MIDI event, such as DRUMS_OFF or TIMESHIFT, while v determines its correspondent value (e.g. pitch for note or time for timeshift). Formally, we want to build a function $\mathcal{S}_2$ defined as

$$\mathbf{m}_{out} = \mathcal{S}_2([V, A], \mathbf{m}_{primer}), \tag{4.11}$$

where V, A are the conditioning Valence and Arousal values already defined for Equation 4.2, while $\mathbf{m}_{primer}$, $\mathbf{m}_{out}$ are two sequences of music tokens, that correspond respectively to the input conditioning MIDI file and the generated output MIDI file.

4.2.2. Architecture and Emotion Conditioning

As anticipated earlier, at the core of this architecture lies the music transformer, a decoder-only Transformer employing relative position embeddings. Specifically, we employ a pre-trained model with a total of 20 layers and a feature dimension of 768, with each layer characterized by 16 heads and a feed-forward layer with a dimension of 3072. In total, this model has approximately 145 million parameters.

The authors of the model initially trained a music transformer on a total of 96 119 songs belonging to the LPD-5-full subset of the Lakh Pianoroll Dataset (LPD) [32]. As a result, they obtained an unconditioned model that we will name *vanilla model*. Next, to obtain the conditioned models they initially transferred the weights learned by the vanilla model.

Starting from these weights, the current model was fine-tuned using the LPD-5-matched dataset, which specifically consists of 5-instrument piano roll data. This subset of 27 361 songs, each one with Valence and Arousal labels associated, allowed the authors to finally achieve the desired emotion conditioning.

In particular, among three possible conditionings proposed by the authors, we chose the approach named *continuous-concatenated* depicted in Figure 4.7, which was implemented applying the following steps during training:

1. **Input features and music tokens**: During training, the model receives as input a sequence $[V, A]$ containing the two values of Valence and Arousal, as well as a sequence $\mathbf{m}$ of music tokens, both defined as in equations 4.2 and 4.11
2. **Vector padding**: The sequence of emotion features is fed to a linear layer, obtaining a condition vector that has a fixed length of 192, while $\mathbf{m}$ is fed to an embedding layer, obtaining for each music token a music embedding of length 576.
3. **Vector Concatenation**: The single condition vector is repeated and concatenated to each music token embedding, reaching a final feature dimension of 768.

As a consequence, the resulting total feature length of the transformer input is constant for each file. Moreover, the initial emotion features are incorporated into every single embedding of the Transformer's input sequence, fully exploiting the conditioning information. This approach was also the preferred choice for the authors of the study, since it performed better than other variants.

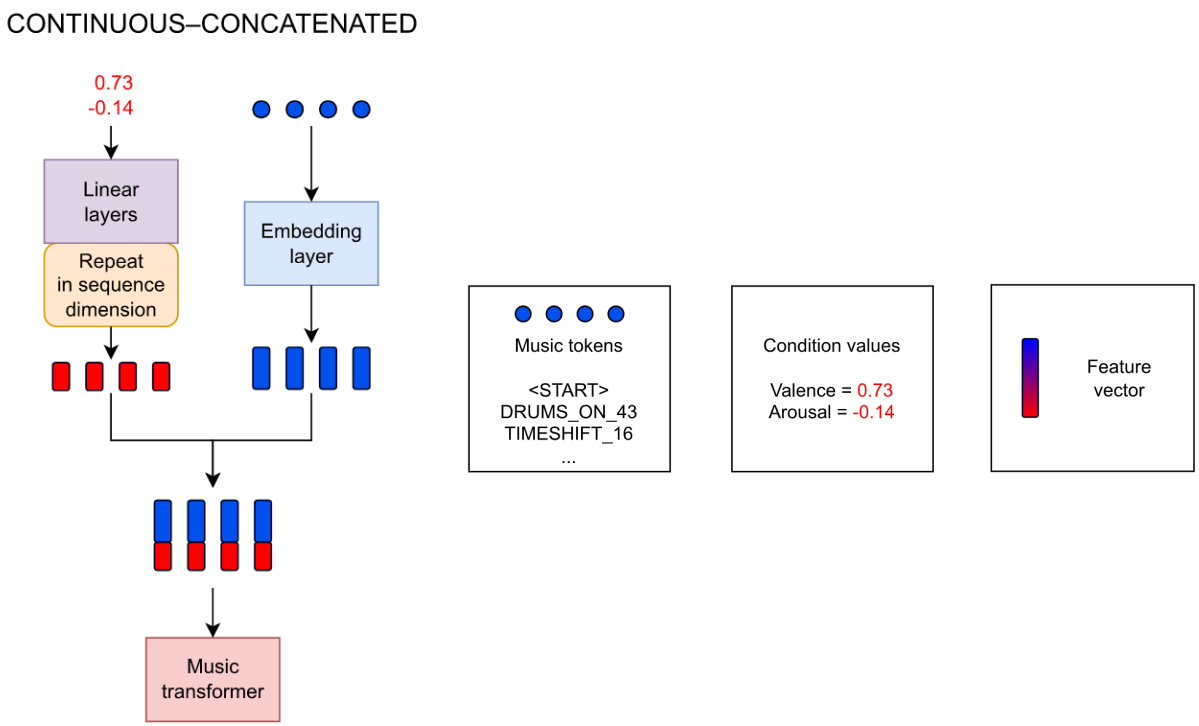


Figure 4.7: Music transformer conditioning process.

4.2.3. Primer conditioning

Due to its design nature, a Transformer model lacks of control over the generation process, meaning that even with the emotion conditioning described in Section 4.2.2 any other musical characteristic of the output is not directly controllable. However, a Music Transformer can be conditioned during inference by using a sub-sequence as a primer [88]. This process (also described in chapter 2, subsection 2.5.3) involves providing the model with a musical fragment as input and predicting its continuation. In this approach, both the condition (primer) and the target (predicted melody) must belong to the same domain. Therefore, we implemented a pre-processing pipeline equivalent to the one used for the training dataset, in order to successfully convert a primer MIDI file to a sequence of music tokens **m**. In practice, starting from a MIDI file containing our desired primer, we perform the following operations:

1. **Import and instrument filtering**: After the MIDI file is imported, all instruments not included in the subset defined in model training (['drums', 'piano', 'strings', 'bass', 'guitar']) are renamed as 'piano' tracks;

2. **trim primer**: the initial MIDI is then trimmed according to a fixed pre-determined length;

3. **convert MIDI to music tokens**: starting from the trimmed MIDI, we generate a sequence of music tokens $\mathbf{m}_{primer}$, already defined for equation 4.11;
4. **feed to the model during inference**: The resulting sequence is finally fed to the transformer, obtaining as output a musical piece based on the initial MIDI.

Figure 4.8 illustrates the overall pipeline presented in this Section.

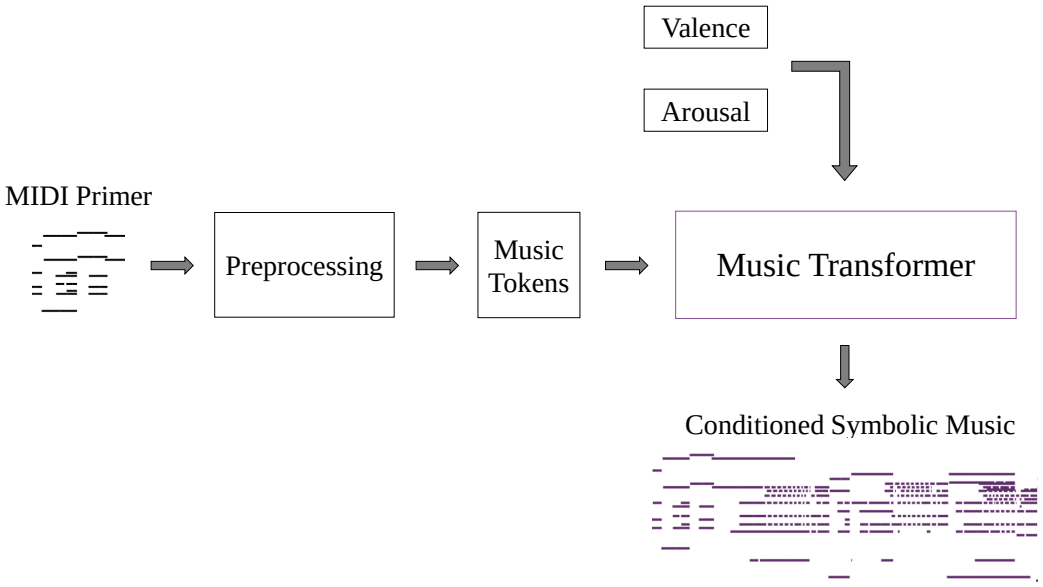


Figure 4.8: Music Generation pipeline.

4.3. Final model

After a detailed description of each component included in the proposed model, we are finally ready to present the resulting overall architecture. Due to the versatility of our proposed approach, we will divide this section in two parts. In the first one (subsection 4.3.1) we will describe the overall architecture in absolute terms, describing its workflow, while in the second one (subsection 4.3.2) we will place our method in the context of video game music systems, highlighting the key features that differentiate it from state-of-the-art approaches.

4.3.1. Overall Architecture

In section 4.1 we presented an architecture that receives video frames as an input and returns two continuous values as defined in Equation 4.2, while in section 4.2 we described

a model able to generate novel multi-instrumental music starting from V-A values and an initial MIDI file, as formalized in Equation 4.11. These two distinct architectures can be easily combined by setting the predicted values of our CNN as the conditioning values of the Music Transformer, as illustrated in Figure 4.9. Formally, we can combine Equations 4.2 and 4.11 obtaining

$$\mathbf{m}_{out} = \mathcal{S}_3(\mathbf{f}, \mathbf{m}_{primer}). \tag{4.12}$$

As a result, we created a system that processes some input video frames, obtains two continuous values representing the emotion elicited by the video and generates music based on this information: in other words, our proposed method aims at generating music emotionally coherent with the input video. The power of this framework lies in its versatility, since by not bounding our method to other specific input sources it can be potentially adapted for any video game with little effort. Furthermore, it could also be applied in other fields involving an evolving visual component, like assisted music composition or media art. For the purposes of this thesis, we will describe and successively evaluate its application as a video game music system.

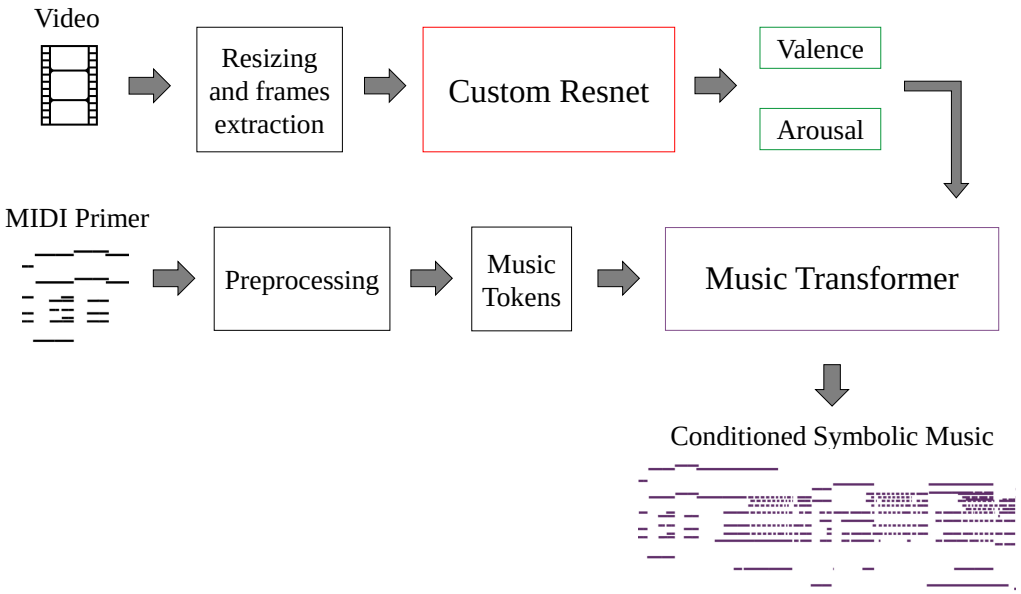


Figure 4.9: V-A estimation and Music generation pipelines combined.

4.3.2. Proposed Architecture as a Video game Music System

Since our goal is to apply our proposed architecture to video games, we will finally analyze it as a generative music system for this medium. First of all, Figure 4.10 shows our proposed model in this context, outlining all the intermediate steps that allow our music system to generate music starting from a user playing a video game.

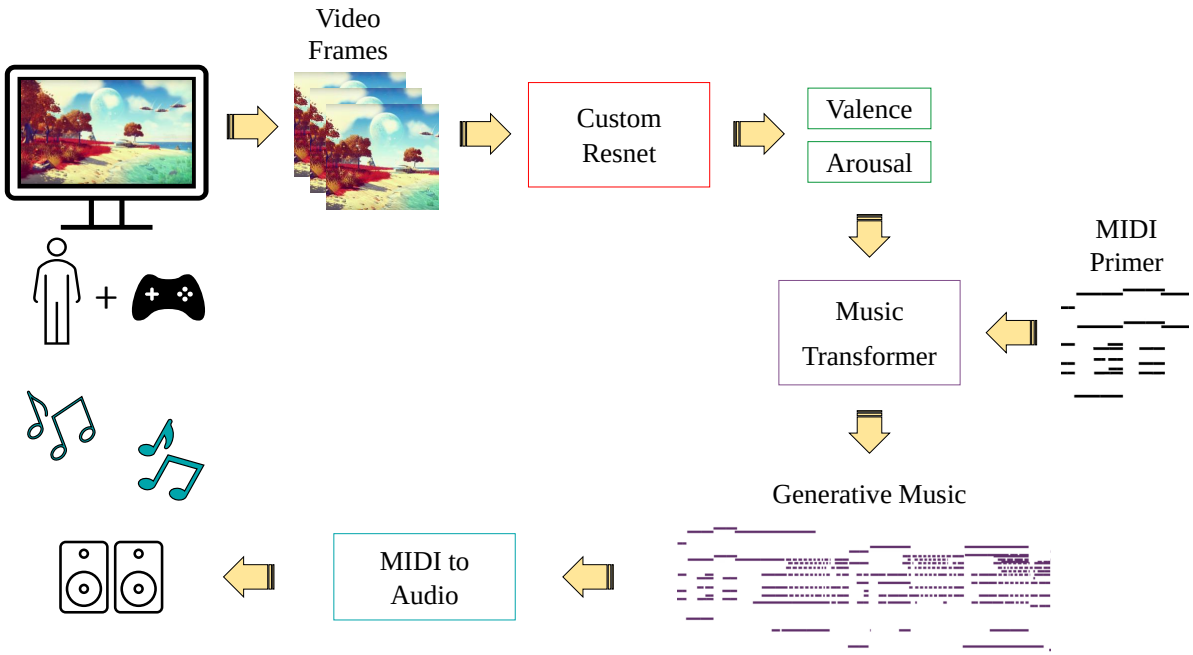


Figure 4.10: Proposed Architecture as a Video game Music System. While the user interacts with a video game, some gameplay frames are extracted and sent to a custom ResNet, that determines the current emotion elicited by the video. Next, according to the predicted emotion a music transformer generates novel music starting from a MIDI primer, that helps controlling the desired style. Finally, the generated symbolic music is synthesized to audio and reproduced for the player. As time goes on, the soundtrack will constantly adapt coherently to the game's visual feedback, enhancing the overall emotional experience

Considering the dimensions presented in chapter 3, subsection 3.1.2, we can now discuss our work according to those definitions.

- **Adaptivity**: Intuitively, our system can be seen as adaptive since its music generation depends on frames taken from the video game's video stream, which is constantly changing depending on player's inputs. Precisely, according to proposition 3.1 one may argue that with this system we are not directly linking any specific game variable to our music system. However, starting from the game's video frames our method determines two continuous values, Valence and Arousal, which represent the current emotional state elicited by the game. From this perspective, our system actually adapts to the game's emotional state, described by those two variables;
- **Generativity**: Moving to the generative dimension, we can more easily define our music system as generative since it composes autonomously novel music, thanks to its transformer model, satisfying proposition 3.2.

4.4. Conclusive Remarks

In this chapter, we've presented a new framework for video game music systems. Our method is designed to produce original music that seamlessly aligns with the emotions evoked by the in-game video stream. In each section we first motivated all the design choices that guided our work, illustrating and formalizing each component. After describing in detail our proposed 3D CNN and the pre-trained music transformer employed, we additionally examined our proposed approach as a whole. With a clear view of the system's architecture and its operational pipeline, we are ready to discuss its experimental evaluation.

5 | Experimental Setup and Evaluation

In this chapter we will present and discuss the results of our experiments, aimed at validating the effectiveness of our proposed approach. Section 5.1 will focus on our proposed architecture for the Valence-Arousal (V-A) prediction task, covering the implementation details and performance results. Then, in section 5.2 we will illustrate the experimental setup defined for the evaluation of our complete framework, finally discussing all the results obtained. All the code written to implement the valence-arousal estimation model [1], to generate the emotion-conditioned MIDI tracks [2] and to analyze the experimental data [3] is publicly available on Github.

5.1. Emotion Estimation Model

In this section we will describe in detail the final implementation of our custom ResNet (presented in chapter 4), defined after testing different parameters, procedures and their relative results. In subsection 5.1.1 we list all implementation details, while the results and their discussion are carried out in subsections 5.1.2 and 5.1.3, respectively.

5.1.1. Dataset and Model Implementation

Ideally, we would have trained our proposed architecture on a dataset composed of gameplay videos associated with V-A labels. However, as discussed in section 3.2.4, currently a dataset satisfying all these ideal requirements does not exist. After analyzing all available options, we decided to train our model on the LIRIS-ACCEDE dataset, which provides both Valence and Arousal annotations for 9800 videos excerpts from movies [21].

Going into detail, every clip contained in the dataset is an mp4 file with resolution 640×376 and 23.98 frames-per-second. For each video, we import 6 frames with a frame step

[1] https://github.com/FrancescoZumo/valence-arousal-video-estimation
[2] https://github.com/FrancescoZumo/midi-emotion-primer-continuation
[3] https://github.com/FrancescoZumo/videogame-procedural-music-experimental-setup

$= 4$ and we resize them to width $= 224$, height $= 224$. We selected a subset of 8000 videos from our train dataset and we split them into Train, Validation and Test sets, with a dataset split policy of 80-10-10.

Since we want to preserve the original distribution of both Valence and Arousal values across each subset created, we perform the following operations. First, we define ten classes corresponding to ten intervals of equal length $[-1, -0.8], [-0.8, -0.6], \ldots, [0.8, 1]$, that together cover the domain $[-1, 1] \subset \mathbb{R}$ of V-A values. Second, for each emotion dimension we assign each video to its corresponding class (i.e. the interval where its affective label lies). Third, we perform the dataset split operation, ensuring that each subset created has a similar number of occurrences for each class and for both affective dimensions.

We implemented our custom ResNet with the Tensorflow library [17]. For training the model we selected the Adam optimizer, setting the learning rate $\text{lr} = 10^{-5}$ and beta parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$. We used the Mean Squared Error defined in equation 4.10 as loss function, which was computed as a global value across both affective dimensions. In order to better monitor the training process, we additionally computed for each epoch the Root Mean Squared Error

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}, \tag{5.1}$$

and the Mean Absolute Error

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\mathbf{y}_i - \hat{\mathbf{y}}_i|, \tag{5.2}$$

where $\mathbf{y}_i = [V_i, A_i]$ is the ground truth value of both affective dimensions, $\hat{\mathbf{y}}_i = [\hat{V}_i, \hat{A}_i]$ contains the two predicted values and N is the number of predictions performed during each epoch. We trained our model for a maximum of 1000 epochs, setting an early stopping behaviour with *patience* $= 100$, monitoring the validation loss. As a result, the training loss is displayed in Figure 5.1

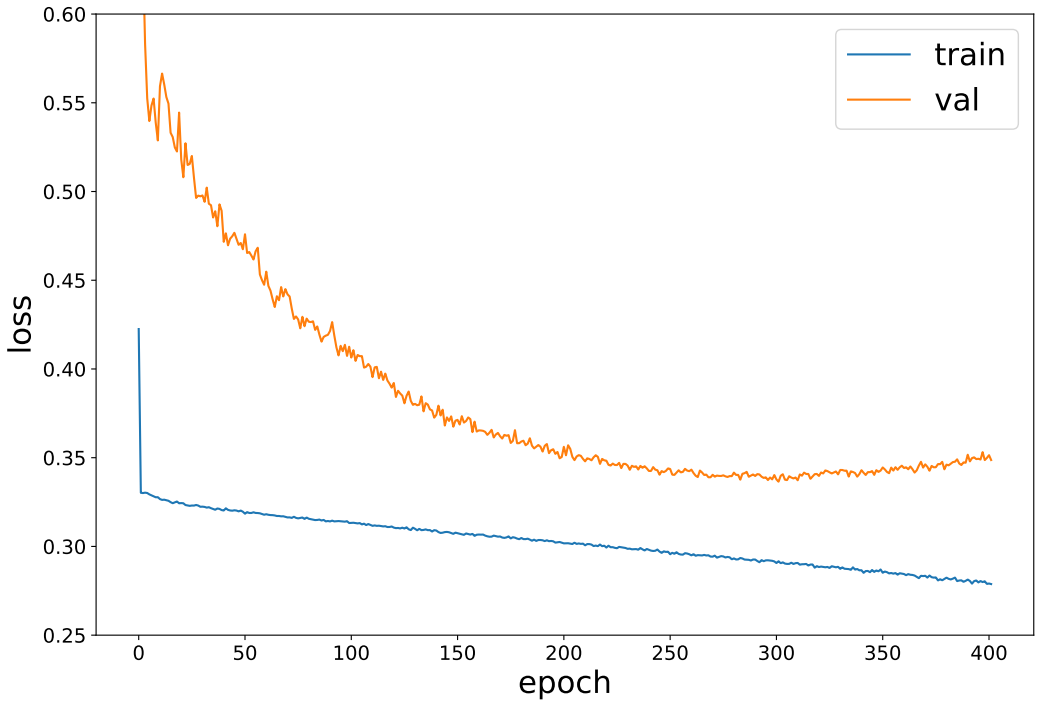


Figure 5.1: Training MSE of custom ResNet

5.1.2. Results

After being trained, our model reached the best performance at epoch 301, with a MSE of 0.291 for Train, 0.338 for Validation and 0.337 for Test sets. The complete performance metrics are reported in Table 5.1, while in Table 5.2 we show performance for the single affective dimensions.

Performance			
Set	MSE	RMSE	MAE
Train	0.291	0.528	0.440
Validation	0.338	0.591	0.483
Test	0.337	0.581	0.489

Table 5.1: Performance of custom ResNet on Train, Validation and Test set

	Valence			Arousal		
Set	MSE	RMSE	MAE	MSE	RMSE	MAE
Test	0.315	0.561	0.485	0.360	0.600	0.493

Table 5.2: Performance of custom ResNet for Valence and Arousal on Test Set

With the obtained model, we conducted two preliminary experiments: one to assess its applicability in a continuous context (predicting time series instead of global values) and another to determine if its learned knowledge can also be applied to video games.

We first analyzed the model's predictions on the *Continuous LIRIS-ACCEDE* dataset [57], an additional collection proposed by the same authors of our training dataset (that we will reference as *Discrete* from now on), comprising 30 short movies with continuous Valence and Arousal annotations. On average, we obtained a MSE of 0.091 for Valence predictions and 0.252 for Arousal: Table 5.3 displays also RMSE and Pearson Correlation Coefficient (PCC) results, which are the most common performance measures for evaluating time series forecasting.

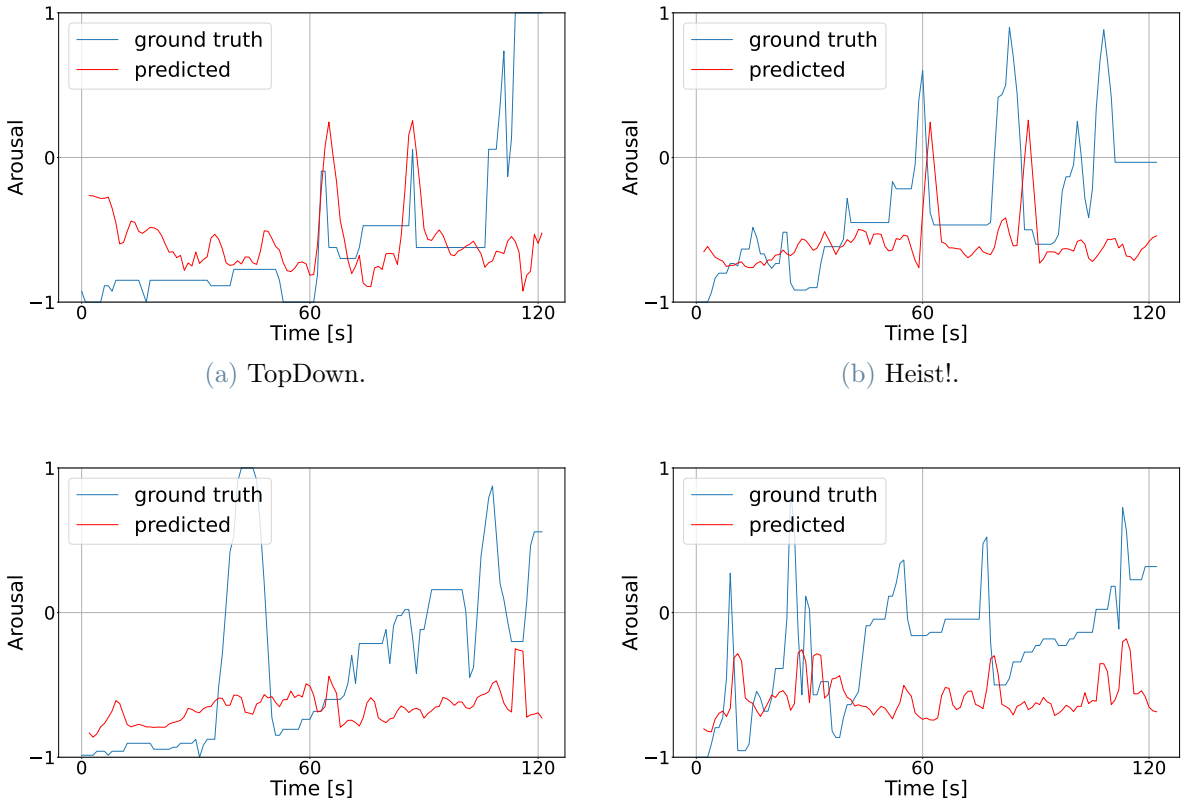
While training our network directly on this continuous version may have seemed a valid choice, it would have introduced a few weakness. Compared to its discrete counterpart, this collection exhibits significantly less variety (30 vs 160 movies represented), limiting our purpose of learning a general visual knowledge (colors, intensity, shapes, etc...) also applicable to video games. As described by researchers, this dataset is best suited for emotion prediction over long movies, where prior scenes can impact the emotional interpretation of subsequent ones. However, since our application involves open-world games with non-linear narratives and non-scripted events, these assumptions do not hold true and the annotations collected could have potentially introduced confounding factors into our problem.

Valence			Arousal		
MSE	RMSE	PCC	MSE	RMSE	PCC
0.091	0.302	0.125	0.252	0.502	0.014

Table 5.3: Performance of custom ResNet on LIRIS-ACCEDE Continuous.

Secondly, we qualitatively evaluated how our model performs on the AGAIN dataset, comparing the ground truth Arousal values (the only annotations available on this dataset,

reason for which we could not train on its videos) with the predicted ones. Specifically, we tested a single gameplay video (duration ≅ 2 minutes) for each of the 9 games composing the dataset. Figure 5.2 shows Arousal predictions on games with a first person perspective, while Figure 5.3 includes two isometric games and Figure 5.4 shows results for 2D side-scrolling games (theoretical background on video game perspectives is provided in section 2.1.2 and Figure 2.1). In all plots we display the moving average (window of 3 seconds) of our model's predictions.



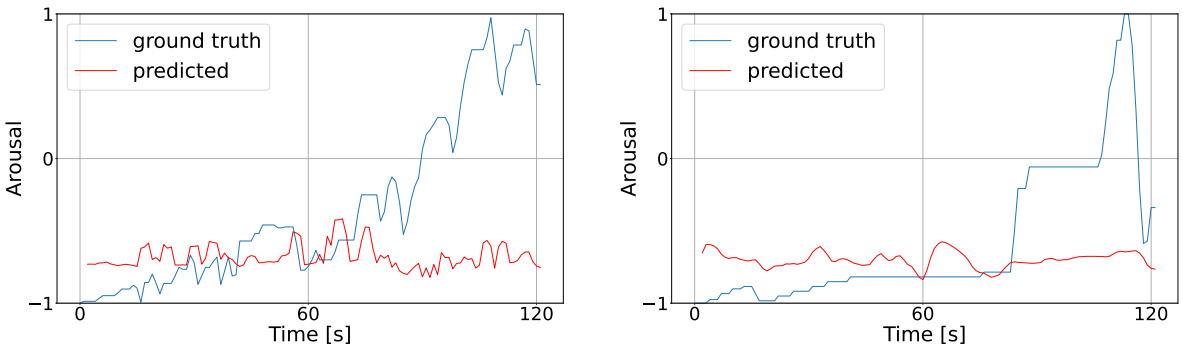
(a) TopDown.

(b) Heist!.

(c) Solid.

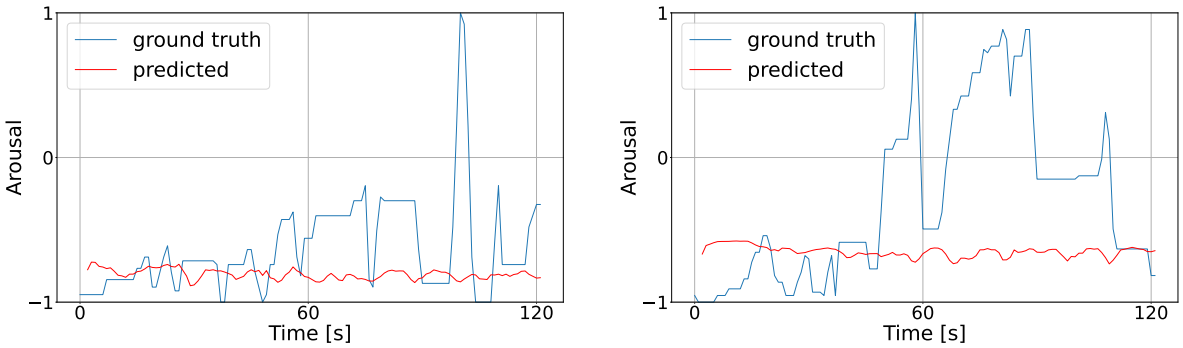
(d) ApexSpeed.

Figure 5.2: AGAIN Games with first person perspective.

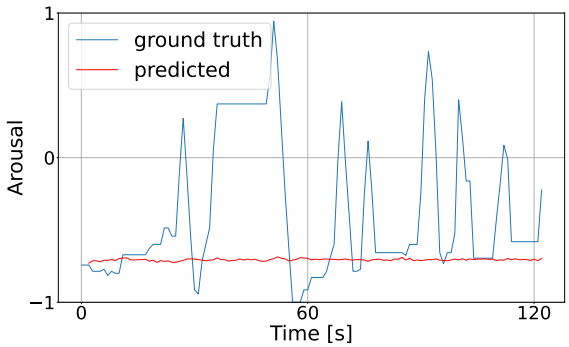


(a) Shootout. (b) TinyCars.

Figure 5.3: AGAIN Games with isometric perspective.



(a) Run'N'Gun. (b) Pirates!.



(c) Endless.

Figure 5.4: AGAIN Games with 2D perspective.

5.1.3. Discussion

In this preliminary evaluation, we can notice that while our model performs similarly for Valence and Arousal in the Test set, the same does not hold when applied to the continuous LIRIS-ACCEDE dataset, where Valence performs significantly better than Arousal for all three metrics. These results are coherent with other studies performed on the LIRIS-ACCEDE collection [95]. Focusing on the inference on the AGAIN dataset, it can be observed how the predicted time series are almost flat for all 2D games, which could be a consequence of their mostly static visual feedback, and the same applies for isometric games. On the contrary, all gameplay videos with a 3D first person view produced a more consistent result, compared to the ground truth.

5.2. Subjective Evaluation

To test the effectiveness of the complete pipeline of our proposed method, we followed an approach similar to [75, 76], where participants were asked to play with a videogame and subsequently annotate the V-A values elicited by gameplay videos soundtracks, obtained with different techniques. Specifically, for this experiment we chose No Man's Sky [12], an action-adventure survival game, characterized by a classic science fiction atmosphere and set in a universe with over 18 quintillion planets (1.8×10^{19}). The player, impersonating a "Traveller", can freely explore an uncharted galaxy with his spaceship, either experiencing infinite freedom or following the story in order to uncover the secrets of the universe.

This game a was chosen for having a first person perspective, which according to our preliminary tests provides the ideal visual feedback for our emotion prediction task. Moreover, one of the key gameplay components of No Man's Sky is its open-world exploration, meaning that players are encouraged to travel, visit and freely explore different planets without forcing a specific goal, making it an ideal game for generating procedural interactive music which is not constrained by a specific narrative or in-game events. In section 5.2.1 we provide a detailed explanation of the experimental setup designed for evaluating of our method, while in section 5.2.2 we analyze and discuss the collected results.

5.2.1. Experimental Setup

For preparing our experimental material, we initially recorded gameplay videos of No Man's Sky and then we extracted 40 videos of $30s$ duration each, which were specifically selected for containing clear visual or emotional changes, based both on our model's predictions and our informal evaluation. These clips were subsequently divided into 4

groups of 10, each corresponding to one of the following categories:

- **A - Conditioned**: videos with original sound effects and generative music conditioned on valence-arousal.
- **B - Unconditioned**: videos with original sound effects and generative music without any conditioning.
- **C - Original**: videos with original sound effects and original music
- **D - None**: videos with original sound effects and no music

Music generation implementation

For each video belonging to category **A - Conditioned**, our custom ResNet predicted a time series for both Valence and Arousal, extracting 6 frames every 1 second and returning the corresponding values. With this information, we want to design an algorithm able to analyze the emotion evolution over time, determining at each time step whether to generate music based current emotion or keep the previous conditioning.

Formally, let $\mathbf{x} = [x_1, \ldots, x_N]$ be a sequence of length N containing the annotations of a single affective dimension, computed for each second s of current video. For both Valence and Arousal, we calculate the *Absolute value of the Difference Quotient* Q^{abs} (which represents how abruptly or smoothly a quantity changes) between each couple of subsequent predictions (x_i, x_{i-1}), defined as

$$Q^{abs}(\mathbf{x}) = \begin{cases} \dfrac{|x_i - x_{i-1}|}{2}, & \forall i : 1 < i \leq N, \\ 0 \qquad\quad , & \forall i : i = 1. \end{cases}, \tag{5.3}$$

obtaining as a result a sequence $\mathbf{q} = [Q^{abs}(x_1), \ldots, Q^{abs}(x_N)]$ for each affective time series (Algorithm 5.1 illustrates the process described until now).

Algorithm 5.1 time series processing for videos of category: **A - Conditioned**

Input: $\mathbf{v} \leftarrow [v_1, v_2, \ldots, v_N]$, $\mathbf{a} \leftarrow [a_1, a_2, \ldots, a_N]$
for each affective time series i **do**
  $\mathbf{t} \leftarrow i$-th time series
  $\mathbf{q}_i = []$
  $s \leftarrow 0$
  for $s <$ duration in seconds of $\mathbf{t}$ **do**
    if $s = 0$ **then**
      $\mathbf{q}_i.append(0)$
    else
      $\mathbf{q}_i.append(abs(\mathbf{t}[s] - \mathbf{t}[s-1])/2)$
    end if
    $s \leftarrow s + 1$
  end for
end for
Output: $\mathbf{q}_{\text{val}} = \mathbf{q}_1$, $\mathbf{q}_{\text{aro}} = \mathbf{q}_2$

Next, the resulting sequences were used to simulate the real-time behaviour of our model, generating a MIDI soundtrack that evolves coherently according to each video. Precisely, we designed Algorithm 5.2 as a function $\mathcal{C}$ that receives as input the predicted sequences of valence and arousal $\mathbf{v}, \mathbf{a}$, the sequences of absolute value of difference quotients $\mathbf{q}_{\text{val}}, \mathbf{q}_{\text{aro}}$, two parameters p (percentile threshold) and d_{th} (minimum duration threshold), and generates a sequence $\mathbf{m}$ of music tokens, which can be modeled as

$$\mathbf{m} = \mathcal{C}(\mathbf{v}, \mathbf{a}, \mathbf{q}_{\text{val}}, \mathbf{q}_{\text{aro}}, p, d_{\text{th}}), \tag{5.4}$$

with $p \in [0, 100] \subset \mathbb{N}$ and $d_{\text{th}} \in \mathbb{N}$. With this procedure, the generation process does not continuously change every second, since running a new generation with this frequency would have been both a waste of computational resources and a too short time window for allowing even a single musical phrase to be composed. Instead, we change emotional conditioning only in most abrupt changes along the time series, and only after a minimum duration of previous generation. For doing so, we first define the p-th percentile P of an ordered sequence $\mathbf{x}$ of N values as

$$P(\mathbf{x}, p) = x_{\left[\frac{p}{100} * N\right]}, \tag{5.5}$$

where $\frac{p}{100} * N$ is the index i of the smallest element in $\mathbf{x}$ such that no more than p percent

of data is strictly less than x_i and at least p percent of data is less or equal to x_i, with $i \in 1, \ldots, N$. Consequently, we compute p-th percentile of both $\mathbf{q}_{\text{val}}, \mathbf{q}_{\text{aro}}$, obtaining two values $v_{\text{th}}, a_{\text{th}}$ that are compared each second with the current Q^{abs} of the correspondent emotion: if previous generation is already longer than d_{th} and one of the current quotients is higher then its threshold, then the generation starts again with new values.

Algorithm 5.2 music generation algorithm for videos of category: **A - Conditioned**

Input: $\mathbf{v}$, $\mathbf{a}$, $\mathbf{q}_{\text{val}}$, $\mathbf{q}_{\text{aro}}$, p, d_{th}
$v_{\text{th}} \leftarrow$ compute p-th percentile of time series $\mathbf{q}_{\text{val}}$
$a_{\text{th}} \leftarrow$ compute p-th percentile of time series $\mathbf{q}_{\text{aro}}$
$d_{\text{curr}} \leftarrow 0$
$\mathbf{m} \leftarrow []$
begin new music generation conditioned on current $\mathbf{v}[s], \mathbf{a}[s]$ values
$s \leftarrow 0$
for $s <$ duration in seconds of $\mathbf{v}$ **do**
    if $d_{\text{curr}} \geq d_{\text{th}}$ and ($\mathbf{q}_{\text{val}}[s] > v_{\text{th}}$ or $\mathbf{q}_{\text{aro}}[s] > a_{\text{th}}$) **then**
        $\mathbf{m}$.*append*(current music generation)
        begin new music generation conditioned on current $\mathbf{v}[s], \mathbf{a}[s]$ values
        $d_{\text{curr}} = 0$
    else
        continue previous music generation
        $d_{\text{curr}} = d_{\text{curr}} + 1$
    end if
    $s \leftarrow s + 1$
end for
Output: $\mathbf{m}$ containing all music generations concatenated

In our case, we chose $p = 80$, $d_{\text{th}} = 3$, meaning that we generate music changing emotional conditioning only in top-20% biggest emotional variations and only when the previous conditioning has already produced at least 3 seconds of music.

To generate the music corresponding to category **B - Unconditioned**, we used the vanilla music transformer from Sulun et al. (presented in section 4.2.2), that generates symbolic music without V-A conditioning. Furthermore, we designed the process described in Algorithm 5.3 in order to generate and concatenate pieces of music with a fixed number T_{max} of music tokens, that are then concatenated as a unique piece of length equal to L,

expressed in seconds. Consequently, we define a function $\mathcal{U}$ as

$$\mathbf{m} = \mathcal{U}(T_{\max}, L), \tag{5.6}$$

which returns a sequence of music tokens $\mathbf{m}$ like in Equation 5.4, but without any emotional conditioning.

Algorithm 5.3 music generation algorithm for videos of category: **B - Unconditioned**

Input: $T_{\max}$, L
$s \leftarrow 0$, $l_{curr} \leftarrow 0$
$\mathbf{m} \leftarrow []$
begin new unconditioned music generation, l_{curr} is constantly updated
while $s < L$ **do**
  if $l_{curr} > T_{\max}$ **then**
    $\mathbf{m}$.*append*(current music generation)
    $s \leftarrow$ current length of $\mathbf{m}$ in seconds
    $l_{curr} \leftarrow 0$
    begin new unconditioned music generation
  end if
end while
Output: $\mathbf{m}$ containing all music generations concatenated

For the experiment we set $L = 30$ seconds, corresponding to each video length, and $T_{\max} = 512$ music tokens.

Finally, videos belonging to category **C - Original** and **D - None** were recorded respectively with and without original soundtrack, and were both left untouched.

Symbolic music to audio implementation

Since the music transformer generates symbolic music, we need to synthesize it to audio, so that it is comparable with the game's original music. For both categories **A** and **B**, the generated MIDI files were loaded into a Digital Audio Workstation (REAPER), which was set with a VST instrument for each of the 5 possible tracks used by our generative model: ['drums', 'piano', 'strings', 'bass', 'guitar'], namely *MT Power Drum Kit*(MANDA AUDIO), *Midi Grand* (AIR Music Technology), *Velvet* (AIR Music Technology), *4Front Bass Module* (4Front) and *Ample Guitar M II Lite* (Ample Sound). All compositions were rendered with the same settings and effects.

Subjective evaluation Procedure

We designed a blind and randomized procedure for analyzing the effectiveness of our generative music system, combining an affective annotation task and a questionnaire. Both these tasks were performed using one video for each category described in subsection 5.2.1, for a total of 4 used for each participant, selected from the complete corpus of 40 clips. For each session, the choice of videos, their presenting order and the order of affective dimensions for the annotation task were randomized and determined prior to defining participant names and schedule. As explained in Moher et al. [66], randomization eliminates selection bias, preventing conscious or unconscious selection of specific types of participants to receive a particular experimental configuration. Additionally, it allows the collected results to be statistically relevant: as stated by Greenland [43] "inferential statistics, such as p-values, confidence intervals, and likelihood ratios, have very limited meaning in causal analysis when the mechanism of exposure assignment is largely unknown or is known to be nonrandom".

All experimental sessions involved a single participant and were conducted by an experimental supervisor in a quiet room, with a setup consisting of a computer, a game controller and a pair of headphones.

A single session articulates in 3 phases (depicted in Figure 5.5):

1. *Gameplay session.* The participant, after being presented with the overall structure of the experimental session, is given 15 minutes to play No Man's Sky, familiarizing with its commands and game flow. A fixed list of instructions is given to each subject, making sure that they experience most relevant gameplay components (e.g. exploring a planet's surface, flying on a space ship, travelling to a new planet). During the playthrough, only in-game sound effects are active, while music is muted in order to prevent the player being biased towards it in the subsequent evaluations;

2. *Emotion annotation task.* The participant is presented a brief explanation of a single affective dimension (either Valence or Arousal, depending on the randomized order) and is then asked to watch and annotate in real-time all 4 videos currently assigned, depending on how he feels according to the described emotion. Once the last video has been annotated, the process repeats for the other affective dimension. Valence was presented as the emotion that "Indicates how happy or sad (unhappy) you feel", while Arousal "Indicates how excited or calm you are feeling". The Self-Assessment-Manikin [23] and more extensive explanations were used to better clarify these concepts.

(a) (b)

(c)

Figure 5.5: Pictures taken during different iterations of the evaluation test performed at the Politecnico di Milano, Milan, Italy. Each picture corresponds to one of the three phases of the experimental procedure: (a) gameplay session, (b) emotion annotation and (c) questionnaire.

For performing the annotation task, we implemented a program [4] that replicates the functionality of the web interface PAGAN [63], in particular following the approach named RankTrace [59]. While a video is being played, the user is asked to press the keyboard arrows up or down every time he feels an increase or decrease of the current emotion perceived. The button inputs are received every 200ms and a live plot gives the user a visual feedback of the annotation process so far. We opted for 30 seconds video-duration in order to keep the annotation task relatively short, preventing fatigue that would have caused poor annotations;

[4] Code available in the Github repositories referenced at page 67

3. *Questionnaire.* Lastly, the participant opens a questionnaire where the same 4 videos annotated during the previous task are presented and ready to be reproduced again, if needed. For each of the following questions, the subject is asked to select the video that best answers them:

   (a) In which video do you feel the music most closely matches the events and actions of the gameplay? (Gameplay match)

   (b) In which video do you feel that the music most closely matches the emotion that you perceive from the gameplay? (Emotion match)

   (c) In which video did you feel most immersed in the gameplay? (Immersion)

   (d) Which video's music did you enjoy the most? (Preference)

   Each question will be referenced in future discussions with the words between parenthesis, i.e. Gameplay match, Emotion match, Immersion, Preference.
   After that, the participant is asked a few questions about anagraphic (age, pronouns) and gaming experience (hours-per-week usually spent playing, previous experience with No Man's Sky). Finally, before submitting the form some free additional comments can be added.

5.2.2. Results

34 subjects ranging from 18 to 29 years old, average age = 23.8, participated to the subjective evaluation. Among these, 20 use he/him pronouns, 13 identify as she/her and one chose they/them. 67.6% of the participants spend less than four hours per week playing video games, 17.6% play between 4 and 8 hours, while the remaining 14.8% spend more than 8 hours weekly. Almost all subjects, 32 out of 34, reported having not played the game before. Most of the participants were recruited through advertising on university groups of Politecnico di Milano using different platforms (Telegram, Slack) and on average each experimental session lasted 30 minutes.

With 34 participants and 8 annotation curves per-participant, we collected 272 time series, which were compared to our model's predictions according to three metrics: Distance, performed with Dynamic Time Warping (DTW), Root Mean Squared Error (RMSE), and Pearson Correlation Coefficient (PCC). The DTW distance allows for a flexible alignment of the two sequences, even when they have different lengths or temporal distortions. Before performing the calculations, we discarded 2 annotations where the user did not press any key. Then, all values of both annotations and predictions were normalized with z-score

normalization

$$Z(x_i) = \frac{x_i - \mu}{\sigma}, \tag{5.7}$$

where x_i represents a single time point of a time series $\mathbf{x}$, while mean μ and standard deviation σ are computed globally across each of the two groups. Then, for each user annotation we calculate its metrics by comparing it with the model's predictions for the same video. The obtained results are presented in Tables 5.4 and 5.5, where the average value is computed across each of the four video categories, respectively for Valence and Arousal. Then, Table 5.6 shows the average metrics merging both affective dimensions, while Table 5.7 combines all video categories and compares the two emotions.

Each metric is associated with its Standard Error of Mean (SEM), a statistical measure that quantifies its variability as an estimate of the population mean. Formally it is defined as

$$\text{SEM} = \frac{\sigma}{\sqrt{S}}, \tag{5.8}$$

where σ refers to the standard deviation and S indicates the number of samples collected for the measurement.

Affective Dimension: Valence

Measure	Result	Conditioned	Unconditioned	Original	None
DTW	Distance	15.164	16.741	22.006	23.529
	SEM	1.221	2.074	1.674	2.313
RMSE	RMSE	1.006	1.113	1.37	1.460
	SEM	0.070	0.129	0.096	0.130
PCC	PCC	0.300	0.323	0.379	0.324
	SEM	0.032	0.034	0.042	0.042

Table 5.4: Results by musical category, using only Valence annotations.

Affective Dimension: Arousal

Measure	Result	Conditioned	Unconditioned	Original	None
DTW	Distance	14.985	16.737	18.129	15.662
	SEM	1.288	1.337	1.917	1.509
RMSE	RMSE	1.061	1.127	1.248	1.084
	SEM	0.085	0.081	0.13	0.088
PCC	PCC	0.275	0.329	0.317	0.369
	SEM	0.034	0.046	0.036	0.036

Table 5.5: Results by musical category, using only Arousal annotations.

All Affective Dimensions

Measure	Result	Conditioned	Unconditioned	Original	None
DTW	Distance	15.074	16.739	20.067	19.476
	SEM	0.881	1.225	1.285	1.439
RMSE	RMSE	1.033	1.120	1.309	1.266
	SEM	0.055	0.075	0.081	0.080
PCC	PCC	0.287	0.326	0.348	0.347
	SEM	0.023	0.028	0.028	0.027

Table 5.6: Results by musical category, merging both dimensions.

All music Categories

Measure	Result	Valence	Arousal
DTW	Distance	19.298	16.378
	SEM	0.964	0.765
RMSE	RMSE	1.234	1.13
	SEM	0.056	0.049
PCC	PCC	0.332	0.322
	SEM	0.019	0.019

Table 5.7: Results by affective dimension, merging all musical categories.

The questionnaire results are displayed in Figure 5.6, containing the answer distribution for all questions described in subsection 5.2.1.

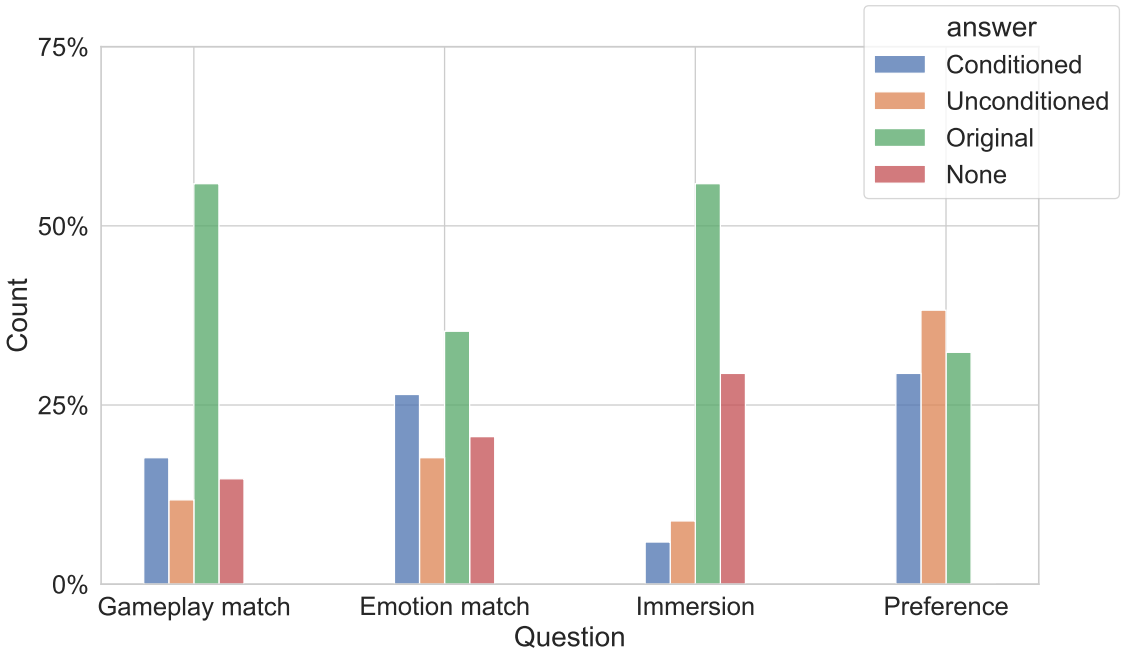


Figure 5.6: Results of the questionnaire part of the subjective experiment

Statistical Analysis

Several statistical procedures, such as correlation, regression, t-tests, and analysis of variance, which are known as parametric tests, rely on the assumption that the data conforms to a normal distribution or Gaussian distribution [38]. For this reason, we test the assumption of normality using D'Agostino-Pearson omnibus test, grouping metrics by category as depicted in Tables 5.8, 5.9, then merging affective dimensions as in Table 5.10, and finally merging all categories while keeping affective dimensions separated (Table 5.11). In each table's description we also indicate the sample size S of each group tested.

Normality test for Valence dimension

Measure	Conditioned	Unconditioned	Original	None
Distance	$p < 0.05$	$p < 0.05$	$p < 0.05$	$p < 0.05$
RMSE	$p = 0.22$	$p < 0.05$	$p < 0.05$	$p < 0.05$
PCC	$p = 0.17$	$p = 0.07$	$p < 0.05$	$p = 0.31$

Table 5.8: Normality test performed for Valence dimension on Distance, RMSE, PCC, for each music category. Number of samples $S >= 32$.

Normality test for Arousal dimension

Measure	Conditioned	Unconditioned	Original	None
Distance	$p = 0.06$	$p = 0.27$	$p < 0.05$	$p < 0.05$
RMSE	$p < 0.05$	$p = 0.39$	$p < 0.05$	$p < 0.05$
PCC	$p = 0.18$	$p < 0.05$	$p = 0.06$	$p = 0.55$

Table 5.9: Normality test performed for Arousal dimension on Distance, RMSE, PCC, for each music category. Number of samples $S >= 33$.

Normality test for All Dimensions

Measure	Conditioned	Unconditioned	Original	None
Distance	$p < 0.05$	$p < 0.05$	$p < 0.05$	$p < 0.05$
RMSE	$p < 0.05$	$p < 0.05$	$p < 0.05$	$p < 0.05$
PCC	$p < 0.05$	$p < 0.05$	$p < 0.05$	$p = 0.20$

Table 5.10: Normality test performed merging all dimensions on Distance, RMSE, PCC, for each music category. Number of samples $S >= 66$.

Normality test for All Categories

Measure	Valence	Arousal
Distance	$p < 0.05$	$p < 0.05$
RMSE	$p < 0.05$	$p < 0.05$

Table 5.11: Normality test performed merging all categories on Distance, RMSE, PCC, for each affective dimension. Number of samples $S >= 134$.

Next, in Table 5.12 we summarize the *ANOVA* test, which compares the value distribution of different groups, that in our case correspond to the four video categories. For each metric employed and for each grouping considered in previous tables, with this test we assess whether there are statistically significant differences between at least two groups among the ones considered.

One-way ANOVA tests

Measure	Valence	Arousal	All Dimensions	All Categories
Distance	$p < 0.05$	$p = 0.50$	$p < 0.05$	$p < 0.05$
RMSE	$p < 0.05$	$p = 0.54$	$p < 0.05$	$p = 0.16$
PCC	$p = 0.49$	$p = 0.38$	$p = 0.33$	$p = 0.74$

Table 5.12: Analysis of variance computed for each metric (rows) and for each grouping (columns), respectively: Valence (Table 5.4), Arousal (Table 5.5), All Dimensions (Table 5.6), All Categories (Table 5.7).

Only for results where ANOVA is significant (p-value < 0.05) and all groups of measures have a normal distribution, we additionally perform a *post hoc Tukey* test, which returns a pairwise comparison for each possible couple among the groups considered, determining if their difference is statistically significant. Specifically, each of the next tables consists of the following columns:

- *Group1* and *Group2*: These columns indicate the two groups or conditions being compared in the post hoc test.
- *Mean Diff* (Mean Difference): This column shows the difference in means (average values) between Group1 and Group2.
- p-adj (Adjusted p-value): This column provides the p-value that has been adjusted for multiple comparisons. If p-adj is less than a chosen significance level (i.e. 0.05) it suggests that there is a statistically significant difference between the groups, hence the null-hypothesis can be rejected.
- *Lower* and *Upper*: These columns provide the lower and upper bounds of a confidence interval for the mean difference. Confidence intervals indicate the range of values within which the true mean difference is likely to fall with a certain level of confidence.
- *Reject*: This column indicates whether the null hypothesis (the hypothesis that there is no difference between the groups) is rejected or not, based on the adjusted p-value.

Post hoc test, Valence - Distance

Category 1	Category 2	Mean Diff	p-Adj	Lower	Upper	Reject
A	B	1.5772	0.9299	-5.1922	8.3465	False
A	C	6.8417	0.0466	0.0723	13.6111	True
A	D	8.3646	0.0102	1.4903	15.2389	True
B	C	5.2646	0.1845	-1.5048	12.0339	False
B	D	6.7875	0.0543	-0.0869	13.6618	False
C	D	1.5229	0.9389	-5.3514	8.3972	False

Table 5.13: Post hoc analysis of Table 5.4 (Valence) for Distance metric, computed since ANOVA in Table 5.12 rejected null-hypothesis.

Post hoc test, all Dimensions - Distance

Category 1	Category 2	Mean Diff	p-Adj	Lower	Upper	Reject
A	B	1.6649	0.7679	-2.7828	6.1127	False
A	C	4.9928	0.0208	0.5450	9.4406	True
A	D	4.4018	0.0563	-0.0796	8.8831	False
B	C	3.3279	0.2162	-1.1199	7.7757	False
B	D	2.7368	0.3924	-1.7445	7.2182	False
C	D	-0.5911	0.9863	-5.0724	3.8903	False

Table 5.14: Post hoc analysis of Table 5.6 (all Dimensions) for Distance metric, computed since ANOVA in Table 5.12 rejected null-hypothesis.

Post hoc test, all Dimensions - RMSE

Category 1	Category 2	Mean Diff	p-Adj	Lower	Upper	Reject
A	B	0.0871	0.8346	-0.1805	0.3547	False
A	C	0.2756	0.0408	0.0080	0.5432	True
A	D	0.2331	0.1166	-0.0366	0.5027	False
B	C	0.1885	0.2658	-0.0791	0.4561	False
B	D	0.1460	0.5006	-0.1236	0.4156	False
C	D	-0.0425	0.9771	-0.3121	0.2271	False

Table 5.15: Post hoc analysis of Table 5.6 (all Dimensions) for RMSE metric, computed since ANOVA in Table 5.12 rejected null-hypothesis.

Post hoc test, all Categories - Distance

Dimension 1	Dimension 2	Mean Diff	p-Adj	Lower	Upper	Reject
Arousal	Valence	2.9197	0.0182	0.5006	5.3388	True

Table 5.16: Post hoc analysis of Table 5.7 (all Categories) for Distance metric, computed since ANOVA in Table 5.12 rejected null-hypothesis.

Considering Valence, we observe that the Distance metrics are normally distributed on all 4 categories. After verifying that the ANOVA test finds significant results, we perform post-hoc tukey test (Table 5.13) concluding that the average Distance computed for Conditioned music is significantly lower compared to category C (Original) and D (None), while for other comparisons the null hypothesis cannot be rejected.

Comparing the 4 video categories while merging Valence and Arousal, both Distance and RMSE metrics result normally distributed for each group. In this case, after performing ANOVA and post-hoc tukey tests we can conclude that videos with Conditioned music lead to significantly better results than the original ones (Tables 5.14 and 5.15, comparison A vs C) for both measures.

Finally, we compare Valence and Arousal overall performance across all annotations, after verifying that both groups are normally distributed. As a result, Arousal significantly outperforms Valence for Distance metric (Table 5.16). Overall, we did not find statistically relevant results by comparing PCC measures for any of the modalities discussed above (Table 5.12, last row), consequently we will not mention this metric in the next discussions.

5.2.3. Discussion

Emotion annotation tasks are always a delicate procedure, which can be penalized by a large amount of factors that can be more or less easy to prevent. Consequently, the larger the sample size is, the more reliable the results are. Even though some articles suggest that when the sampling size is greater than 30 its distribution tends to approximate a normal distribution [38], others state that there is limited or insufficient documented evidence exists to substantiate that this specific value is the definitive threshold for non-normal distributions [51]. For this reason, we only considered statistical measures after assessing the normality of our data, thus preventing weak or inconsistent conclusions.

Looking at results obtained from the analysis of the annotation task, we surprisingly observe that overall Valence annotations (Table 5.4) are predicted with higher error compared to Arousal (Table 5.5), as concluded by the post hoc analysis in Table 5.16. This

result is in contrast with our model's performance on continuous movie annotations (Table 5.3), suggesting a significant difference between the two media. As a matter of fact, many games like the one used in our evaluation present colorful and cartoony graphics, which strongly differ from film images. This result motivates the need to collect also valence annotations for future releases of affective video game datasets, since for now there isn't this availability as discussed in section 3.2.4.

Additionally, videos without music have generally the lowest prediction performance among all categories. This result, and the high SEM associated to the same category, suggests that, especially for Valence, participant's perception of emotions without music is highly subjective and changes substantially across different subjects. While we can observe that there is still room for improvement for the emotion prediction task, on the other hand we can see this result as a hint that our model's generative music effectively reinforces the desired emotion value, which without music remains too ambiguous. In fact, this difference is statistically proven for Valence category, where predictions significantly improve when emotionally conditioned music is added (Tables 5.13, category A vs D). Lastly, comparing Conditioned and Unconditioned annotations leads to inconclusive results, although for both emotions we see lower errors and SEM for the first category (Tables 5.4, 5.5). In order to effectively obtain relevant results and to better investigate other aspects described earlier we would need a higher number of participants.

Moving our focus to the questionnaire, it can be noticed that the original score outperforms other categories in the first three questions. This result is expected (and also observed in [75]) due to several elements. Firstly, the original music had a professional musical production, resulting in a huge advantage compared to our generative approach, since we rendered all tracks using free VST instruments, with fixed settings and no mixing specific for every video. This approach was necessary in order to not introduce any bias in the evaluation, although in a real-time application these issues could be easily overcome by using one of the multiple audio plugins currently available for games (to name a few, FMOD and Elias 3). Secondly, while the composers hired for No Man's Sky created a corpus of music with a style specifically chosen to be coherent and effective with the game, we used a music transformer trained on a dataset of pop music, which consisted of the best available choice, though it might not suit properly a gaming scenario. Overall, we do not see this as a negative result, but instead as an indication that the proposed music-generation procedure would benefit from a larger availability of affective music datasets.

Focusing on the "Gameplay match" and "Emotion match" questions, we observe that conditioned music outperforms the unconditioned one. This is an interesting result, since it demonstrates that participants perceived the emotional coherency that we tried to

obtain, distinguishing it from the "unconditioned" alternative most of the times.

The question regarding "Immersion" shows the lowest number of choices for generative music (category A) and the most votes for original soundtrack (category C): this can be explained by the fact that our music transformer generates general purpose "pop" style music, while the original composers had a more "ambient" approach, the latter being better suited for the game's atmosphere. Also, it can be observed that this question obtained more than 25% of preferences for category D (no music), suggesting that immersion may be penalized by prominent or engaging music (like pop music), whereas a more subtle approach could be preferred (i.e. the ambient music of the original soundtrack). On top of that, a participant wrote at the end of the questionnaire "I'm a huge fan of non-invasive sounds when it comes to exploration games": unfortunately, to the best of our knowledge no affective dataset of ambient music exists in the literature.

Analyzing the last column, related to musical preference, we can make a first observation by summing the votes for category A and B (conditioned and unconditioned), since their only difference lies in the coherency with the video, which was not mentioned in this question. As a result, we obtain that the music transformer architecture clearly outperforms the original score in terms of musical preference. Then, keeping the first two categories apart we see a similar performance of all three categories, with a slight predominance for category B compared to A, which could be explained by the difference between Algorithms 5.2 and 5.3: Conditioned music can start a new generation after only 3 seconds from the previous one and in general more frequently than in Unconditioned videos, thus creating more transitions between different pieces. As demonstrated by Cheung et al. [27], when inside a musical piece both uncertainty (how unclear are expectations when anticipating an event) and surprise (how much what is heard deviates from expectations) are high, musical pleasantness is low. Nonetheless, in a real-time application, with sessions lasting minutes or hours, the minimum interval between musical changes could be increased without preventing the model to effectively react to most emotional changes, hence improving overall musical pleasantness with less abrupt changes.

5.3. Conclusive Remarks

In this chapter, we have presented and thoroughly examined our experimental setup and the results collected in order to validate the efficacy of our proposed approach. Initially, section 5.1 provided all the implementation details of our architecture designed for the valence-arousal prediction task. We additionally presented and discussed its performance with different datasets in detail.

Moving forward, section 5.2 delved into the specifics of the experimental setup defined for our complete framework. Here, we formalized and explained each detail regarding the continuous generation of both conditioned and unconditioned music. Then, the subjective evaluation procedure was described, providing all the results collected as well as their statistical analysis. Our experiments have yielded valuable insights, and in the next chapter we will draw further conclusions from our observations.

6 | Conclusions and Future Developments

In this thesis, we have introduced a novel framework for generative video game music, particularly well-suited for open-world games, where the player's freedom and the consequent high number of gameplay scenarios become unsuitable for a human composer. Moreover, our approach towards emotions holds potential for games aiming to convey artistic and emotional experiences.

In this concluding chapter, we will first outline the key steps in our contribution and report the main results obtained (section 6.1). Next, we will address the limitations identified during our evaluations and propose directions for future research (section 6.2). Finally, section 6.3 will summarize the conclusions drawn from this thesis.

6.1. Main Contributions and Results

Our proposed framework consists of a first model that constantly extracts affective features from gameplay frames and another architecture that generates music coherently with those predictions. To the best of our knowledge, this is the first work that combines these two components and assesses the applicability of this approach in the video game field.

For the emotion detection model, we initially designed a 3D convolutional neural network, which receives a sequence of video frames and returns two continuous values, namely Valence and Arousal, that combined can represent a wide range emotions, as described by the circumplex model of affect [77]. We trained our network on the LIRIS-ACCEDE dataset, consisting of movie clips globally annotated with both affective dimensions, and we tested its performance in predicting the evolution of those values over time. Afterwards, for the procedural music generation task we employed a pre-trained music transformer proposed by Sulun et al. [92], that generates novel symbolic music conditioned on V-A input values. We additionally modified its inference in order to compose a musical continuation starting from an input piece, improving our control over musical style. As a result, we combined

the two components described obtaining the overall architecture desired.
Afterwards, guided by a preliminary assessment on the AGAIN dataset, we chose No Man's Sky as the game to use for testing our final architecture, designing and implementing a subjective evaluation procedure based on the works of Plut et al. [75, 76]. In practice, we asked each participant to perform an emotion annotation task, consisting of watching 4 gameplay videos recorded with our chosen game, each one accompanied by a different music category: "A - Conditioned", "B - Unconditioned", "C - Original", "D - None". Subsequently, the subjects were asked to fill a questionnaire comparing the same 4 videos they were assigned to, according to the following characteristics determined by music: "Gameplay match", "Emotion match", "Immersion", "Preference".

Regarding the emotion annotation results, after performing a statistical analysis (i.e. normality test, ANOVA, post hoc test) we observed that our model (category A) effectively elicits the desired emotions, significantly improving similarity between annotations and predictions with respect to videos of categories C and D. Ulterior comparisons, like Conditioned vs Unconditioned music, did not yield statistically relevant results, even though our model's predictions are always more accurate in videos with conditioned music, with respect to other categories. On top of that, results show that in absence of a musical stimuli emotions are highly subjective, since annotations on videos without music have the highest variance among them. This suggests that different users can interpret the same visual stimuli in varying emotional ways, but adding background music reduces this ambiguity.

The questionnaire results provided a more qualitative evaluation of our method. Comparing Conditioned and Unconditioned music, the majority of voters found music of the first category more coherent to both gameplay events and their perceived emotions ("Gameplay match" and "Emotion match"), confirming the effectiveness of our framework. Then, we observed that for the first three questions Original in-game music outperforms the generative approach, while for the question regarding musical "Preference" the situation is the opposite. These observations suggest that while the music transformer produces music of high quality and also emotionally coherent, its application in the gaming context is still challenging and requires further improvements.

6.2. Future Developments

Overall, in this thesis we addressed multiple issues and opportunities regarding our proposed approach. Below, we highlight the most relevant points and suggest directions for future research:

- Reviewing affective datasets of gameplay videos, only in 2022 a sufficiently large dataset named AGAIN [64] was proposed, but currently it provides only Arousal annotations, significantly limiting its usefulness for approaches like ours. A similar situation is found for symbolic music: apart from the Lakh Pianoroll Dataset [32], containing a large amount of pop music with V-A labels, there are not many equally valuable alternatives covering different musical genres. The availability of more affective datasets in both domains (video and music) with a complete emotion labeling would certainly allow further improvements in the proposed architecture.
- Focusing on the versatility of our proposed framework, the choice of not constraining the emotion-prediction procedure to specific input sources (such as in-game parameters, as discussed in sections 3.1 and 3.2) makes it easily adaptable to a wide range of video games. This flexibility may also offer creative and artistic freedom to developers and composers. Furthermore, beyond the gaming context we envision potential applications of our framework in other fields, such as assisted music composition or media art installations. Therefore, future researchers could focus on validating all the mentioned scenarios.
- Lastly, our proposed method has not yet been implemented as a real-time system, since deep neural networks rely on GPUs also during inference. This creates a conflict with the video game medium, since a game's performance strongly relies on the same device. However, in future works our proposed approach could be implemented with lower level programming languages compared to python, obtaining significant performance improvements. Furthermore, the recent diffusion of cloud gaming (which consists of rendering a video game remotely in the cloud and streaming its scenes as an audiovisual sequence back to the player [87]) provides new prospects for computationally demanding applications, including our work.

6.3. Conclusive Remarks

To sum up, although not definitive, our results have demonstrated that emotion-conditioned procedural music generation for video games is a viable path to pursue. We can conclude that a real-time implementation of this procedure, along with improvements in the areas mentioned earlier, has the potential to enhance and innovate within the context of open-world games and artistic experiences, which are increasingly gaining popularity in the entertainment community.

Bibliography

[1] Chord charts and maps. https://www.mugglinworks.com/chordmaps/chartmaps.htm. Accessed: 2023-06-12.

[2] Space Invaders. Taito, 1978.

[3] Halo: Combat Evolved. Bungie, 2001.

[4] Halo Reach. Bungie, 2010.

[5] StarCraft II: Wings Of Liberty. Blizzard Entertainment, 2010.

[6] The Legend Of Zelda: Mystery Of Solarus. Solarus Team, 2011.

[7] Crypt Of The NecroDancer. Brace Yourself Games, 2012.

[8] Journey. Thatgamecompany, 2012.

[9] Assassin's Creed: Black Flag. Ubisoft Montreal, 2013.

[10] Undertale. Toby Fox, 2015.

[11] Abzû. 505 Games, 2016.

[12] No Man's Sky. Hello Games, 2016.

[13] Hollow Knight. Team Cherry, 2017.

[14] Celeste. Extremely OK Games, 2018.

[15] Red Dead Redemption 2. Rockstar Games, 2018.

[16] Hades. Supergiant Games, 2020.

[17] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas,

O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[18] G. Amaral, A. Baffa, J.-P. Briot, B. Feijó, and A. Furtado. An adaptive music generation architecture for games based on the deep learning transformer model. In *2022 21st Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 1–6. IEEE, 2022.

[19] T. Anders. A model of musical motifs. In *International Conference on Mathematics and Computation in Music*, pages 52–58. Springer, 2007.

[20] T. H. Apperley. Genre and game studies: Toward a critical approach to video game genres. *Simulation & gaming*, 37(1):6–23, 2006.

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

[22] T. Bhosale, S. Kulkarni, and S. N. Patankar. 2d platformer game in unity engine. *International Research Journal of Engineering and Technology*, 5(04):3021–3024, 2018.

[23] M. M. Bradley and P. J. Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.

[24] J.-P. Briot and F. Pachet. Deep learning for music generation: challenges and directions. *Neural Computing and Applications*, 32(4):981–993, 2020.

[25] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[26] N. Charness and W. R. Boot. Technology, gaming, and social networking. In *Handbook of the Psychology of Aging*, pages 389–407. Elsevier, 2016.

[27] V. K. Cheung, P. M. Harrison, L. Meyer, M. T. Pearce, J.-D. Haynes, and S. Koelsch. Uncertainty and surprise jointly predict musical pleasure and amygdala, hippocampus, and auditory cortex activity. *Current Biology*, 29(23):4084–4092, 2019.

[28] R. I. Clarke, J. H. Lee, and N. Clark. Why video game genres fail: A classificatory analysis. *Games and Culture*, 12(5):445–465, 2017.

[29] P. M. Cole, S. E. Martin, and T. A. Dennis. Emotion regulation as a scientific construct: Methodological challenges and directions for child development research. *Child development*, 75(2):317–333, 2004.

[30] K. Collins. *Game sound: an introduction to the history, theory, and practice of video game music and sound design*. Mit Press, 2008.

[31] E. Dellandréa, L. Chen, Y. Baveye, M. V. Sjöberg, and C. Chamaret. The mediaeval 2016 emotional impact of movies task. In *CEUR Workshop Proceedings*, 2016.

[32] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[33] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[34] F. Eyben. *Real-time speech and music classification by large audio feature space extraction*. Springer, 2015.

[35] L. N. Ferreira and J. Whitehead. Learning to generate music with sentiment. 2019.

[36] L. N. Ferreira, L. Mou, J. Whitehead, and L. H. Lelis. Controlling perceived emotion in symbolic music generation with monte carlo tree search. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 163–170, 2022.

[37] E. Geslin, L. Jégou, and D. Beaudoin. How color properties can be used to elicit emotions in video games. *International Journal of Computer Games Technology*, 2016:1–1, 2016.

[38] A. Ghasemi and S. Zahediasl. Normality tests for statistical analysis: a guide for non-statisticians. *International journal of endocrinology and metabolism*, 10(2):486, 2012.

[39] R. Gilbert. Monkey Island 2: LeChuck's Revenge. LucasArts, 1991.

[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[41] R. Gozalo-Brizuela and E. C. Garrido-Merchán. Chatgpt is not all you need. a state of the art review of large generative ai models. *GRACE: Global Review of AI Community Ethics*, 1(1), 2023.

[42] M. Granato, D. Gadia, D. Maggiorini, and L. A. Ripamonti. Feature extraction and selection for real-time emotion recognition in video games players. In *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 717–724. IEEE, 2018.

[43] S. Greenland. Randomization, statistics, and causal inference. *Epidemiology*, 1(6): 421–429, 1990.

[44] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[45] C. Hernandez-Olivan and J. R. Beltran. Music composition with deep learning: A review. *Advances in speech and music technology: computational aspects and applications*, pages 25–50, 2022.

[46] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[47] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[48] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2018.

[49] H.-T. Hung, J. Ching, S. Doh, N. Kim, J. Nam, and Y.-H. Yang. EMOPIA: A multi-modal pop piano dataset for emotion recognition and emotion-based music generation. In *Proc. Int. Society for Music Information Retrieval Conf.*, 2021.

[50] P. E. Hutchings and J. McCormack. Adaptive music composition for games. *IEEE Transactions on Games*, 12(3):270–280, 2019.

[51] M. R. Islam. Sample size and its role in central limit theorem (clt). *Computational and Applied Mathematics Journal*, 4(1):1–7, 2018.

[52] E. Joosten, G. v. Lankveld, and P. Spronck. Colors and emotions in video games. In *11th International Conference on Intelligent Games and Simulation GAME-ON*, pages 61–65. sn, 2010.

[53] D. Keltner and J. J. Gross. Functional accounts of emotions. *Cognition & Emotion*, 13(5):467–480, 1999.

[54] C. Klimmt, D. Possler, N. May, H. Auge, L. Wanjek, and A.-L. Wolf. Effects of soundtrack music on the video game experience. *Media Psychology*, 22(5):689–713, 2019.

[55] A. Krenker, J. Bešter, and A. Kos. Introduction to the artificial neural networks. *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pages 1–18, 2011.

[56] J. H. Lee, N. Karlova, R. I. Clarke, K. Thornton, and A. Perti. Facet analysis of video game genres. *IConference 2014 Proceedings*, 2014.

[57] T. Li, Y. Baveye, C. Chamaret, E. Dellandréa, and L. Chen. Continuous arousal self-assessments validation using real-time physiological responses. In *Proceedings of the 1st International Workshop on Affect & Sentiment in Multimedia*, pages 39–44, 2015.

[58] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. In *International Conference on Learning Representations*, 2016.

[59] P. Lopes, G. N. Yannakakis, and A. Liapis. Ranktrace: Relative and unbounded affect annotation. In *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 158–163. IEEE, 2017.

[60] K. Makantasis, A. Liapis, and G. N. Yannakakis. From pixels to affect: A study on games and player experience. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–7. IEEE, 2019.

[61] K. Makantasis, A. Liapis, and G. N. Yannakakis. The pixels and sounds of emotion: General-purpose representations of arousal in games. *IEEE Transactions on Affective Computing*, 2021.

[62] Marty Stratton, Hugo Martin. DOOM. id Software, 2016.

[63] D. Melhart, A. Liapis, and G. N. Yannakakis. Pagan: Video affect annotation made

easy. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 130–136. IEEE, 2019.

[64] D. Melhart, A. Liapis, and G. N. Yannakakis. The arousal video game annotation (again) dataset. *IEEE Transactions on Affective Computing*, 13(4):2171–2184, 2022.

[65] S. Miyamoto. Super Mario Bros. Nintendo Creative Department, 1985.

[66] D. Moher, S. Hopewell, K. F. Schulz, V. Montori, P. C. Gøtzsche, P. J. Devereaux, D. Elbourne, M. Egger, and D. G. Altman. Consort 2010 explanation and elaboration: updated guidelines for reporting parallel group randomised trials. *International journal of surgery*, 10(1):28–55, 2012.

[67] P. M. Niedenthal and F. Ric. *Psychology of emotion*. Psychology Press, 2017.

[68] Z. Niu, G. Zhong, and H. Yu. A review on the attention mechanism of deep learning. *Neurocomputing*, 452:48–62, 2021.

[69] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, 32: 955–967, 2020.

[70] K. O'Shea and R. Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[71] Y. Ou, Z. Chen, and F. Wu. Multimodal local-global attention network for affective video content analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(5):1901–1914, 2020.

[72] D. Plans and D. Morelli. Experience-driven procedural music generation for games. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(3):192–198, 2012.

[73] C. Plut and P. Pasquier. Music matters: An empirical study on the effects of adaptive music on experienced and perceived player affect. In *2019 ieee conference on games (cog)*, pages 1–8. IEEE, 2019.

[74] C. Plut and P. Pasquier. Generative music in video games: State of the art, challenges, and prospects. *Entertainment Computing*, 33:100337, 2020.

[75] C. Plut, P. Pasquier, J. Ens, and R. Tchemeube. Preglam-mmm: Application and evaluation of affective adaptive generative music in video games. In *Proceedings of the 17th International Conference on the Foundations of Digital Games*, pages 1–11, 2022.

[76] C. Plut, P. Pasquier, J. Ens, and R. Bougueng. Preglam: A predictive, gameplay-based layered affect model. *IEEE Transactions on Games*, 2023.

[77] J. Posner, J. A. Russell, and B. S. Peterson. The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, 17(3):715–734, 2005.

[78] C. Raffel. Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching. 2016, 2016.

[79] C. Raffel and D. P. Ellis. Optimizing dtw-based audio-to-midi alignment and matching. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 81–85. IEEE, 2016.

[80] R. A. Rensink. The dynamic representation of scenes. *Visual cognition*, 7(1-3): 17–42, 2000.

[81] K. Roose. An ai-generated picture won an art prize. artists aren't happy. *The New York Times*, 2(September), 2022.

[82] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[83] J. A. Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.

[84] J. A. Russell and L. F. Barrett. Core affect, prototypical emotional episodes, and other things called emotion: dissecting the elephant. *Journal of personality and social psychology*, 76(5):805, 1999.

[85] M. Scirea, J. Togelius, P. Eklund, and S. Risi. Metacompose: A compositional evolutionary music composer. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design: 5th International Conference, EvoMUSART 2016, Porto, Portugal, March 30–April 1, 2016, Proceedings 5*, pages 202–217. Springer, 2016.

[86] P. Shaw, J. Uszkoreit, and A. Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, 2018.

[87] R. Shea, J. Liu, E. C.-H. Ngai, and Y. Cui. Cloud gaming: architecture and performance. *IEEE network*, 27(4):16–21, 2013.

[88] Y.-J. Shih, S.-L. Wu, F. Zalkow, M. Muller, and Y.-H. Yang. Theme transformer:

Symbolic music generation with theme-conditioned transformer. *IEEE Transactions on Multimedia*, 2022.

[89] M. Sjöberg, Y. Baveye, H. Wang, V. L. Quang, B. Ionescu, E. Dellandréa, M. Schedl, C.-H. Demarty, and L. Chen. The mediaeval 2015 affective impact of movies task. In *MediaEval*, volume 1436, 2015.

[90] W. Strank. The legacy of imuse: Interactive video game music in the 1990s. In *Music and Game: Perspectives on a popular alliance*, pages 81–91. Springer, 2012.

[91] B. L. Sturm, O. Ben-Tal, Ú. Monaghan, N. Collins, D. Herremans, E. Chew, G. Hadjeres, E. Deruty, and F. Pachet. Machine learning research that matters for music creation: A case study. *Journal of New Music Research*, 48(1):36–55, 2019.

[92] S. Sulun, M. E. Davies, and P. Viana. Symbolic music generation conditioned on continuous-valued emotions. *IEEE Access*, 10:44617–44626, 2022.

[93] E. R. Tait and I. L. Nelson. Nonscalability and generating digital outer space natures in no man's sky. *Environment and Planning E: Nature and Space*, 5(2):694–718, 2022.

[94] J. Teuwen and N. Moriakov. Convolutional neural networks. In *Handbook of medical image computing and computer assisted intervention*, pages 481–501. Elsevier, 2020.

[95] H. T. P. Thao, B. Balamurali, D. Herremans, and G. Roig. Attendaffectnet: Self-attention based networks for predicting affective responses from movies. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8719–8726. IEEE, 2021.

[96] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018.

[97] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[98] J. Vidqvist. Open-world game design: case study: The legend of zelda: Breath of the wild. 2019.

[99] Q. Wang, X. Xiang, J. Zhao, and X. Deng. P2sl: Private-shared subspaces learning for affective video content analysis. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022.

[100] Wikipedia. Indie game — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Indie%20game&oldid=1170291406, 2023. [Online; accessed 18-August-2023].

[101] Wikipedia. List of best-selling video games — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=List%20of%20best-selling%20video%20games&oldid=1170908671, 2023. [Online; accessed 18-August-2023].

[102] K. Zheng, R. Meng, C. Zheng, X. Li, J. Sang, J. Cai, J. Wang, and X. Wang. Emotionbox: A music-element-driven emotional music generation system based on music psychology. *Frontiers in Psychology*, 13:5189, 2022.

[103] A. Zlatintsi, P. Koutras, G. Evangelopoulos, N. Malandrakis, N. Efthymiou, K. Pastra, A. Potamianos, and P. Maragos. 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, 2017.

[104] F. Zumerle, L. Comanducci, M. Zanoni, A. Bernardini, F. Antonacci, and A. Sarti. Procedural music generation for videogames conditioned through video emotion recognition. (accepted to the 4th International Symposium on the Internet of Sounds - IS^2 2023).

A | Appendix A

Appendix containing a detailed plot of the custom Resnet model presented in chapter 4, section 4.1

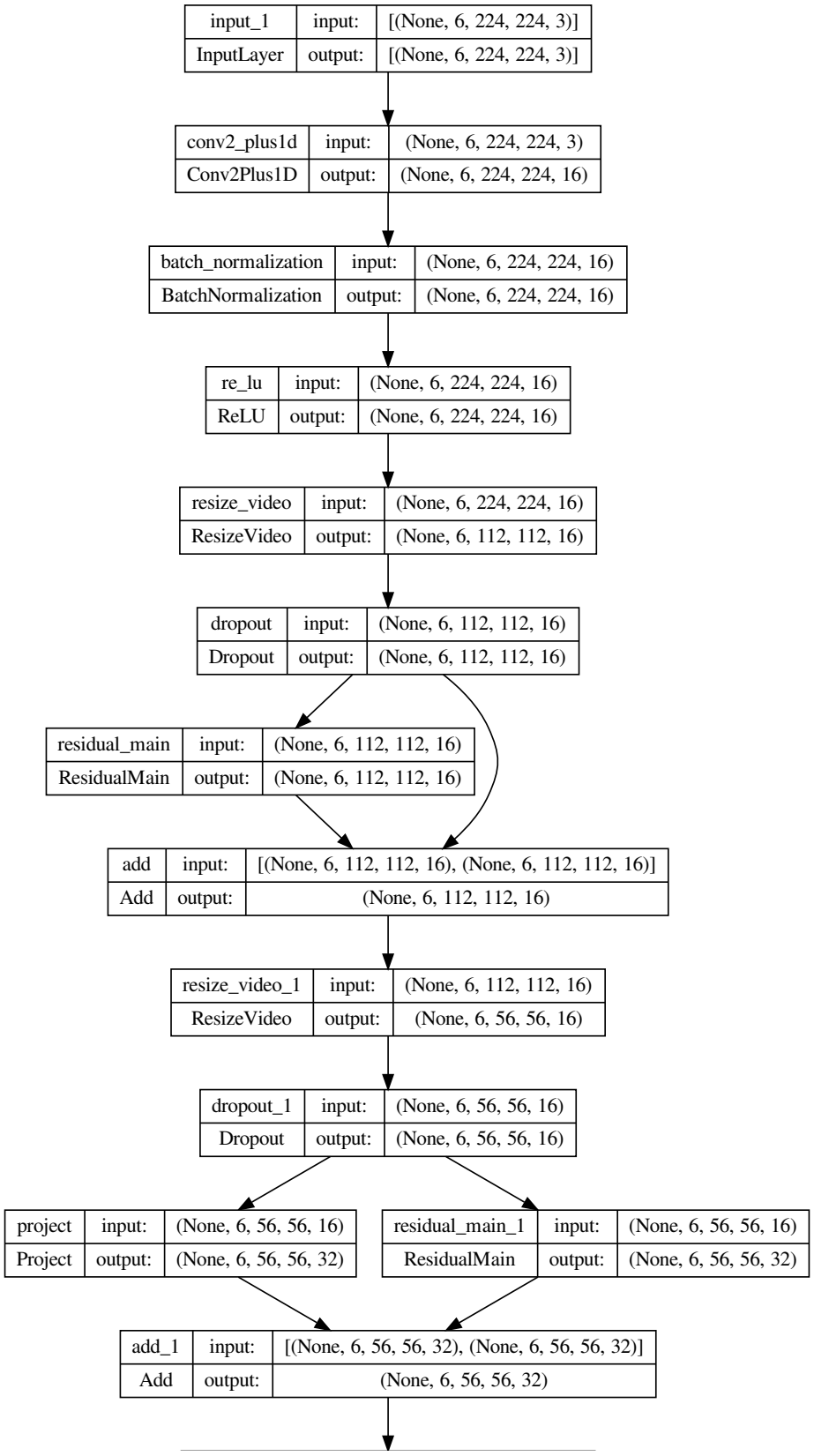


Figure A.1: Tensorflow plot of the proposed Neural Network for Valence-Aroual prediction, part 1.

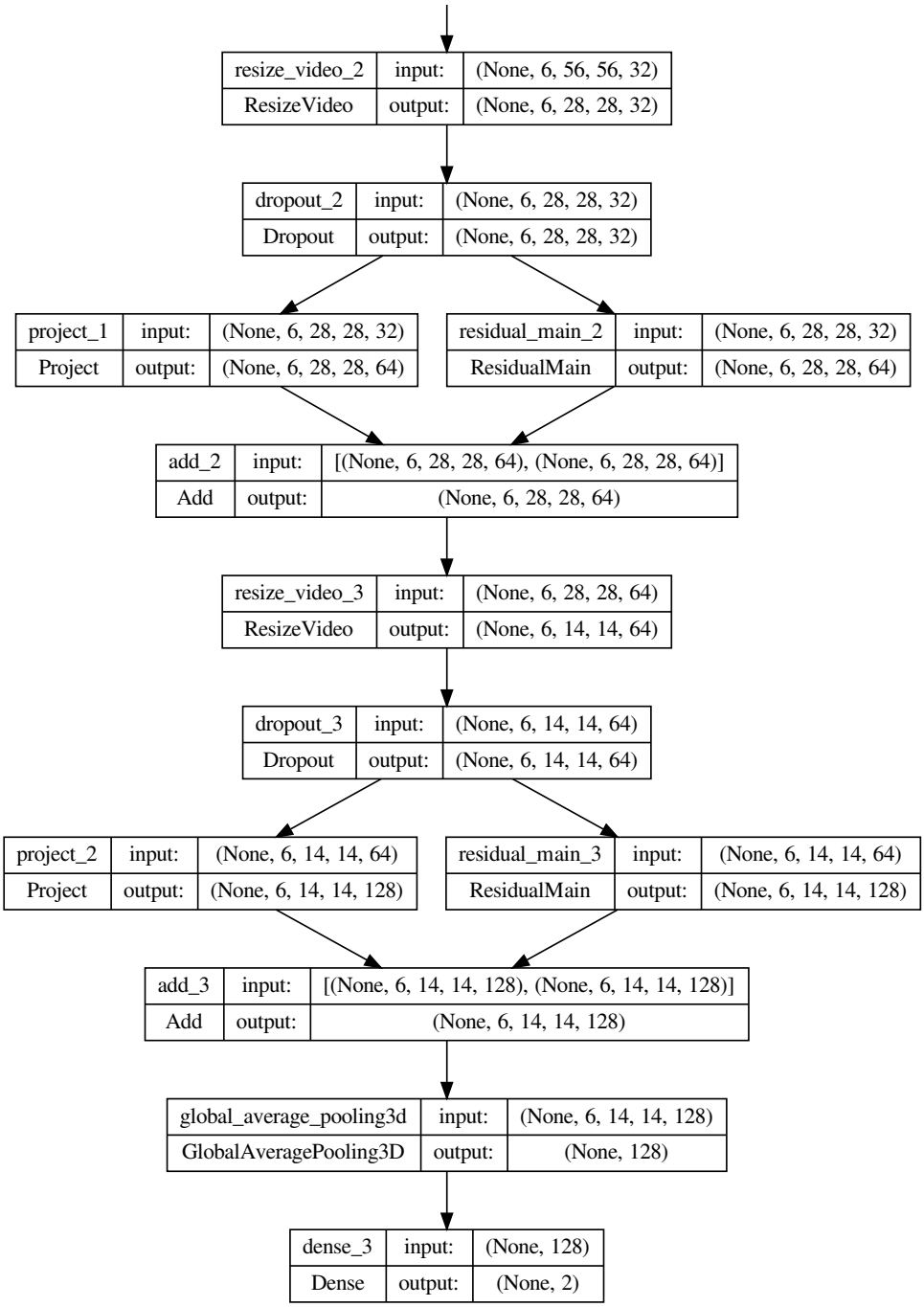


Figure A.2: Tensorflow plot of the proposed Neural Network for Valence-Aroual prediction, part 2.

List of Figures

2.1 A comparison between four common video game perspectives 9
2.2 Circumplex model of affect, from [84] 10
2.3 Example of the two main Artificial neural network topologies 12
2.4 Scheme of the convolution operation performed by the convolutional layer, including a numerical example. 16
2.5 Example of max-pooling operation. In this case the feature map's dimensionality is effectively reduced by a factor of 2 17
2.6 Example of CNN trained for object classification, from Matworks ebook . . 18
2.7 Encoder-Decoder basic structure. 19
2.8 Scheme of Transformer's encoder-decoder architecture, from [97] 20
2.9 Heatmap of some reviews with 5-star score, from [58]. Red intensity indicates what the model mostly focuses on. 22
2.10 Comparison of two encoder-decoder architectures, focusing on the attention mechanism, from [68] 23
2.11 A segment of a piano performance aside with its representation as performance events, from [48] 26
2.12 Relative attention visualized during music generation, from [48]. 28
2.13 Primer continuation performed with the Music Transformer. 29

3.1 A screenshot of Monkey island 2 [39] and Doom [62], discussed in this subsection 34
3.2 Common chord progression map for major scales, from [1] 37
3.3 Visualization of the spreading activation model proposed in [50] 38
3.4 Emotionally evocative events in-game, defined in [75] 39
3.5 A Circumplex model for color scripting in video games, proposed by [37]. . 40
3.6 One of the CNN architectures proposed by Makantasis et al. 41
3.7 Accuracy of emotion detection models proposed by Thao et al. [95] 42
3.8 A Survey of Affective Datasets of Audiovisual Content, from [64] 44
3.9 Circumplex model of affect limited to discrete values (high-low) for both dimensions. 47

3.10 Emotion annotation task conducted for creating VGMIDI dataset [35]. . . 48
3.11 Comparison between some existing emotion-labeled music datasets contained in [49] 49

4.1 Scheme of the proposed approach for video emotion detection. 52
4.2 (2+1)D vs 3D convolution. 55
4.3 A residual block from [44]. 56
4.4 Scheme of the custom ResNet proposed for the emotion detection task . . . 58
4.5 Scheme of the overall training pipeline presented in this section. 59
4.6 Scheme of the proposed approach for conditioned music generation. 60
4.7 Music transformer conditioning process. 62
4.8 Music Generation pipeline. 63
4.9 V-A estimation and Music generation pipelines combined. 64
4.10 Proposed Architecture as a Video game Music System 65

5.1 Training MSE of custom ResNet 69
5.2 AGAIN Games with first person perspective 71
5.3 AGAIN Games with isometric perspective 72
5.4 AGAIN Games with 2D side-scrolling perspective 72
5.5 Pictures taken during different iterations of the evaluation test performed at the Politecnico di Milano, Milan, Italy. 79
5.6 Results of the questionnaire part of the subjective experiment 83

A.1 Tensorflow plot of the proposed CNN for Valence-Aroual prediction, part 1. 106
A.2 Tensorflow plot of the proposed CNN for Valence-Aroual prediction, part 2. 107

List of Tables

5.1 Performance of custom ResNet on Train, Validation and Test set 69
5.2 Performance of custom ResNet for Valence and Arousal on Test Set 70
5.3 Performance of custom ResNet on LIRIS-ACCEDE Continuous. 70
5.4 Results by musical category, using only Valence annotations. 81
5.5 Results by musical category, using only Arousal annotations. 82
5.6 Results by musical category, merging both dimensions. 82
5.7 Results by affective dimension, merging all musical categories. 82
5.8 Normality test performed for Valence dimension on Distance, RMSE, PCC, for each music category. 83
5.9 Normality test performed for Arousal dimension on Distance, RMSE, PCC, for each music category. 84
5.10 Normality test performed merging all dimensions on Distance, RMSE, PCC, for each music category. 84
5.11 Normality test performed merging all categories on Distance, RMSE, PCC, for each affective dimension. 84
5.12 Analysis of variance computed for each metric (rows) and for each grouping (columns). 85
5.13 Post hoc analysis of Table 5.4 (Valence) for Distance metric. 86
5.14 Post hoc analysis of Table 5.6 (all Dimensions) for Distance metric. 86
5.15 Post hoc analysis of Table 5.6 (all Dimensions) for RMSE metric. 86
5.16 Post hoc analysis of Table 5.7 (all Categories) for Distance metric. 87

Ringraziamenti

Per prima cosa, partiamo dai ringraziamenti più pratici e direttamente connessi a questa tesi. Voglio ringraziare Luca Comanducci, che mi ha seguito per tutto lo sviluppo di questo lavoro fino alla sua conclusione, mostrando grande disponibilità ed esperienza nel darmi sempre i migliori consigli, soprattutto nei momenti più difficili. Grazie anche a Massimiliano Zanoni, che mi ha guidato nell'analisi dello stato dell'arte e soprattutto nella scelta dell'argomento di tesi, dandomi libertà creativa e numerosi stimoli. Con l'aiuto di entrambi ho potuto racchiudere e approfondire diverse mie passioni in questa tesi, trovando così la motivazione per raggiungere risultati che altrimenti avrei ritenuto inarrivabili.

Voglio poi ringraziare Michele per il controller Xbox, Gabriele per le cuffie over-ear e l'ISPL lab per il monitor: tutti oggetti fondamentali, che per giorni ho preso gentilmente in prestito per il test percettivo della mia tesi. Grazie infine a tutte le persone che hanno preso parte all'esperimento in questi ultimi mesi: far giocare più di 30 studenti ai videogiochi in piena sessione estiva è traguardo che ricorderò con gran soddisfazione.

Detto questo, restano un bel po' di persone che più indirettamente mi hanno aiutato a raggiungere questo traguardo. Dopo essermi trasferito a Milano, entrando a far parte di *Polifonia* ho potuto pian piano conoscere un incredibile numero di persone. Ad oggi, dopo ormai tre anni si è creato un bel gruppo di amici con cui ho condiviso innumerevoli esperienze e con cui mi impegnerò a trascorrere ancora molto tempo assieme, nonostante le inevitabili distanze geografiche che si creeranno.

Grazie anche ai miei compagni di corso con cui ho avuto il piacere di ideare e sviluppare progetti di gruppo: abbiamo condiviso maratone estenuanti per rispettare le scadenze di diversi esami, e nonostante le poche ore di sonno e le continue ansie ricordo con grande piacere ogni nostra fatica.

Voglio anche menzionare *Atletica Polimi* e *Alpine Polimi*: grazie a questi gruppi di studenti ho riscoperto il piacere dello sport e tutte le esperienze che ho raccolto saranno fondamentali per le mie scelte future.

Grazie anche al *Polipsi* e alla mia psicologa, che mi hanno aiutato a vivere più serenamente questi anni di studio e a riconoscere il valore di ciò che faccio, distinguendolo da ciò che sono.

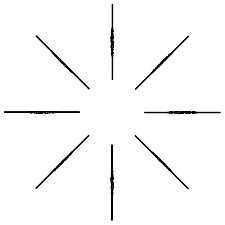
E per chiudere la parentesi milanese, non posso non ringraziare i miei coinquilini Asia, Elena, Luca, Giacomo e Nicolò. Con la vostra presenza ho potuto chiamare il nostro appartamento in via Bassini "Casa".

Volgendo lo sguardo verso Verona, voglio innanzitutto ringraziare i miei genitori: nonostante io sia sempre stato molto criptico riguardo i miei studi e la mia vita a Milano, mi hanno supportato ogniqualvolta avessi bisogno di qualcosa.

Prima di arrivare fin qui, la mia esperienza universitaria è iniziata a UniVr: senza le innumerevoli studiate di gruppo e il sostegno di molti miei compagni di corso non sarei riuscito a superare le enormi insidie della laurea in informatica, che a volte sembrava più un brutto e difficile videogioco che un corso di studi.

Un super ringraziamento va poi ai diversi amici conosciuti a *Scout*, con cui tuttora ci vediamo ad ogni mio ritorno a casa e grazie ai quali trascorro ogni estate intense settimane in montagna. Molti hanno anche partecipato all'esperimento della mia tesi, quindi per voi doppi punti.

Infine, per chiudere ringrazio i *5 pezzi di exodia*, un gruppo storico che non ha mai perso coesione, nonostante le nostre frequenti distanze. Qualunque sia il contesto, se ci troviamo assieme il decoro e la compostezza sono inevitabilmente compromessi, in favore dell'ilarità.



Questo è un traguardo che innanzitutto ho costruito io con impegno e perseveranza, ma tutti i contributi elencati, più quelli che purtroppo avrò dimenticato, hanno arricchito e allietato il viaggio, che sono felice di aver concluso con grande soddisfazione e una buona dose di fiducia verso il futuro.

F. Z.