Metric learning for video to music recommendation

Laure Prétet

▶ **To cite this version:**

Laure Prétet. Metric learning for video to music recommendation. Multimedia [cs.MM]. Institut Polytechnique de Paris, 2022. English. NNT : 2022IPPAT005 . tel-03638477

HAL Id: tel-03638477
https://theses.hal.science/tel-03638477

Submitted on 12 Apr 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2022IPPAT005

Thèse de doctorat

Metric Learning for Video to Music Recommendation

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité de doctorat : Signal, Images, Automatique et Robotique

Thèse présentée et soutenue à Palaiseau, le 24 janvier 2022, par

LAURE PRÉTET

Composition du Jury :

Frédéric Bevilacqua Directeur de recherche, IRCAM-CNRS-Sorbonne Université (UMR STMS)	Président, Rapporteur
Jenny Benois-Pineau Professeur, Université de Bordeaux	Rapporteure
Estefania Cano Directrice de recherche, AudiosourceRe	Examinatrice
Guillaume Gravier Directeur de recherche, CNRS	Examinateur
Stéphane Lathuilière Maître de conférences, Télécom Paris (LTCI)	Examinateur
Alexander Schindler Docteur, Austrian Institute of Technology	Examinateur
Geoffroy Peeters Professeur, Télécom Paris (LTCI)	Directeur de thèse
Gaël Richard Professeur, Télécom Paris (LTCI)	Co-directeur de thèse

TÉLÉCOM PARIS

DOCTORAL THESIS

Metric Learning for Video to Music Recommendation

Author:
Laure PRÉTET

Supervisors:
Dr. Geoffroy PEETERS
Dr. Gaël RICHARD

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

ADASP group
Image, Data, Signal Department

January 24, 2022

Abstract

Music enhances moving images and allows to efficiently communicate emotion or narrative tension, thanks to cultural codes common to the filmmakers and viewers. A successful communication requires not only a choice of tracks matching the video's mood and content, but also a temporal synchronization of the audio and visual main events. The art of selecting and synchronizing music tracks to existing videos is at the heart of an industry called music supervision. Traditionally, the music supervision task is carried out manually by experts. It therefore requires to collect, annotate and organize a dedicated music catalog. The larger the catalog, the more diverse the recommendations can be, but the longer it is to annotate and browse it. A good knowledge of visual communication through video is also a key factor for success.

In this dissertation, we study the automation of tasks related to music supervision. The music supervision problem generally does not have a unique solution, as it includes external constraints such as the client's identity or budget. It is thus relevant to proceed by recommendation. As the number of available musical videos is in constant augmentation, it makes sense to use data-driven tools. More precisely, we use the metric learning paradigm, which allows to learn relevant projections of multimodal data, and to efficiently retrieve recommendations for each query.

First, we address the music similarity problem, which is used to broaden the results of a music search. We implement an efficient content-based imitation of a tag-based similarity metric. To do so, we present a method to train a convolutional neural network from ranked lists. Then, we focus on direct, content-based music recommendation for video. We adapt a simple self-supervised system and we demonstrate a way to improve its performance, by using pre-trained audio features and learning their aggregation. We then carry a qualitative and quantitative analysis of official music videos to better understand their temporal organization. Results show that official music videos are carefully edited in order to align audio and video events, and that the level of synchronization depends on the music and video genres. With this insight, we propose the first recommendation system designed specifically for music supervision: the Seg-VM-Net, which uses both content and structure to perform the matching of music and video.

Résumé substantiel

À l'écran, la musique permet de communiquer, à travers des codes culturels établis, des émotions ou des éléments narratifs clés. Cette communication repose non seulement sur un choix judicieux de morceaux pour la bande son, mais aussi sur leur synchronisation avec les événements saillants de la vidéo. La tâche qui consiste à conseiller et réaliser cette association est au centre de l'industrie de la supervision musicale. Traditionnellement, elle est effectuée manuellement par des experts. L'activité de supervision musicale implique donc de rassembler, annoter et organiser de vastes catalogues musicaux dédiés. Plus le catalogue est étendu, plus les recommandations peuvent être fines et variées, mais plus il est coûteux en temps d'annoter et de parcourir tous les morceaux. Il est également essentiel de maîtriser les codes de la communication visuelle pour faire les meilleures suggestions de titres, étant donné une vidéo.

Dans cette thèse, l'on étudie l'automatisation des tâches liées à la supervision musicale. La quête de la musique idéale n'a généralement pas de solution unique, puisqu'en plus de l'analyse des contenus audiovisuels, il faut tenir compte de contraintes légales et budgétaires. Nous procédons donc par recommandation.
Alors qu'une quantité toujours croissante de musique et de vidéo est produite chaque jour, il est raisonnable d'envisager une approche par apprentissage. Plus précisément, nous utilisons l'apprentissage métrique, qui produit des représentations communes aux données musicales et visuelles. Ces représentations sont étudiées pour permettre des recommandations rapides selon un critère pré-déterminé d'appariement entre deux médias.

Dans un premier temps, nous abordons un problème de similarité musicale, dont le but est d'élargir le champ de recherche dans les catalogues musicaux. Nous implémentons une imitation efficace d'une métrique de similarité par critères, applicable directement aux fichiers audio non annotés. Cette méthode repose sur des réseaux de neurones convolutionnels, pour lesquels nous proposons une méthode d'entraînement directement à partir de listes de recommandation. Nous démontrons l'efficacité de notre démarche comparée à une approche indirecte par estimation des critères. À l'aide d'exemples audio, nous explorons l'espace métrique obtenu et nous mettons en évidence l'apprentissage de certaines caractéristiques audio, matérialisées par les dimensions de cette espace.

Puis, nous nous concentrons sur la recommandation directe de musique à partir d'une vidéo. Nous adaptons un système autodidacte simple et montrons comment en améliorer les performances, grâce à de meilleures représentations audio en entrée et un apprentissage de leur agrégation temporelle. Le critère d'appariement utilisé ici est la co-occurrence des deux médias au sein d'un même document audiovisuel (clip musical) d'une base de données. Nous menons ensuite une étude sur les clips musicaux, afin de mieux comprendre comment y sont articulés les événements audio et vidéo. Cette étude compred un premier volet qualitatif au cours duquel nous interrogeons des experts du domaine et formulons des hypothèses sur

la façon dont sont couplés musique et image. Un deuxième volet avec étude quantitative d'un corpus de clips musicaux vient confirmer ces hypothèses. Nos résultats démontrent avec quel soin la musique et l'image peuvent être synchronisés, mais aussi que le niveau de co-occurrence des différents événements dépend de plusieurs critères, tels que le genre musical ou vidéo. Partant de ce constat, nous présentons le premier système de recommandation dédié à la supervision musicale : le Seg-VM-Net. Le Seg-VM-Net reprend la méthode d'appariement de contenu du premier système autodidacte. Mais il tient également compte de la structure des musiques et des vidéos, en découpant ces derniers en segments sémantiquement cohérents. À l'inférence, la recommandation est faite en alignant les séquences de segments dont le contenu correspond au mieux. Avec des métriques simples et des exemples, nous démontrons la supériorité du Seg-VM-Net comparé à l'état de l'art pour la recommandation musique-vidéo.

Acknowledgements

This research project would not exist without the vision of Clément Souchier, entrepreneur in the music industry and technology enthusiast. Clément saw straight away the potential of AI for the music supervision industry and understood extremely quickly what was possible to do or not, resulting in a very realistic project perfectly suited for a PhD topic: this one. Thank you Clément for funding my PhD work through your companies - Creaminal first, and then Bridge.audio.

This thesis would clearly not be such a success without the expertise of my two advisors Geoffroy and Gaël. Gaël, you always came to the rescue when we were looking for innovative ideas, related work, help with the administration or contacts in the MIR community. Thank you for the time you took out of your busy schedule to guide me through these three years. It was always a pleasure to chat with you and your eternal good mood was appreciated, not only by me, but by all the department. President Peeters, you are not only a distinguished, rigorous and creative researcher. You are also a Professor, and I must admit that I was impressed by your pedagogical skills. You knew when to give me time and when to take action; when I needed to be cheered up and when I needed directions; and you always answered my numerous emails entitled "help", "SOS" or "oskour". I cannot think of a better match for my thesis supervisor. Soon enough it came to be my time to help; I hope I delivered.

Thank you to all my colleagues of the ADASP team and especially the MIP-Frontiers squad for all the good time we spent together and the mutual support along this journey: in a word for your friendship. Special thanks go to Stéphane, Slim, Attilio, Marco and Alasdair for their valuable insight on the multimodal and video domains which were brand new to us.

Thank you to all my colleagues at Creaminal and Bridge.audio, especially to Mahel and Rémy; we had a lot of fun implementing restaurant selection algorithms and figuring out what to do with hypothetical thousands of euros for research. Also, special thanks go to Félix and Stéphanie for introducing me to the music supervision world. Of course, I would like to express my gratitude to Maxime, Jack and Alexandre for the time they took to answer my questions. A warm thank you goes to my jury for accepting to review my work, and to Mathilde for proofreading it.

Finally, it's time to end this embarrassingly long, and still probably incomplete enumeration of supportive fellows. I am very grateful to my family and my friends who supported me during all these years of higher education. Nico, Manon, Victor, Marion, Bastien and all climbing partners deserve a highlight for the many intense moments we spent around this addictive activity. Thank you Guillaume for all these years together. From leaving La Réunion to having a home with you, there were many steps (but Touille is definitely an important one!).

Contents

List of Figures

List of Tables

List of Abbreviations

MIR	Music Information Retrieval
STFT	Short Time Fourier Transform
CQT	Constant-Q transform
DNN	Deep Neural Network
CNN	Convolutional Neural Network
FCN	Fully-Connected Network
SSL	Self Supervised Learning
MAP	Mean Average Precision
nDCG	Normalized Discounted Cumulative Gain
AV	Audio/Video
V/M	Video/Music
V2M	Video-to-Music
M2V	Music-to-Video
OMV	Official Music Video
SE	Structural Events
DTW	Dynamic Time Warping

List of Symbols

	Music similarity
$\mathcal{S}$	oracle music similarity function
$R^{\mathcal{S}}(t) = [r_1(t), \ldots, r_{N-1}(t)]$	reference list of recommended tracks for the target track t
$R^{\hat{\mathcal{S}}}(t) = [\hat{r}_1(t), \ldots, \hat{r}_{N-1}(t)]$	estimated list of recommended tracks for the target track t
	Language of music-video clips
$l_{segment}$	music functional segments positions
Δ_{bar}	bar duration
l_{bar}	downbeat positions
Δ_{beat}	beat duration
$f_{shot}(t)$	shot boundary probability
l_{shot}	shot boundary positions
$\Delta_{shot}^{\max}$	most common shot duration
	Video/Music recommendation
$c \in \{1, \ldots, C\}$	a specific AV clip
C	number of clips in the catalog
c^m / c^v	full music/ video track of c
x_c^m / x_c^v	input music/ video feature for c
e_c^m / e_c^v	music/ video embedding for c
$f_m(\cdot)$/ $f_v(\cdot)$	embedding function for music/ video
$\{c_1, \ldots, c_{K_c}\}$	sequence of temporal segments of c
K_c	number of segments for c
$\{c_1^{m/v}, \ldots, c_{K_c}^{m/v}\}$	sequence of music/ video segments of c
$\{x_{c,1}^{m/v}, \ldots, x_{c,K_c}^{m/v}\}$	input audio/ video features of the segments of c
$\{e_{c,1}^{m/v}, \ldots, e_{c,K_c}^{m/v}\}$	audio/ video embeddings of the segments of c

Chapter 1

Introduction

Music is a specific type of sound that is structured both in time (rhythm, i.e. events location and duration) and in frequency (melody and harmony). It has been used since prehistory for spiritual purposes, to convey stories, or to accompany dancing. Today, as listening to music is a hobby for many of us, a full industry has developed around producing, performing, and distributing music. Several challenges have thus emerged around the understanding of music, the organization of music collections, and music recommendation. As such, it is the topic of a dedicated research field: Music Information Retrieval (MIR).

Video is a visual representation produced by the evolution of images over time. Each day, an ever-growing quantity of videos is created by professionals (for advertisement, movies, TV series, etc) and individuals (for Instagram, TikTok, YouTube, etc). Watching video is not only a popular hobby, but also a powerful way to acquire knowledge (through documentaries, news, CCTV). While video media are used every day to convey information, video analysis has gained a lot of interest in research, with high stakes in domains such as augmented reality, video coding or medical imaging. Therefore, video analysis is a wide research area, which is part of the computer vision domain.

Although sound and image result from two different physical phenomena, they share many relationships in our everyday life. They often complement each other to convey information about our environment: when a person is speaking, an orchestra is playing, or a car is passing by, we use both visual and audio information to best understand the scene.

When creating a video, it is necessary to take this semantic correlation into account: video is best comprehended when associated with sound. Spoken voice is employed to communicate stories (in movies) or concepts (in advertisements). But when emotions or atmospheres must be conveyed, music is commonly used too (see Figure 1.1). Conversely, video can be used as an illustration for musical works (in official music videos). To analyze audiovisual documents, multimodal studies and systems have grown rapidly in the past years: the SoundNet of Aytar, Vondrick, and Torralba, 2016 performs acoustic object and scene classification; Afouras et al., 2018 trained a model for lip-reading (multimodal speech recognition); Noulas, Englebienne, and Kröse, 2012 used both audio and video for enhanced speaker diarization; and Parekh, 2019 studied extensively audiovisual scene analysis.

However, when addressing the music-video domain specifically, the associated research area is still emerging. As music videos have become a non-negligible part of music consumption with TikTok, YouTube, and other platforms, we expect music-video analysis to be a more and more widespread research topic.

FIGURE 1.1: Different types of video that use a music soundtrack: a video game [top left], a short movie [top right], an official music video [bottom left] and a commercial [bottom right].

1.1 Context: Music supervision

Music can influence our perception of video, and this has been used to convey messages or enhance storytelling. For example, in Hitchcock's movies, music accentuates the suspense and the narrative tension. In commercials, certain brands like Apple or Yves Saint Laurent have a distinctive musical signature, which conveys their identity. Music is also commonly used in video games, TV shows, audiobooks, and podcasts. To best associate both media, a dedicated profession was created. **Music supervision** is the industry that aims specifically at recommending and synchronizing music to video. The activity of music supervision experts consists in finding the most suitable music to serve as a soundtrack for a video. Such videos can be commercials, movies, TV series, or even fashion shows (Inskip, Macfarlane, and Rafferty, 2008). To make the best suggestions, music supervision companies generally maintain a vast catalog of music tracks with carefully curated metadata. Music supervision specialists (also called **music supervisors**) rely on their musicological expertise and on the knowledge of their catalog. A brief description of a music supervision company, as well as an overview of a typical music supervision workflow, can be consulted in Appendix A.

As of today, the field still requires a large amount of manual work, either to annotate each track or to listen to as many tracks as possible to find the best match. While an increasing quantity of music is produced each day, representation of the full artistic landscape has become a real challenge for music supervisors, who usually work in very limited time. In this thesis, we are interested in creating and adapting machine learning tools to alleviate repetitive tasks related to music supervision. Driven by this industrial use case, we will address **two music recommendation tasks**: music similarity and Video-to-Music (V2M) recommendation (see Figure 1.2).

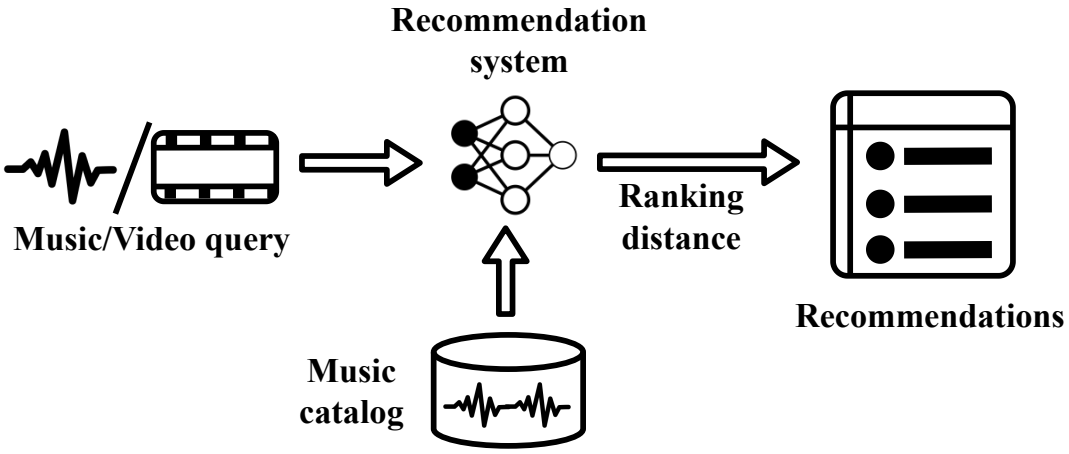


FIGURE 1.2: All recommendation systems studied in this thesis fulfil the same requirements: from a raw music track or video, retrieve relevant music tracks from a catalog.

1.2 Objective and motivation

1.2.1 Music similarity

In the first part of this work, we aim at developing an efficient music similarity system for music supervision. Here, we address the music similarity problem as a recommendation problem, for which both the query and the recommended items are music tracks.

In music streaming services, automatic recommendation algorithms are developed to help users navigate the large music catalogs. These algorithms can be used to retrieve a ranked list of music tracks based on their similarity with a target track. Similarity-based music recommendation is equally valuable to music supervision companies, which also have to manage and organize large catalogs. Indeed, music supervision projects often require several similar track suggestions in order to best handle clearance and budget issues.

Common music similarity algorithms employ collaborative filtering data (i.e., user listening history, as in Oord, Dieleman, and Schrauwen, 2013), or tags (Clifford, 2007) associated to each music track. However, manually tagging the music tracks is unfeasible for large datasets, and usage data is not available for private music supervision catalogs. The more additional information is required to perform the similarity task, the longer it takes to integrate new tracks into the catalog. Therefore, the growth of the catalog is limited by the type of algorithm used for searching it.

The last option is to rely directly on the audio content to perform the matching (Aucouturier and Pachet, 2002). Starting from a tag-based music similarity system, we want to train an audio-based algorithm to mimic its behavior. We explore various network architecture and training strategies, in order to achieve satisfying retrieval performance without tagging any track.

1.2.2 Video/Music recommendation

The second part of this work addresses music recommendation given a video query. Finding an appropriate soundtrack to emphasize the video content is a time-consuming exercise. This explains the success of commercial systems such as MatchTune or research papers such as "Look, Listen and Learn" (Arandjelovic and Zisserman, 2017).

Our motivation is to assist music supervision experts in their activity when they produce music recommendations for each project's videos. We therefore study cross-modal Video/Music (V/M) recommendation, i.e. recommending a music track for a video or vice versa.

So far, many approaches in V/M recommendation rely on the Self Supervised Learning (SSL) paradigm, which is inspired by the day-to-day human experience. In our everyday life, the co-occurrence of events across our senses allows us to efficiently acquire a large amount of knowledge without the need for ground truth annotations. For example, we are able to associate engine sounds to cars, musical sounds to specific instruments, a sour taste to the picture of a lemon, and so on. From a machine learning perspective, it is possible to learn a matching of audio and video data, using only their co-occurrence in large non-annotated datasets. This is how SSL systems can learn effective data representations without any external annotation. SSL systems for V/M recommendation can be trained using a classification task (Aytar, Vondrick, and Torralba, 2016), a regression task like source separation (Zhao et al., 2018), or more commonly a metric learning task (Arandjelovic and Zisserman, 2017). Thereby, a first step is to reproduce and adapt a SSL system for V/M matching.

On the other hand, the music supervision industry represents a valuable source of domain knowledge for the video-music recommendation problem. Our second objective is to combine SSL and industrial domain knowledge into a proficient V/M recommendation system.

Although we eventually target an application scenario featuring any type of video (commercials, movies, TV series), in this dissertation, we focus on music videos. Indeed, this type of data is easy to collect for training and less noisy, as it generally does not feature speech or non-musical sounds.

1.3 Challenges

A new scientific domain. To our knowledge, there is very little literature about the music supervision problem. The main contribution, which helped define the problem, was made by Inskip, Macfarlane, and Rafferty, 2008, who interviewed experts in the domain and described their workflow. The authors mention that "the clearest briefs appear to be moving images", suggesting that other types of data (emotion, textual description, reference soundtracks) are not always necessary to perform the task. In this work, we will focus on the uninformed scenario, where only music and video data is available.

Music similarity learning from ranked lists. Without explicitly tagging the music tracks, we want to estimate the similarity between two tracks based on their audio content. Since we have access to similarity data in the form of oracle recommendations (ranked lists), we propose and evaluate several strategies to learn a similarity metric from ranked lists.

Efficient and unconstrained content-based recommendation. The specificity of our study is that the matching criterion for the V/M recommendation is ill-defined: we are searching for music tracks that "would make a good soundtrack for the video query". While mood can be used to match music and video, we do not wish our system to be restricted to a matching based only on a mood criterion. Although

ill-defined, the matching of music and video can be learned to some extent. However, current methods require to train heavily parametrized systems from scratch on huge datasets. We believe a more computationally efficient approach is possible, by reusing state-of-the-art systems instead of training them again. We put effort into designing data- and energy-efficient systems and we track their CO_2 emissions in Appendix B.2.

Structure-aware recommendation. In addition to the adequacy of content, adequacy of structure is crucial in music supervision to obtain relevant recommendations. While both music and video have specific temporal organizations, most current works in cross-modal recommendation do not consider those and only focus on globally recommending a media. In our case, we want to integrate temporal structure constraint, as both media are devoted to being synchronized on-screen (typically by editing the music). This implies first defining what audiovisual structure is, by analyzing professionally produced music videos.

1.4 Contributions and Outline

In this work, we explore ways to assist the music supervision task by automating part of the related recommendation tasks. In each chapter, we propose one way to assist the music supervision task.

Method to learn a ranking of music tracks for similarity search: In Chapter 4, we study music ranking by similarity. Inspired by image recognition models, we train a supervised system with tag-based similarity annotations to rank the tracks based only on audio. Imitation of a tag-based similarity metric involves performing metric learning from ranked lists. To achieve this, we train a convolutional neural network with a triplet loss. We compare different triplet mining methods and neural network architectures. We demonstrate that the metric learning method leads to superior results compared to an auto-tagging method. We obtain a high-performing music similarity system: the MuSimNet. This work led to the following publication:

Laure Prétet, Gaël Richard, and Geoffroy Peeters (2020). "Learning to Rank Music Tracks Using Triplet Loss". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Barcelona, Spain. DOI: 10.1109/icassp40776.2020.9053135

Improvement of content-based Video/Music recommendation: In Chapter 5, we study and improve a V/M recommendation system. We build upon a recent V/M retrieval system (the VM-Net), which originally relies on a music representation obtained by a set of statistics computed over handcrafted features. We show that using pre-trained audio features significantly improves performance compared to using handcrafted features. Additionally, we show that learning the temporal aggregation of the features improves performance compared to using a statistical aggregation of the features. This work led to the following publication:

Laure Prétet, Gaël Richard, and Geoffroy Peeters (2021a). "Cross-Modal Music-Video Recommendation: A Study of Design Choices". In: *Special Session on RLASMP, IJCNN (International Joint Conference on Neural Networks)*. Virtual Event. DOI: 10.1109/IJCNN52387.2021.9533662

Study on relevant structure elements for recommendation: In Chapter 6, we study the relationship between music and video temporal organization. We do this for the case of official music videos, with a quantitative and a qualitative approach. Our assumption is that the movement in the music (downbeats and functional segments) is correlated to the one in the video (cuts). To validate this, we first interviewed three internationally recognized music video experts and asked detailed questions about V/M matching, structure, and synchronization. We then performed a large-scale quantitative analysis of a corpus of official music video clips, using MIR description tools and computer vision tools. To do so, we manually annotated the video genres. Our study confirms that a "language of music-video clips" exists; i.e., editors favor the co-occurrence of music and video events. We also highlighted that the amount of co-occurrence depends on the music and video genres. This work led to the following publication, which won the "Best Video Award" at ISMIR 2021:

Laure Prétet, Gaël Richard, and Geoffroy Peeters (2021b). "Is there a "language of music-video clips"? A qualitative and quantitative study". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Virtual Event. URL: https://archives.ismir.net/ismir2021/paper/000067.pdf

Structure-aware Video/Music recommendation system: In Chapter 7, we combine the SSL metric learning paradigm of the VM-Net to the previously acquired domain knowledge on audiovisual structure. Our novel approach, the Seg-VM-Net, considers not only the full music-video clips, but rather shorter segments for training and inference. We find that using semantic segments and ranking the tracks according to sequence alignment costs significantly improves the system's performance on the V/M retrieval task. We investigate the impact of different ranking metrics and segmentation methods. This work was submitted as a patent and a journal paper:

Laure Prétet et al. (2021). *Procédé d'entraînement d'un modèle statistique pour qu'il soit configuré pour être utilisé pour recommander, à partir d'un média d'un premier type, un média d'un deuxième type, et système associé*. Patent application filed.

Laure Prétet et al. (2022). "Video-to-Music Recommendation using Temporal Alignment of Segments". In: *IEEE Transactions on Multimedia*. DOI: 10.1109/TMM.2022.3152598.

Finally, Chapter 8 concludes this dissertation and discusses its perspectives.

Chapter 2

Fundamentals and Related work

In this chapter, we state the scientific scope of our work, as well as useful definitions for the rest of this manuscript. We first introduce the general machine learning tools that we used to develop specialized music recommendation systems. We then give an overview of the related work for our two main domains: music similarity and Video/Music matching.

2.1 Tools and useful definitions

In the scope of this study, we will rely on **machine learning** tools to perform recommendation tasks. Machine learning designates a category of algorithms where processing steps are not programmed explicitly, but directly inferred by the model from example data. Machine learning models are parametric functions f_θ which transform a given input x into an output $\hat{y}$:

$$\hat{y} = f_\theta(x). \tag{2.1}$$

In this dissertation, we only consider the case where both x and $\hat{y}$ belong to a multidimensional space. For example, in a music classification problem, x represents the audio data, and $\hat{y}$ represents the class(es) associated to x by the function, which is an approximation of the ground truth class(es) y. The set of parameters θ of the model is adjusted during an iterative process called **training**. During training, batches of example data are provided to the model. The data samples are drawn from an ensemble called the **training dataset**. Training datasets are meant to mimic the true distribution of x and y, such that the model's performance can be transferred to real-world data - in practice however, they only are a rough approximation. The quality of the model's outputs $\hat{y}$ are then evaluated using a **loss function** $\mathcal{L}$. The choice of $\mathcal{L}$ is made according to the task at hand. The parameters θ are updated towards a direction that minimizes the error $\mathcal{L}(y, \hat{y})$, and the procedure starts over with new example data and the refined model. Training is interrupted as soon as the model's outputs are judged of sufficient quality. The model can then be used to produce predictions for unseen data: it is the **inference** step.

Different **paradigms** can be used for training. In the case of **supervised learning**, the loss function $\mathcal{L}$ will compare the model's output $\hat{y}$ to a **ground truth** output y obtained from the training dataset. Ground truth data are typically obtained from human annotations. $\mathcal{L}$ will apply a high penalty if $\hat{y}$ and y do not match for a given input x. **Unsupervised learning** is used when no ground truth data y is available. In this case, $\mathcal{L}$ forces the model to discover patterns within the input data and structure the output space according to said patterns. If no ground truth output is available, but it is possible to collect prior knowledge about the input data, then a third paradigm is preferred: **Self-supervised learning**. Here, $\mathcal{L}$ ensures that the model

encapsulates the desired knowledge and creates relevant, yet compressed representations of the data. General knowledge can be based on common sense (e.g., picture x was rotated by 180 degrees, video x is played in reverse), or more specific domain knowledge (audio file x^m constitutes the soundtrack of video file x^v). In practice, this is done by automatically generating target outputs and training the model on the resulting **pretext task**, without any external annotation.

We will use supervised learning for music similarity in Chapter 4. Self-supervised learning will be used to learn Video/Music matching from large non-annotated datasets of music videos in Chapters 5 and 7.

2.1.1 Neural networks

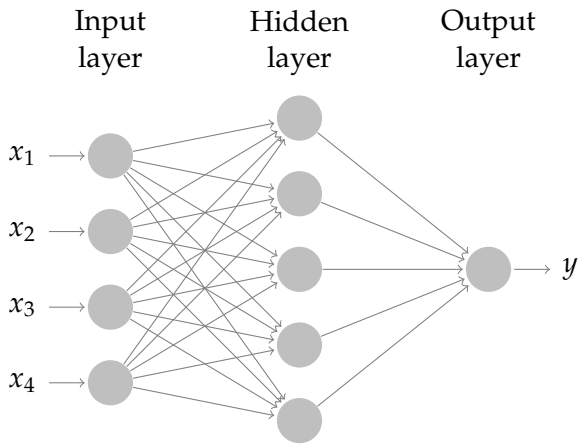


FIGURE 2.1: A fully-connected network with one input layer, one processing (hidden) layer and one output layer.

Nearly all machine learning algorithms used in this work are artificial neural networks. Artificial neural networks (or simply **neural networks)** are algorithms whose design is inspired by the human brain, hence the name. Neural networks have become essential in today's research because they led to state-of-the-art performance in the most popular machine learning tasks: image recognition (Horak and Sablatnig, 2019), audio source separation (Mitsufuji et al., 2021), natural language processing (Priyadarshini, Bagjadab, and Mishra, 2020), etc.

Neural networks are a family of models which are composed of **units** (also called neurons), organized in a succession of **layers**. Each unit processes n inputs $x = (x_1, ..., x_n)$ and applies a non-linear **activation function** g before passing the resulting output y to the units of the next layer. A widely used activation function is the ReLu function (Nair and Hinton, 2010): $\sigma(y) = \max(y, 0)$. The output of a unit is called the **activation**. The units are connected to each other and each connection is affected a weight $w = (w_1, ..., w_n)$. The total set of weights form the parameters θ that are tuned during the training. In this manuscript, we only consider neural networks whose inputs and weights are real-valued.

The training of a neural network can be decomposed into two phases (Rumelhart, Hinton, and Williams, 1986). During the **feed-forward** step, the input data goes through the succession of transformations defined by the network layers. The error of the resulting estimates is computed using a differentiable **loss function** $\mathcal{L}$. Then, during the **back-propagation** step, the gradient of the loss with respect to the weights is computed and θ is updated accordingly by gradient descent:

$$\theta \leftarrow \theta - \lambda * \frac{\partial \mathcal{L}}{\partial \theta}, \tag{2.2}$$

where λ is called the **learning rate** and defines the "speed" at which the parameters will be updated. These two phases are repeated a constant number of times to form an **epoch** (usually a full pass over the training dataset), and several epochs are required to complete the training. Over the years, a large diversity of gradient descent algorithms has been developed to optimize training (Goodfellow, Bengio, and Courville, 2016). In this manuscript, we focus on **minibatch** gradient descent methods, in which θ is updated after a small number of data samples have gone through the forward pass. Minibatches offer a compromise between optimizing on the full dataset (computationally infeasible on large datasets) and optimizing one sample at a time (often too unstable).

Overfitting. Neural networks are trained until they converge or reach satisfying estimation performance. One way to identify the final state is by monitoring the loss value on a held-out set called the **validation data**, which is not used for parameter updates. After a certain number of epochs, networks typically continue improving on training data, while degrading on unseen data, i.e., they tend to **overfit**. Overfitting models are functions whose parameters depend only on the example data of the training set, and not the intended trend or structure inherent to the task at hand. Interrupting the training before it reaches this critical point is called **early stopping**: this ensures that the system's performance will be optimal for inference. Real-life application scenarios are then emulated by computing metrics on a third, unseen data collection called the **test set**. Complex models (which have a high number of parameters) are more powerful and can learn the most complex interactions, but they also tend to overfit faster. As a result, they are not able to generalize to unseen data. The smaller the training set, the higher the risk of overfitting. That is why other **regularization** techniques, such as L_1, L_2, dropout (Srivastava et al., 2014) or batch normalization (Ioffe and Szegedy, 2015) are commonly used to limit the model's complexity (in complement to early stopping).

Hyperparameters. Unlike the weights θ, the learning rate λ cannot be tuned by gradient descent, and has to be adjusted manually via a grid search; it is hence considered a hyperparameter of the model. Neural networks are heavily configurable models, and thus have a variety of hyperparameters, including the number of layers, of units per layer, their arrangement, or the choice of the activation functions g. In this thesis, we favor the reuse of pre-existing, well-tried architectures over exploration of the hyperparameter space.

Input data. Neural networks take input data in the form of numerical arrays. 1D arrays may represent temporal series; 2D arrays can represent monochrome images; 3D arrays, color images; 4D arrays, videos; etc. Audio and video documents can be given as input directly or preprocessed to extract specific **feature** vectors (characteristic values that describe the data). Precomputed features help the network train faster and use less computational resources. For example, raw audio data (sampled version of the physical sound waves) is stored as waveforms, which are one-dimensional temporal series. Waveforms can be processed directly for optimal time precision (Dieleman and Schrauwen, 2014; Ravanelli and Bengio, 2018), but as music

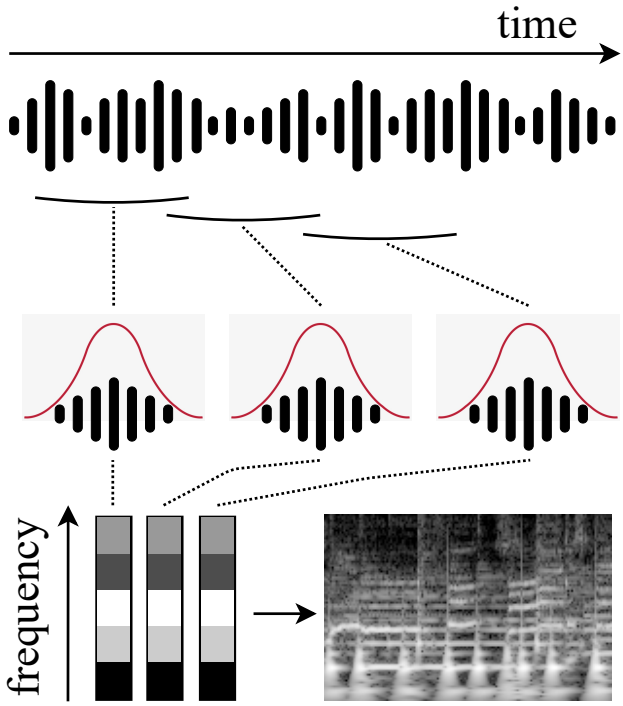


FIGURE 2.2: Illustration of the decomposition of an audio signal into a time-frequency representation (spectrogram).

is typically sampled at 44,100 Hz, this is computationally costly. As a result, for recommendation applications like ours, it is more sensible to convert waveforms into **spectrograms** before training. Spectrograms (or time-frequency representations) are 2D arrays that represent the evolution of the frequency decomposition of the signal across time. A common way to compute a spectrogram is via the Short Time Fourier Transform (STFT), which consists in a concatenation of the Fourier transform of short overlapping frames, as shown on Figure 2.2. In Chapter 4, we will build our music similarity system on a Constant-Q transform (CQT) representation. The CQT (Brown and Puckette, 1992) is an enhanced version of the STFT, where the frames have variable duration to ensure optimal resolution for each frequency band. More detailed information about the usual transforms for MIR can be found in the Fundamentals of Music Processing (Müller, 2015).

Images are generally represented as 2D or 3D tensors, of which each coefficient represents a pixel for display. From 2D time-frequency representations or images, it is again possible to extract more compact, higher-level features. Such characteristics include (but are not restricted to) edges and simple shapes for images; rhythmic and harmonic patterns for music. Their benefits are common to a variety of MIR and computer vision tasks. Learning feature extraction operations along with each recommendation system would be quite inefficient in terms of time and energy. It is therefore common practice to train a neural network f_θ specifically to extract high-level features. The corresponding task is called **representation learning** and can be unsupervised, self-supervised or supervised. Representation learning allows to leverage intermediate representations learned from large, and sometimes private datasets before moving to more specific training data. We will use representation learning extensively throughout the thesis to preprocess audio and video data in Chapters 5 and 7. Pre-trained networks f_θ with frozen weights θ are there used

offline as a sophisticated feature extractor for our specialized, shallow recommendation networks.

Fully-connected networks (FCNs). Fully-connected networks are adapted for input data presented in the form of n-dimensional vectors $x = (x_1, ..., x_n)$. The output of each unit consists in a linear combination of the input by the weights $w = (w_1, ..., w_n)$ to which a **bias** b is added and a non-linear function g is applied:

$$y = g(w^T x + b). \tag{2.3}$$

Units are then grouped into layers, in which each unit takes as input all outputs of the previous layer (see Figure 2.1). The collection of weights w and biases b of all layers form the parameters θ that are learned during training. We use Fully-connected networks in Chapters 5 and 7 to perform video/music matching from high-level features.

Convolutional Neural Networks (CNNs). Convolutional networks (Fukushima, 1980; LeCun and Bengio, 1995) are a special case of neural networks which showed excellent performances in image recognition and in MIR during the recent years (Cohen-Hadria, 2019; Stoller, 2020). They allow to discover patterns in images using the natural spatial correlation between the coefficients (adjacent pixels in natural images).

CNNs are suitable for input data shaped as tensors: $x = \{x_{m,n,k}\}_{m\in[\![1,M]\!],n\in[\![1,N]\!],k\in[\![1,K]\!]}$. In a convolutional layer, each unit processes the input tensor by convolving it with a filter, i.e. a small matrix of weights $w = \{w_{u,v}\}_{u\in[\![1,U]\!],v\in[\![1,V]\!]}$, adding a scalar bias b and then applying an activation function:

$$y = g(x \star w + b), \tag{2.4}$$

where:

$$\forall m, n \in [\![1, M]\!] \times [\![1, N]\!] \quad (x \star w)_{m,n} = \sum_{k=1}^{K} \sum_{u=1}^{U} \sum_{v=1}^{V} x_{m-u+1,n-v+1,k} w_{u,v}. \tag{2.5}$$

Each filter w convolved with the tensor x produces a **feature map** y of shape $\bar{M} \times \bar{N}$. In this thesis, we will focus on the case where $\bar{M} = M$ and $\bar{N} = N$, obtained with adequate padding of the input tensor. If the layer contains F units, then F feature maps are produced and stacked together (see Figure 2.3). This way, the shape of the output tensor is $M \times N \times F$. The coefficients of the filters w and biases b of all layers form the parameters θ that are learned during training.

CNNs used for classification or regression often use dimensionality reduction operations. To do so, convolution operations are interleaved with **pooling** operations. The idea of such architectures is to progressively transform the input data into a compact representation. In a pooling layer, the number of feature maps is preserved, while the size of each filter map is reduced (it is a spatial subsampling operation). The feature maps are partitioned into patches (the "pools") and only one value is kept for each patch. This way, the values that are too low or not significant in each patch are removed. Common pooling operations include max-pooling and average pooling.

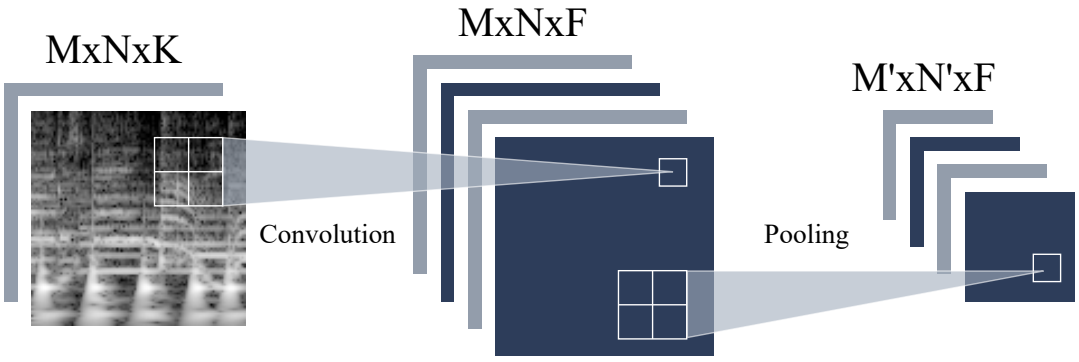


FIGURE 2.3: A convolutional neural network with one convolutional layer and one pooling layer. In the convolutional layer, each filter is slid over the input tensor and produces one activation per "pixel" location. The resulting feature map is stacked along the depth dimension with the ones produced by the other filters.

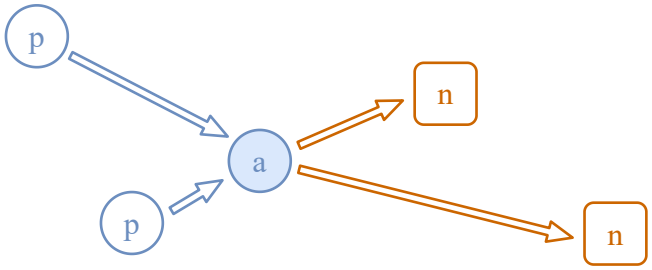


FIGURE 2.4: Using the triplet loss for metric learning. The distance between the embeddings of the anchor a and the positive samples p is minimized, while the distance between the embeddings of a and the negative sample n is maximized.

Although CNNs were initially designed for image processing, they are popular in MIR as well, since they can be used on spectrogram data. As opposed to natural images, the vertical (frequency) and horizontal (time) axes of a spectrogram have different roles. Nevertheless, CNNs have proved to be efficient on those as well. One hypothesis is that CNNs are able to create different categories of filters for processing low frequencies versus high frequencies. Convolution operations are robust to spatial translations and produce sparse representations, thus making training faster and reducing the risk of overfitting, even on high-dimensional input data (Mallat, 2020). These regularity properties allow stacking more layers than in fully-connected networks without loss of stability. **Deep** architectures progressively transform the input data into abstract, high-level representations, which are expected to make downstream tasks (such as recommendation) simple. In this manuscript, we use deep CNNs to learn music similarity in Chapter 4.

2.1.2 Metric learning

The task of retrieving items in a collection and sorting them by relevance arises in a variety of domains. The input document is usually called the **query**, while the relevant documents which are expected to appear at the top of the ranking are called **targets**. In the context of music supervision, the collection from which the target documents must be extracted will be often named **catalog**, due to the fact that the items are music tracks. In practice, it is common to first learn a projection such that the

relevance score of an object with respect to another can be computed as a distance, or metric, of both projections. This method is called **metric learning**. We denote by the term **embedding** the representation of the data after its non-linear projection in a high dimensional space. As we study recommendation tasks, the main objective of our metric learning systems is to learn how to rank music and video data according to predefined criteria (e.g., similarity, content adequation, structure matching).

A common approach is to learn a matching from classes. In this case, it is assumed that the similarity of items within the same class should be higher than the similarity of items from two different classes. As a result, the model's recommendations are associated with a binary relevance score: the recommended item does or does not belong to the expected class. Such problems have been studied for example by Weinberger and Saul, 2009 and Hoffer and Ailon, 2015, who employed CNNs with innovative loss functions to perform higher-accuracy image classification. Another option is to train siamese networks (Bromley et al., 1994) with a contrastive loss (Hadsell, Chopra, and LeCun, 2006), but this was later pointed out for being very sensitive to hyperparameter tuning (Manmatha et al., 2017).

Today, a widely used loss for metric learning is the **triplet loss**, as introduced by Schroff and Philbin, 2015 for face recognition (see Figure 2.4). In their work, the authors use the triplet loss to force the CNN to learn a projection such that the projections of images of the same person will be pulled together, and the ones from different persons will be pushed apart. More generally, the objective of a triplet loss system is to organize the projection f_θ of three samples in the embedding space such that the distance between an anchor a and a positive sample p is minimized, while the distance between the anchor a and a negative sample n is maximized:

$$\mathcal{L}(a, p, n) = \max(||f_\theta(a) - f_\theta(p)||_2^2 - ||f_\theta(a) - f_\theta(n)||_2^2 + \alpha, 0). \tag{2.6}$$

In this expression, the triplet (a, p, n) is created such that the positive is more similar to the anchor than the negative according to the ground truth criterion. $\alpha \in \mathbb{R}_+^*$ is a margin parameter that enforces a minimal distance between the positive and negative pairs. f_θ denotes the neural network embedding function. In this manuscript, we will use the squared Euclidean distance, although other differentiable metrics can be employed too (Hermans, Beyer, and Leibe, 2017). The advantage of triplet loss systems is their rapidity at retrieval time: producing recommendation only requires computing pairwise Euclidean distances and sorting them.

We use the metric learning paradigm and the triplet loss throughout this dissertation. In Chapter 4, we propose a method to learn music similarity from ranked lists. In Chapters 5 and 7, we propose a Video-Music recommendation system trained with triplet loss.

2.1.3 Audiovisual structure estimation

To accurately recommend music and video, it is appropriate to take into account their temporal structure, and thus to estimate this said structure. Music structure estimation is a hierarchical problem that goes from beat tracking (regular accentuation of musical phrases) to boundaries detection (between choruses, verses and bridges). Similar to music, videos can be divided into segments of various duration, from shots to scenes to chapters and longer sequences. We will describe here the methods which we used in this manuscript.

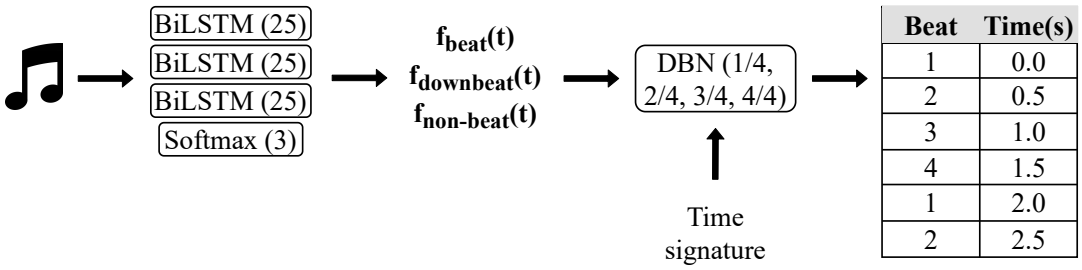


FIGURE 2.5: Block diagram of the downbeat tracking method implemented in Madmom. In this example, the provided time signature is 4/4, hence the DBN has four states corresponding to each beat of the bar. For a tempo of 120 bpm, beats occur every 500 ms.

Downbeats are systematically accentuated beats that happen periodically within musical phrases (every three or four beats in most Western genres). Downbeat tracking is a popular MIR task (Jia, Lv, and Liu, 2019). As a result, several ready-to-use libraries are available to estimate downbeat positions from audio files (Böck et al., 2016; Bogdanov et al., 2013). Early approaches used heuristics to estimate a downbeat likelihood function. For example, Peeters and Papadopoulos, 2011 computed a distance between a beat template and harmonic features, before extracting the downbeat sequence via a linear discriminant analysis. Machine learning methods typically rely on Bayesian modeling to recognize rhythm patterns in data Krebs, Böck, and Widmer, 2013, or on Recurrent Neural Networks which are suited to process sequential data Vogl et al., 2017. The state-of-the-art **Madmom** method of Böck, Krebs, and Widmer, 2016 combines both ideas in a two-step algorithm (see Figure 2.5). First, a set of supervised recurrent neural networks estimates jointly the beat and downbeat activation functions. The output of the neural networks represents the probability of each frame being a beat or downbeat position. These activation functions are then fed as observations to an unsupervised Dynamic Bayesian Network (DBN). The DBN outputs the beat positions of highest likelihood, along with their position inside the bar. We will use Madmom in our study on music-video clips (Chapter 6).

At a larger timescale, the automatic detection of boundaries between functional segments (choruses, verses and bridges), keeps receiving a lot of attention from the MIR community (Nieto et al., 2020). In this respect, specialized and open-source libraries were developed for this task too. The MSAF library (Nieto and Bello, 2016) implements most state-of-the-art algorithms. We will present three of them, which approach the problem from very different angles, and which we will use for our study. Pioneer work by **Foote, 2000** started using self-similarity matrices of low-level audio features to detect novelty in the signal. Foote's unsupervised method detects points of significant change by applying a checkerboard kernel along the diagonal of the self-similarity matrix (see Figure 2.6). This convolution results in a temporal function of audio novelty, on which peak detection is performed to obtain the boundaries. The segments are thus defined as homogeneous units on either part of their boundaries. It is worth mentioning that Foote's method can be enhanced by being applied to learned features, rather than low-level audio features (McCallum, 2019), although we will stick to the basic version for practical reasons. The unsupervised **Structural Features (SF)** algorithm introduced by Serra et al., 2012, on the other hand, emphasizes repetition as a key characteristic of functional segments. The time series structure features are computed using a circular time-lag matrix of

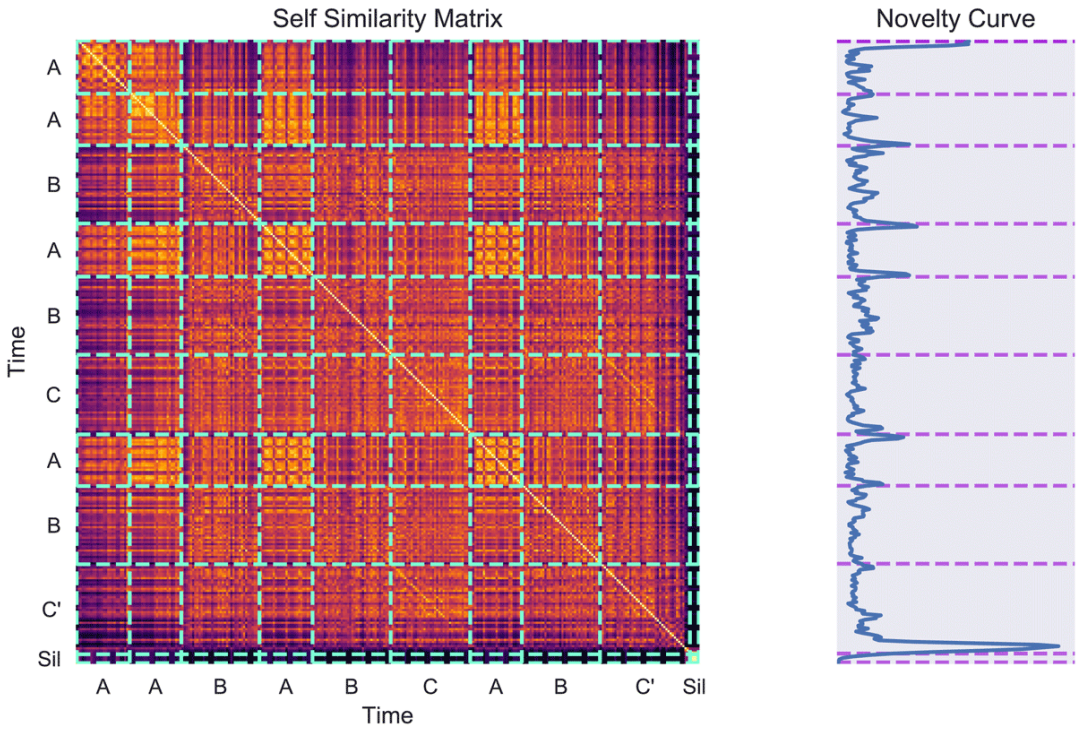


FIGURE 2.6: Detecting music segment boundaries from a self-similarity matrix (left). Peaks in the associated novelty curve (right) indicate potential boundaries. Figure taken from Nieto et al., 2020.

homogeneity and recurrence features. Then, similar to Goto, 2006, the novelty curve is obtained by computing the Euclidean distance between two successive columns of the lag matrix. Finally, the **Ordinal Linear Discriminant Analysis (OLDA)** algorithm by McFee and Ellis, 2014 relies on supervised learning. In this method, an array of handcrafted, beat-synchronous features X is linearly transformed via a learned matrix W. The frames of the resulting matrix $W^T X$ are finally clustered to obtain the segments. The total number of segments is chosen by AIC cost minimization (Akaike, 1992). "Ordinal" in this case designates a relaxation of the linear discriminant analysis projection that only attempts to separate adjacent segments. Then, the obtained features are clustered with a temporal constraint: only similar successive segments are merged together.

Video structure is a whole research area with its own challenges and tasks, including video summary, video chaptering and more (see the TRECVID challenge, Smeaton, Over, and Kraaij, 2006). In this thesis, we focus on the least ambiguous type of segments: the shots. In music videos, there are several types of transitions between successive shots, such as cuts, but also camera movements, large objects passing in front of the camera, scenes observed through reflecting surfaces, etc (Schindler and Rauber, 2019). Most of these can be estimated by the **TransNet** (Souček, Moravec, and Lokoč, 2019). The TransNet system is a CNN which employs 3D ($3 \times 3 \times 3$) convolutions. Each convolution employs a specific dilatation rate, which increases its receptive field, and the outputs are concatenated along the channel axis. The dimensions are progressively reduced by max-pooling, and two fully-connected layers are added to perform the *cut vs no cut* classification. It was trained in a supervised way for a shot boundary detection task on the TRECVID IACC.3 dataset.

We use Madmom, OLDA and the TransNet in Chapter 6 to analyze official music

videos. We use the cuts estimated by the TransNet, and the music boundaries detected by Foote, SF and OLDA in Chapter 7 to recommend music tracks for videos based on segment alignment.

2.2 Systems for music supervision: A state of the art

Music Information Retrieval (MIR) is the interdisciplinary research domain dedicated to processing and analyzing music. Computer vision is a huge research area related to the processing and analysis of images. Both fields aim at understanding their respective media using concepts and methods from signal processing, machine learning, perception, cognition, and humanities. In this section, we will review the MIR and computer vision literature related to the music supervision task.

2.2.1 Music auto-tagging

In MIR, we call **music auto-tagging** the task of predicting tags directly from audio signals (see Figure 2.7). **Tags** are keywords that describe a music track in terms of genre, mood, instrumentation or any other high-level attributes. The set of valid tags in a given dataset, as well as their potential hierarchy, is referred to as the **taxonomy** of this dataset.

Auto-tagging typically uses supervised machine learning to train a system which predicts the tags given the audio content as input. Therefore, it necessitates annotated data. Traditional approaches for music auto-tagging rely on the extraction of handcrafted features (such as MFCCs) to feed a classifier (Eck et al., 2008; Costa, Oliveira, and Silla, 2017; Tingle, Kim, and Turnbull, 2010). Recently, deep learning approaches have allowed to learn their own feature representation directly from the data, leading to improved performances. Both waveform (Pons et al., 2018; Kim, Lee, and Nam, 2018) and spectral representations (Choi, Fazekas, and Sandler, 2016a; Pons et al., 2017; Ferraro et al., 2021; Kim et al., 2020) were successfully used. Some of these systems have proven their capacity to learn useful information from audio (Dieleman and Schrauwen, 2014; Choi, Fazekas, and Sandler, 2016b).

In Chapter 4, we employ an auto-tagging system as a baseline for our music similarity task.

2.2.2 Music similarity

Music similarity is the task of retrieving music tracks similar to a reference track from a music catalog. Although music similarity is not a universal notion, it is a popular task with numerous applications such as music recommendation (Bozzon et al., 2008), playlist creation (Lin and Jayarathna, 2018) or cover identification (Doras and

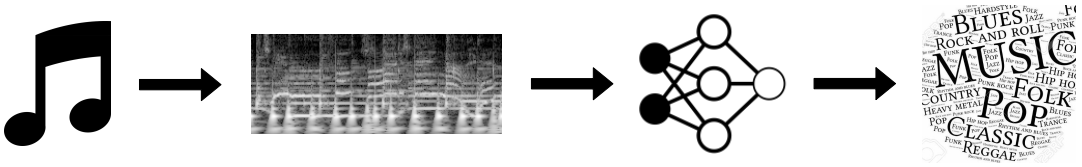


FIGURE 2.7: A typical music auto-tagging system. A neural network is trained in a supervised way to predict tags from time-frequency audio representations.Tag could illustration: alamy.com.

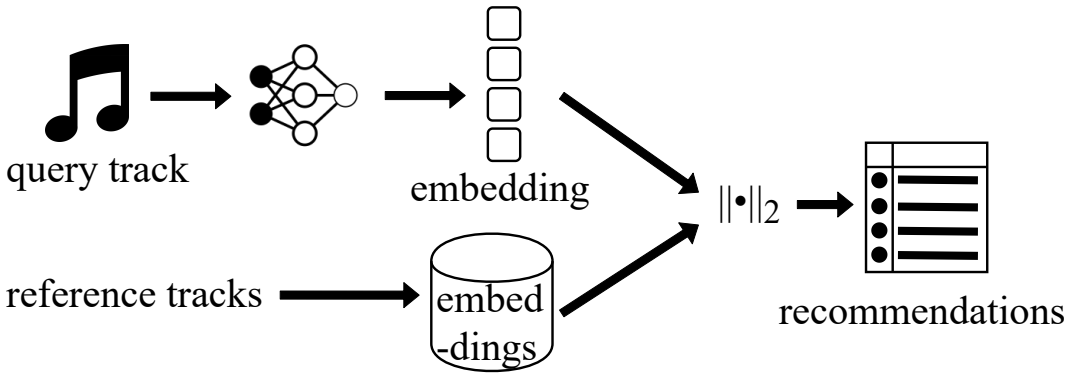


FIGURE 2.8: Recommending music tracks by similarity with respect to a query track.

Peeters, 2019). Music similarity is usually addressed by computing a pairwise similarity score between the reference (query) track and each candidate, before ranking those by decreasing score. The similarity score can be computed based on tags (annotated manually or by an auto-tagger), or directly from the audio content (scenario illustrated on Figure 2.8).

Fully unsupervised methods include the one of Charbuillet et al., 2011, who employ GMM supervectors to represent complex statistical models by Euclidean vectors. Using text labels for supervision, Slaney, Weinberger, and White, 2008 propose a set of linear transforms to embed tracks into a Euclidean metric space and evaluate them on a nearest neighbor task for album, artist and blog recognition.

On the supervised learning side, McFee, Barrington, and Lanckriet, 2012 use collaborative filtering data to train the *metric learning to rank* algorithm (Mcfee and Lanckriet, 2010). This algorithm consists of a linear projection trained with a listwise loss function to learn text document retrieval from ranked lists. Other text metadata can also be used, such as release date, artist biography, or web-crawled critics about the album. In this case, text retrieval models can be applied, such as *tf-idf* and *Word2Vec* (Cunha, Caldeira, and Fujii, 2017), or path scoring, if the information can be represented in a knowledge graph (Gabbolini and Bridge, 2021). Wolff and Weyde, 2014 insist on modeling *relative* music similarity (i.e., rankings), rather than absolute similarity ratings, in order to avoid consistency issues due to subjective user ratings. Following this idea, Lu et al., 2017 employ a CNN to predict relative music similarity. Once trained, CNN-based methods are considerably more efficient than their predecessors.

In Chapter 4, we take inspiration from Lu et al., 2017 but we propose a mining strategy from ranked lists instead of partial similarity annotations.

2.2.3 Video/Music matching by semantic association

In this part, we review works related to V/M matching, i.e. the matching of video and music modalities. Music and video are two expressive data **modalities**, which are commonly associated using widespread cultural codes. Schindler and Rauber, 2016 analyze the correlation between visual contents (objects present in the scene) and music genre in official music videos. For example, cowboy hats are almost systematic in country music videos. Liem, Larson, and Hanjalic, 2013 demonstrated in their user study that V/M matching could reliably be performed by users via textual descriptions.

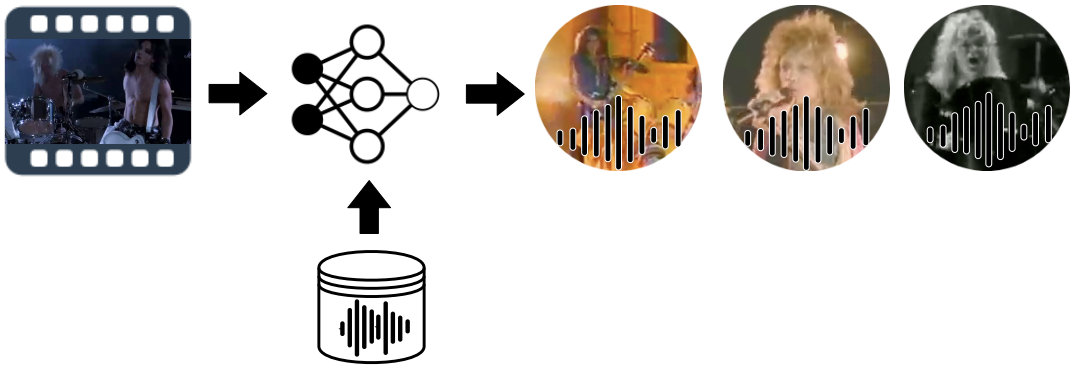


FIGURE 2.9: A Video/Music matching system: given a query video and a catalog of music tracks, retrieve matching music tracks.

In this manuscript, we denote by the term **Audio/Video (AV) clip** a matching data pair c made of the silent video track c^v and the audio music track c^m extracted from the same **media** (document), hence synchronized in time: $c = (c^v, c^m)$. Previous works have shown that it is possible to learn a matching between music and video files, in a supervised or self-supervised way (see Figure 2.9). The idea is to associate each element of the pair to a single representation in a joint multimodal embedding space.

In general, music-video embeddings are computed via a pair of projection functions (f_m, f_v). f_m takes as input a music audio excerpt x^m and outputs an embedding vector $e^m = f_m(x^m)$. f_v takes as input a video sample x^v and outputs an embedding vector $e^v = f_v(x^v)$. Both vectors belong to the same high-dimensional embedding space. This allows to perform cross-modal retrieval or recommendation, for example by computing Euclidean distances between two embedding vectors: $d_{a,b} = ||f_m(x_a^m) - f_v(x_b^v)||^2$. V/M recommendation is a wide domain which has applications in automatic video editing (Shah, Yu, and Zimmermann, 2014), automatic MTV generation (Liao, Wang, and Zhang, 2009), and general music recommendation (Zeng, Yu, and Oyama, 2018).

f_m and f_v can be trained jointly to associate music and video samples according to some matching criteria. In a broader sense, this also includes systems designed for any type of sound or images (single frame videos). A popular design choice is to implement f_m and f_v as neural networks. Those can be trained with the help of another network that performs a fusion between both modalities, or using a specific loss (e.g., the triplet loss), that directly organizes the embedding vectors produced by f_m and f_v. The training of such embeddings can be done using two paradigms: supervised or self-supervised.

In the case of supervised learning, the matching criterion that associates the audio and video modalities is deduced from additional sources of information. Shah, Yu, and Zimmermann, 2014 used Support Vector Machines to estimate mood tags from both modalities and then perform the final matching decision. Another interesting method is to project each sample directly into the valence-arousal plane via a rule-based model (Sasaki et al., 2015) or a linear regression model (Shin and Lee, 2017). Later, Zeng, Yu, and Oyama, 2018 have developed a Deep Canonical Correlation Analysis model with mood supervision for cross-modal retrieval. While Hsia et al., 2018 trained a CNN to learn a semantic association of image and music via keywords extracted from the lyrics, Surís et al., 2018 proposed to train a fully-connected network for the audiovisual correspondence objective, regularized by a

video classification task. Tian et al., 2018 trained a dual multimodal residual network with audio-guided visual attention for an audio-visual event localization application. More recently, Li and Kumar, 2019 trained a CNN for two emotion classification tasks, one for each modality, along with the cross-modal distance learning task. Shang et al. Shang et al., 2021 exploited semantic and emotion labels from lyrics to learn a connotation-aware association of music and image. The use of mood information accelerates the training and allows the systems to reach promising retrieval performances. The matching criteria can be expressed as a score that is not necessarily binary, allowing nuances. However, this restricts the systems to learning a certain type of audiovisual correspondence, and requires to collect additional information to train. Another challenging aspect of this approach is that both modalities must have previously been associated to a shared mood taxonomy (Won et al., 2021).

In the case of self-supervised learning, a binary matching criterion is used. Several learning objectives can be used, such as translating two matching modalities into each other (Vukotic, Raymond, and Gravier, 2018). In V/M matching, the usual learning objective is to classify pairs of music and video samples between the *matches* (i.e. both samples are extracted from the same clip and from the same time position) and *non-matches*. Depending on the pretext task, music-video embeddings can be reused as feature vectors for downstream tasks (e.g., audio/image classification) or directly to perform V/M matching in the context of cross-modal recommendation.

Early approaches have learned correlations between audio and video modalities (Kuo, Shan, and Lee, 2013) using multiple-type latent semantics analysis (Su et al., 2013) or dual-wing harmonium models (Liao, Wang, and Zhang, 2009). Later, methods based on deep neural networks were proposed. With an appropriate loss function, neural networks can learn directly the matching between music and video. Aytar, Vondrick, and Torralba, 2016 first experimented with a KL-divergence loss between the audio and video embedding vectors, in the context of acoustic scene and object classification. Alternatively, the L3-Net proposed by Arandjelovic and Zisserman, 2017 stacks a fully-connected fusion network after the two-branch embedding extraction network. The whole system is trained with a binary cross-entropy loss, to predict whether both samples were extracted from the same video clip. Initially introduced for environmental sound classification, the L3-Net became a popular baseline system for the audiovisual correspondence task and pre-trained audio and video embeddings. Numerous variants were proposed, such as adding a per-location matching score (Arandjelović and Zisserman, 2018), a CCA in the loss function (Yu et al., 2019) or a distribution matching stage (Hu et al., 2021) to perform sounding object localization. Source separation can also be achieved with a curriculum learning strategy (Hu et al., 2020). In Chapter 5, we show that OpenL3, the open-source version of the L3-Net (Cramer et al., 2019) performs well when reused as a feature extractor for learning the Video/Music recommendation task.

The VM-Net (Hong, Im, and Yang, 2018) is a music-video embedding system developed specifically for cross-modal recommendation and retrieval (music recommendation given video as input or video recommendation given music as input). Inspired by the L3-Net, it is trained on an audiovisual correspondence task. Its architecture is kept simple, which limits the computational cost of training on large music-video datasets. In Chapters 5 and 7, we use the VM-Net for the V/M recommendation task.

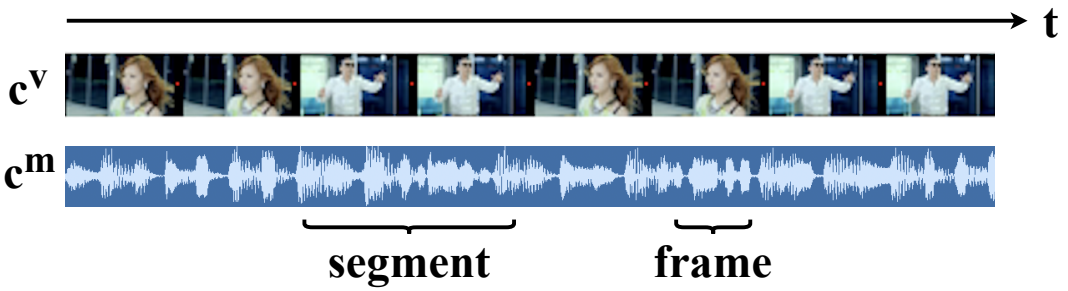


FIGURE 2.10: Illustration of an AV clip and the different timescales associated: full clip, segment and frame.

2.2.4 Video/Music matching by synchronization

All previous systems associate each modality (or part thereof) to a single timeless vector. One embedding vector can represent either the full AV clip, a segment thereof, or a short frame (typically an image and the associated 40 ms of audio, see Figure 2.10 for an illustration). To perform the recommendation, it is also possible to exploit the temporal relationship between music and video.

Video/Music matching by sequence comparison. Several studies suggest to compute alignment metrics between each music-video pair to rank the candidates for the recommendation. Early work by Gillet, Essid, and Richard, 2007 performed the cross-modal recommendation task using a correlation metric between sequences of frame-level acoustic and visual features. This system relies on the synchronization of music structure (onsets and functional segments) and video structure (motion intensity and cuts) to perform music-video recommendation, without external data. Ruiduo Yang and Brown, 2004 and Mulhem et al., 2003 proposed similar approaches. This work is particularly interesting in the context of music supervision, as it allows a frame-level temporal alignment between the music and video. However, the ranking of music tracks only relies on the variations of a low-level scalar descriptor across time. Consequently, the semantic representation of each modality is reduced to this single descriptor. In the context of source association, Li et al., 2019b proposed to align frame-level sequences of higher dimensional audio and visual features using the Hungarian algorithm. In this case too, the considered descriptors are not trained specifically for the task. Conversely, the pretext task of the self-supervised system by Owens and Efros, 2018 is to predict whether an audio excerpt and a video excerpt extracted from the same AV clip were temporally shifted or not. The performances of the system are illustrated by audio-visual action recognition and on-screen (speaker visible on the screen) vs off-screen (not visible) audio source separation. Wang, Fang, and Zhao, 2020 also train an end-to-end, frame-level model with attention and time warping for dance-music alignment and speech-lip alignment. However, this is only meant to align two preselected media.

To take into account the temporal evolution of each modality for recommendation, another option is to model those as a sequence of segments. For example, Lin, Wei, and Wang, 2015; Wang et al., 2012 and Lin et al., 2017 use correspondence of emotion sequences to compute an alignment cost between a music track and a video. In these cases, the matching metric is learned a posteriori, as the Euclidean distance between the considered representations cannot be directly interpreted as a matching indicator. In general, segments-based approaches use fixed-length excerpts that do not represent semantically meaningful units of the AV clips. Cheng and Shen Cheng

and Shen, 2016 represented songs and venues as sequences of concepts for context-aware music recommendation. Gabeur et al. Gabeur et al., 2020 tackle the caption-to-video retrieval problem using a multi-modal Transformer. Second-level similarity scores are then aggregated via a learned weighted sum. Only Shin and Lee, 2017 use a semantic segmentation into semantically homogeneous units.

Video/Music matching by leveraging music structure. Compared to general audio signals, music has a specific structure. Music tracks can be divided into introductions, choruses, verses, and other semantically meaningful units, or segments. Several studies (Wang et al., 2007; Wang, Chng, and Xu, 2006; Hua, Lu, and Zhang, 2004) use music structure estimation, paired with alignment techniques, to perform *music video generation*. This means that both the video and the music are preselected, and the system only performs the editing and synchronization of one of the modalities with the other. However, there is little work about exploiting this type of variable-length segmentation for recommendation.

In Chapter 7, we take inspiration from the music video generation techniques of structure-aware music segmentation and alignment of sequences. But rather than using them for music video generation, we use them for music recommendation.

2.3 Conclusion

From the technical point of view, we will perform our study using machine learning tools such as convolutional and fully-connected networks, supervised and self-supervised learning, and metric learning. We will get inspiration from the existing literature in music auto-tagging, music similarity and Video/Music matching. Finally, we will employ existing algorithms for music and video structure estimation.

In the following chapters, we will study several music recommendation systems. The music similarity task will be addressed in a supervised learning setting. The Video/Music matching models will employ self-supervised learning. All the presented systems are trained on a metric learning objective with a triplet loss. In all experiments, the training and evaluation of the machine learning systems require data. We develop this aspect in the following chapter.

Chapter 3

Datasets

In this dissertation, we focus on data-driven methods (supervised and self-supervised) to learn the similarity of music tracks and the matching of music and video. Since these systems rely on examples, they require data for both training and evaluating their performance. This chapter is dedicated to our research of suitable datasets.

The inference capacities of machine learning algorithms are directly impacted by the choice of the training dataset. For optimal performance, the training dataset should be as representative as possible of any situation that can be encountered during the inference task. Noise in the training dataset can also reduce the model's effectiveness. This is especially true in the self-supervised learning scenario, where the target outputs are generated automatically, but incorrect data can also happen in supervised training situations. Large datasets tend to mitigate this problem, as well as reducing the risk of overfitting. The desired properties of training datasets are therefore their diversity, size, quality and accessibility.

The evaluation of recommendation systems is a wide topic with dedicated conferences such as CLEF or FIRE. Evaluation protocols generally vary depending on the type of object to be recommended (text documents, music tracks, products). Music recommendation is particularly subjective and thus difficult to evaluate. In order to evaluate the relevance of candidate recommendations, high-quality datasets are needed. Other desired properties of evaluation datasets are diversity (in terms of music genres, epochs, artists, language) and accessibility. In this work, we favor cross-datasets evaluation when possible (training and test samples are taken from different datasets). This is a common good practice, whose aim is to prevent overfitting on biases inherent to any dataset.

Chapter organization. In Section 3.1, we review existing music similarity datasets and we detail the creation of the Creaminal dataset. In Section 3.2, we review state-of-the art datasets for Video/Music matching.

3.1 Datasets for music similarity

In this part, we describe the work we did about the selection of datasets for music similarity. We then describe the curation process that resulted in the creation of the Creaminal dataset.

3.1.1 Related work

As we saw in Chapter 2, music similarity can be defined in various ways (from tags, musicological expertise, subjective user ratings, etc.). As a consequence, several datasets were proposed, each of them using its own criteria. We provide here a

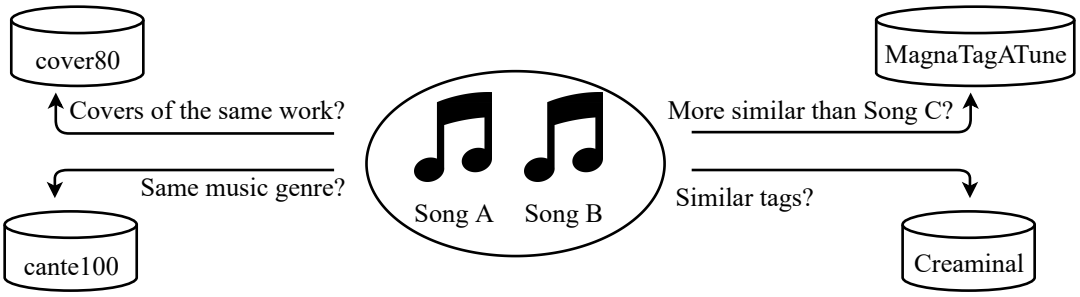


FIGURE 3.1: Illustration of how different datasets define the *music similarity* concept.

couple of examples (see Figure 3.1 for an overview). A more extensive list of MIR datasets is regularly maintained on the website *Audio Content Analysis*.

One way of defining music similarity is via cover identification. When two songs are covers from each other, they share a compositional similarity. The **cover80** dataset (Ellis, 2007) includes 80 songs, each played by two different artists, with noticeable differences in terms of harmony and style. Its main limitations are its small dimensions and bias toward pop music. Similar datasets include the **SHS100K** by Doras and Peeters, 2019 and **Da-TACOS** by Yesiler, Serrà, and Gómez, 2020, for which the audio is unfortunately not available.

Another interesting example of similarity annotations is using musicological knowledge of certain music genres. The **cante100** from the COFLA corpus (Kroher et al., 2016) comprises 10 songs for each of 10 families of flamenco music. However, the audio files are not publicly available and the dataset is de facto constrained to one specific genre.

Finally, the **MagnaTagATune** dataset was introduced by Law et al., 2009 for multilabel tagging. It features tag annotations, similarity annotations and audio [1]. 25,863 clips of 29 seconds each were extracted from 5,405 different tracks by 230 artists and encoded in 16 kHz, 32 kbps, mono MP3. Similarity data was collected from the TagATune video game. Non-expert players produced subjective judgments such as: "given a triplet of songs (A, B, C), song A/B/C is most different from the others". The main limitation of the MagnaTagATune dataset for similarity learning is its size: 7,650 judgments resulting in 533 similarity triplets.

We used the MagnaTagATune dataset for preliminary experiments only, and we did not include the results in this dissertation.

3.1.2 Creaminal music catalog

To train and evaluate our music similarity system, we chose to use an extract from the catalog of **Creaminal**, a music supervision company. The two main advantages of this original dataset are its size and quality.

Each track was professionally annotated with a number of tags comprised between 5 and 35, the average number of tags per track being 16.8. We selected the $N = 14,246$ tracks that were annotated with at least 5 key tags between June 2013 and November 2018, and which duration was comprised between 45s and 5 minutes. This filtering allows a good homogeneity in the tagging process. The corresponding

[1] Statistics about the MagnaTagATune annotations can be found at https://musicmachinery.com/2009/04/01/magnatagatune-a-new-research-data-set-for-mir/ and https://github.com/keunwoochoi/magnatagatune-list.

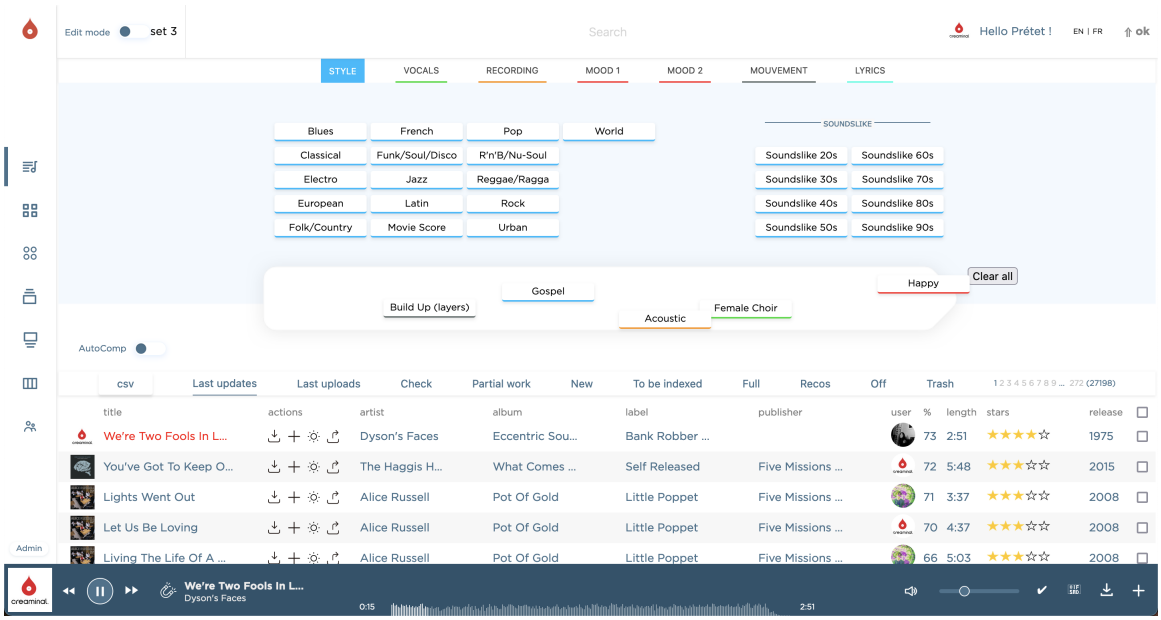


FIGURE 3.2: Screenshot of Creaminal's backoffice, a software designed specifically to annotate tracks and organize the music supervision work. For each song, relevant tags are placed on an arrow which symbolizes their relative importance (left means low, right means high).

TABLE 3.1: Some statistics describing the Creaminal dataset.

Number of different tracks	14,246
Number of different artists	3,565
Number of different tags	488
Sampling rate	44100 Hz
Encoding	32kbps Stereo MP3
Average track duration	3'44" (see Figure A.2)
Total number of tags attributed	250,349
Average number of tags per track	17.57 (see Figure A.3)

taxonomy is made of $m = 488$ tags, organized in 5 categories: Genre (e.g. Blues, Reggae, Electro-funk, Japanese Pop), Recording (e.g. Acoustic, Saturated, Guitar bass), Mood (e.g. Epic, Dancing, Nostalgic), Movement (e.g. Acceleration, Repetitive), and Lyrics (e.g. Death, Freedom, Nature). Figure 3.2 shows an example of the web interface used for tagging the tracks. The dataset features a majority of Pop tracks, along with Electro, Rock, Country and Movie soundtracks. More information and statistics about the tracks are available in Table 3.1. Histograms of tracks duration, number of tags and genre are available in Appendix A.3.

In addition to the audio files and track-level tag annotations, the dataset includes a specific tag-based similarity measure $\mathcal{S}$. $\mathcal{S}$ will be used as ground truth in Chapter 4 to define the notion of music similarity to be learned. The main limitation of the Creaminal dataset is that the music tracks and their manual annotations are proprietary and cannot be shared.

TABLE 3.2: Specialized MIR datasets for V/M matching.

Dataset name	Type of video	Intended task	Dimensions
AIST (Shuhei Tsuchida et al., 2019)	Street dance	Dance analysis	63 hours
DEAP (Koelstra et al., 2012)	Face videos	Emotion analysis	2 hours
Piano Gestures (Sarasúa et al., 2017)	Piano performance	Performance analysis	210 lieder
URMP (Li et al., 2019a)	Chamber music performance	Performance analysis	1.3 hours
AudioSet (Gemmeke et al., 2017)	All	Audio event detection	5,800 hours
MVD (Schindler, 2019)	Official music videos	Music classification	2,212 songs

3.2 Datasets for Video/Music matching

In this part, we propose an overview of the Video/Music (V/M) matching datasets used in this dissertation.

Historically, V/M matching is often performed in the context of specific tasks such as performance analysis or dance analysis. This results in datasets that are specialized in one single type of video. Table 3.2 summarizes common MIR datasets that feature video.

A more diverse dataset is **AudioSet**, as proposed by Gemmeke et al., 2017. The dimensions of AudioSet are unique: 2,084,320 clips were manually labeled into 527 categories. However, in the context of V/M matching, the data would first require filtering, as they do not necessarily contain music. We did not consider the AudioSet dataset for our study, because the duration of the clips (10 seconds) does not allow a deep analysis of the temporal organization of the clips.

3.2.1 HIMV-50K

Recently, Hong, Im, and Yang, 2018 proposed the **HIMV-200K** dataset, a large collection of musical videos for self-supervised audiovisual correspondence training. This set corresponds to the part of AV clips of the YouTube-8M dataset (Abu-El-Haija et al., 2016) annotated as "music video". It consists of 205,000 video-music pairs without annotations. It should be noted that while the audio of these clips always contains music, the video can be anything from professional promotional clips to amateur montages of still images (see Figure 3.3).

The size of the HIMV-200K dataset makes it suitable for training V/M recommendation systems. Its main limitation is the heterogeneous quality of the videos. From the list provided by the authors[2], we fetched all relevant YouTube IDs to recreate the dataset. However, due to country-specific restrictions, dead links and storage limitations, we only have access to 51,000 video-music pairs. We denote the resulting dataset as **HIMV-50K**. The average duration of each AV clip in HIMV-50K is 3 minutes 54 seconds with a standard deviation of 1 minute 2 seconds. In total, the size of the dataset is 988 Gb. Once downloaded, we stored it on an SSD RAID 5 system of capacity 1.8 Tb.

We use HIMV-50K in Chapters 5 and 7 to train our Video/Music recommendation systems. We partition the dataset by leaving out 1,000 (randomly selected) pairs for validation, and we use the remaining 50,000 pairs for training.

[2] https://github.com/csehong/VM-Net/blob/master/data/

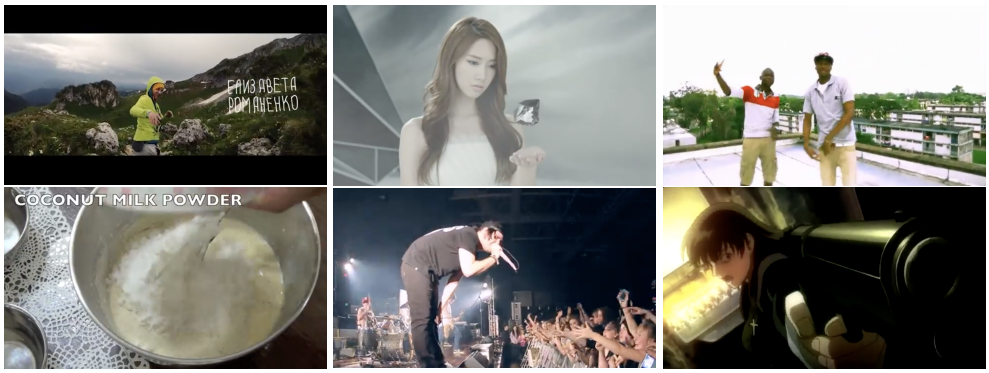


FIGURE 3.3: Randomly selected clips from the HIMV-50K dataset. We selected one representative frame to illustrate each video/music pair.

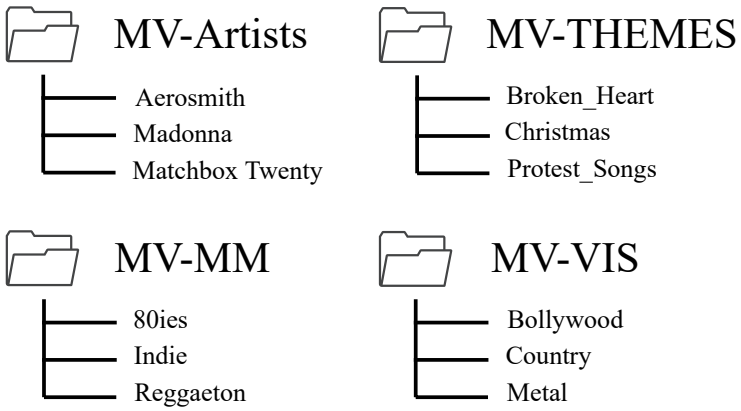


FIGURE 3.4: Taxonomy of the clips proposed in the MVD. For each of the four main categories, we include three examples of subcategories. All tracks are sorted into subcategories, which means that each track is annotated by one single tag corresponding to its subcategory.

3.2.2 Music-Video Dataset

While the HIMV-50K dataset is large and diverse therefore suitable for training a neural network, the quality of the video part of the AV clips is uneven. Evaluating Video/Music recommendation systems on this noisy dataset would make results difficult to interpret. Therefore, to evaluate the performance of our systems, we use a cleaner dataset, the Music Video Dataset (Schindler and Rauber, 2015; Schindler and Rauber, 2016).

The **Music Video Dataset** (MVD) is a valuable resource for V/M matching, as it is composed exclusively of manually curated official music videos. (Schindler, 2019) gathered it in order to perform an in-depth analysis of this specific media. As a result, all music tracks and videos are of professional quality, a unique characteristic. The MVD also features partial manual annotations into music genres, themes, and artists (see Figure 3.4). The average duration of each AV clip is 4 minutes with a standard deviation of 57 seconds. Based on frame rate homogeneity, we selected $C = 1,000$ of these clips. We use the filtered MVD in Chapters 5 and 7 to evaluate our V/M recommendation systems. We do not train any Video/Music recommendation system on the Music Video Dataset because of its relatively small size (2,212 music video clips).

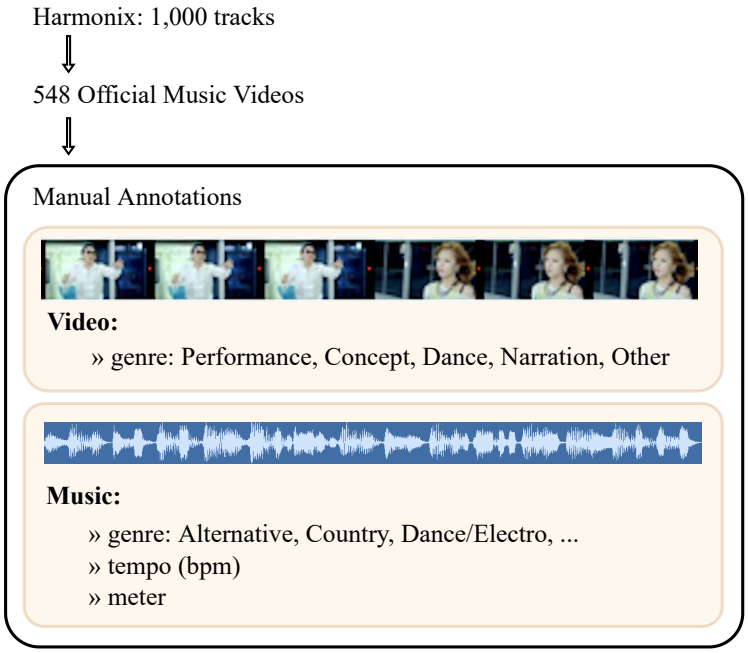


FIGURE 3.5: An overview of annotations of the Harmonix set which can be used for V/M matching.

3.2.3 Harmonix set

To perform accurate V/M recommendation, it is crucial to consider not only the content but also the temporal structure of both media. In addition to the music and video content, annotations into structural elements are valuable data for studies about V/M matching.

Harmonix (Nieto et al., 2019) is a recent dataset for automatic estimation of beat, downbeat and functional music segments. It features popular Western music tracks for which there is a high probability of having an associated music video, and YouTube URLs for each track are provided. From the list of 1,000 YouTube video links, 899 were successfully retrieved, of which 40% contained only still images and 2.4% were duplicates. We manually annotated each of the remaining 548 Official Music Videos (OMVs) into video genres. We provide the list and URLs of the OMVs as well as the genre annotations [3]. We also used the music genre annotations and the tempo annotations for our analysis.

We use the Harmonix set in Chapter 6 to analyze official music videos. We examine the visual and audio structure and their temporal correlation. We summarized the metadata we used for our study on Figure 3.5.

3.3 Summary

In this chapter, we introduced the four datasets used in the following of the manuscript: Creaminal dataset, HIMV-50K, MVD and Harmonix. We justified the choice of these particular datasets in light of the related literature. We sum up here our two main contributions:

[3] Our list is accessible at: https://gitlab.com/creaminal/publications/ismir-2021-language-of-clips/-/blob/master/video_genres.csv.

1. We proposed a method for creating a music similarity training dataset from ranked lists (Creaminal dataset).

2. We introduced a taxonomy of video genres and the manual annotations of the Harmonix set.

Chapter 4

Music Similarity: Learning to Rank Music Tracks

Music supervision, like music streaming, makes use of large music catalogs. To organize these tracks, it is necessary to provide efficient retrieval mechanisms. While browsing by tags (genre, mood, instrumentation) can be efficient for small-scale catalogs, it does not provide an efficient retrieval mechanism at scale. This is why it is sensible to proceed by music recommendation. Recommendation systems are able to retrieve a ranked list of music tracks based, for example, on their similarity with a query music track. In this chapter, we address the problem of ranking a music dataset by similarity with a query track.

For a given project, music supervision experts are often required to find a track similar to a reference track but with other desirable attributes such as a free license or a specific artist. Provided a similarity metric is defined, the music recommendation problem reduces to a music similarity problem. This task has been the subject of many publications (see Urbano, 2013 for an overview) and can be tackled in different ways. In **collaborative filtering recommendation**, two music tracks are considered similar if they are listened to by the same audience (Oord, Dieleman, and Schrauwen, 2013). Of course, if no one has ever listened to a music track (for example a new track in the catalog), it cannot be recommended. This is known as the cold start problem (Schein et al., 2002). However, when applicable, collaborative filtering has proved to give very good results for recommendation. In **tag-based recommendation**, a tag-based similarity measure is designed to compare music tracks based on their respective tags. Tags can be manually annotated (such as in Pandora - Clifford, 2007), crowd-sourced (such as in Last.fm), or automatically inferred from the audio content (auto-tagging case - Choi, Fazekas, and Sandler, 2016a). In **direct recommendation**, it is possible to compute directly a distance between two music tracks based on their audio content. For example, in one of the pioneer works of Aucouturier and Pachet, 2002, track MFCCs were represented by a Gaussian Mixture Model, and a Kullback-Leibler divergence was used to compare two tracks. But because this method uses a costly pairwise comparison function, it cannot scale to large catalogs.

In this thesis, we introduce the MuSimNet, a system for **direct recommendation** based on the audio content. For its implementation, we study the case of the Creaminal dataset, a large music catalog professionally annotated with tags. An associated function $\mathcal{S}$ was designed to compute a similarity score between two sets of tags. For a given query track, $\mathcal{S}$ allows us to retrieve similar tracks based on their tags. Our goal is to reproduce the similarity ranking given by $\mathcal{S}$ *without explicitly tagging the tracks*. We denote our approximation by $\hat{\mathcal{S}}$. For a given query track, $\hat{\mathcal{S}}$ allows us to retrieve similar tracks based on audio directly. This allows to skip the long and expensive manual tagging step. In this manuscript, we formulate the music ranking

problem as a nearest neighbor search problem in a d-dimensional Euclidean space. We train the MuSimNet, a CNN with metric learning to define a projection of the audio signal such that the proximity between two projected music tracks accounts for their similarity $\mathcal{S}$. We compare our direct recommendation approach to the one obtained by a CNN trained to estimate automatically the related tags which are then used in $\mathcal{S}$.

Chapter contributions:

1. We propose several strategies to perform metric learning from ranked lists. This allows to apply a triplet loss to the related music similarity problem.
2. We compare and validate these different triplet mining strategies on a large-scale experiment against the auto-tagging based approach.
3. We demonstrate the efficiency of the recently proposed Auto-pooling layer (McFee, Salamon, and Bello, 2018) for a music task.

The MuSimNet and the associated methods for triplet mining from ranked lists were presented at the ICASSP conference (Prétet, Richard, and Peeters, 2020).

Chapter organization. In Section 4.1, we review works related to our specific problem. In Section 4.2, we describe our proposed method to mine triplets from ranked lists. In Section 4.3, we evaluate the proposed approach, along with an auto-tagger baseline, on a music retrieval task. In Section 4.4, we draw conclusions of our results.

Reproducibility: Although we cannot distribute the Creaminal dataset and its oracle similarity function, to allow reproducibility of our work we provide the architecture and experimental code [1].

4.1 Related work: Metric learning from ranked lists

As we saw in Chapter 2.1.2, most metric learning systems are trained on data already organized in classes. For the triplet loss, this facilitates the **triplet mining** process (design of the training triplets). The corresponding strategy is to mine positive samples from the same class as the anchor, while negative samples belong to a different class.

Yet, music description is inherently a multi-criteria problem (each track has its own genre, instrumentation, mood, and these directions can be independent). Basing a similarity metric on the adherence to one single criterion (or class) would result in perceptually poor results. Instead, we want our similarity metric to imitate $\mathcal{S}$, a multi-criteria tag-based function. Outside the Creaminal dataset, functions like $\mathcal{S}$ have been implemented as tag-based Jaccard similarities by Kaminskas and Ricci, 2011 to rank geographical places of interest from a music query. To mimic $\mathcal{S}$, we need to learn the projection from the reference ranked lists given by $\mathcal{S}$, rather than from classes. In this case, the triplet mining strategy is the main challenge.

Such a problem has already been addressed outside the music case, for example in face recognition. Wang et al., 2014 propose to use the triplet loss to learn a ranking of similar images, and demonstrate improvement compared to a Deep Neural Network (DNN) trained on a classification task. In MIR, Wolff and Weyde, 2014 and Lu et al., 2017 designed models for relative music similarity (i.e., rankings), rather

[1] https://gitlab.com/creaminal/publications/icassp2020-learning-to-rank-music-tracks

than absolute similarity ratings. However, no mining strategies from ranked list were proposed. This is because these two studies train their model using the similarity triplet annotations provided in the MagnaTagATune dataset (Law et al., 2009), which only has partial similarity information, annotated by non-expert users (see Chapter 3.1.1).

Our proposal takes inspiration from Lu et al., 2017 but proposes a mining strategy from ranked lists instead of partial similarity annotations. Some triplet mining strategies are inspired by the work of Wang et al., 2014, adapted from computer vision to MIR. In terms of architecture, we take inspiration from a simple and popular music auto-tagger proposed by Choi, Fazekas, and Sandler, 2016a. We reimplement this Convolutional Neural Network for our baseline auto-tagger system and adapt it to the metric learning task.

4.2 Proposed method

4.2.1 Problem definition

Let $D = \{t_1, \ldots, t_N\}$ be a set of N tracks annotated with a taxonomy of m tags. The problem addressed in this study is the following: given a query track t, compute a ranked list of tracks from the dataset D ordered by descending similarity to t. Let $\mathcal{S}$ be an oracle similarity function that, given two sets of tags $t_1 \in \{0,1\}^m$ and $t_2 \in \{0,1\}^m$, returns a similarity score $\mathcal{S}(t_1, t_2) \in \mathbb{R}_+$.

For any $t \in D$, let $R^{\mathcal{S}}(t) = [r_1(t), \ldots, r_{N-1}(t)]$ be the ordered list of the other tracks of D ranked by decreasing similarity according to the function $\mathcal{S}$. This is the ground truth, against which our system will be evaluated. For any $t \in D$, let $R^{\hat{\mathcal{S}}}(t) = [\hat{r}_1(t), \ldots, \hat{r}_{N-1}(t)]$ be the estimated list of recommended tracks made by the system for the query track t, using the estimated similarity function $\mathcal{S}$.

4.2.2 Mining triplets from ranked lists

We denote by $f_\theta(t) \in \mathbb{R}^d$ the embedding of the track t. f_θ is obtained by training a CNN with a triplet loss. To train such a system, it is necessary to prepare the data in the form of triplets (a, p, n). While the triplet loss is commonly used to project the data such that class separation is maximized (meaning that n represents a different class than the one of a and p), we use it here to reproduce music similarity rankings. In our scenario, we thus select a, p and n such that a is more similar to p than to n, according to $\mathcal{S}$. Let t be a query track and $R^{\mathcal{S}}(t) = [r_1(t), \ldots, r_{N-1}(t)]$ the associated reference ranking. We define training triplets by using the track t as the anchor and by mining a positive and a negative element from $R^{\mathcal{S}}(t)$. A triplet is considered *valid* if the index of the positive element is lower than the one of the negative element: $(a, p, n) = (t, r_i(t), r_j(t)) \quad \forall i < j$.

In practice, for large datasets, it is infeasible to use all valid triplets for training, because their number grows cubically with N. Additionally, all triplets may not be useful for training. Figure 4.1 (top) shows an illustration of the average similarity scores $[\mathcal{S}(t, r_1(t)), \ldots, \mathcal{S}(t, r_{N-1}(t))]$ for each track $t \in D$. The curve shows that a few first tracks in the ranking are very similar to their query, while most tracks in the dataset are actually irrelevant: their similarity score is lower than 50%. Therefore, after a certain rank in $R^{\mathcal{S}}(t)$, mining positive samples does not make sense.

Given these observations, we limit the set of possible positives per anchor to the N_p first elements of the reference ranking: $p \in [r_1(t), \ldots, r_{N_p}(t)]$. We also limit the overall number of negatives per anchor-positive pair to N_n. Then, to select the N_n

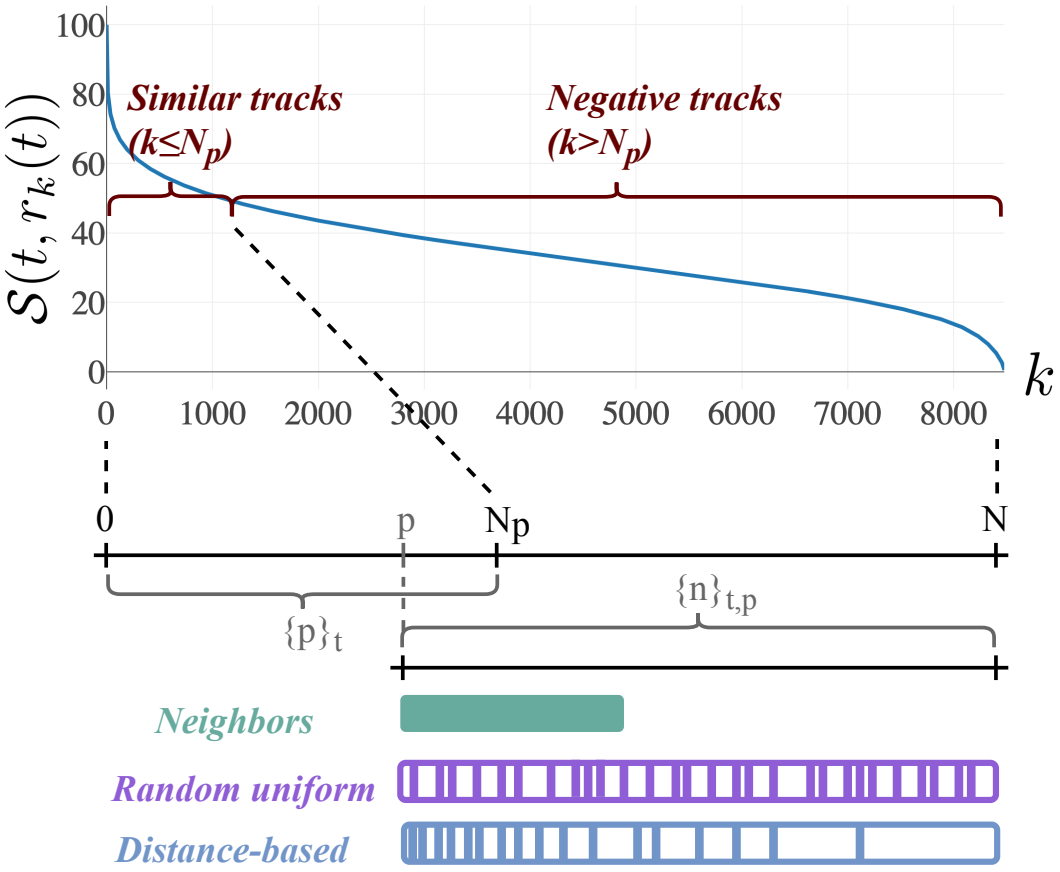


FIGURE 4.1: Illustration of the three proposed triplet mining strategies. Top: Similarity scores in a typical similarity ranking (in percentage), as a function of the rank k in the list $R^{\mathcal{S}}(t)$. Bottom: Three strategies to mine negative samples for a given anchor-positive pair (t, p).

negative examples for a given anchor-positive pair $(t, r_i(t))$, three different strategies are tested (see Figure 4.1, bottom):

- **Neighbors:** The negatives are the N_n elements in $R^{\mathcal{S}}(t)$ that come directly after the positive: $n \in [r_{i+1}(t), \ldots, r_{i+1+N_n}(t)]$.
- **Random uniform:** The negatives are sampled uniformly among the full $R^{\mathcal{S}}(t)$ list after the positive: $n \in \{r_j(t)\}_{j>i}$ s.t. $P(n) \sim \mathcal{U}(1/(N-i+1))$.
- **Distance-based:** The negatives are sampled among the full $R^{\mathcal{S}}(t)$ list after the positive with a probability that is proportional to its similarity with the anchor: $n \in \{r_j(t)\}_{j>i}$ s.t. $P(n) \propto \mathcal{S}(t,n)$ [2].

This way, in all three variants, the set of triplets can be pre-selected offline and we do not mine triplets during training. The resulting total number of triplets is $N \times N_p \times N_n$.

4.2.3 Auto-pooling

In usual CNNs applied to audio signals, the time dimension of the audio signal is progressively discarded by a succession of max-pooling layers. This implies a strong assumption related to how information over time is processed: we only keep the maximum activation over successive time frames. Other choices have been made in the past such as the combination of Mean, Max and L2 Pooling (Dieleman, 2014). A very elegant formulation has been proposed by McFee, Salamon, and Bello, 2018 with the Auto-Pooling layer, which allows to interpolate between several pooling operators (such as min-, max-, and average-pooling) via a learned parameter α:

$$\hat{P}_\alpha(Y|X) = \sum_{x \in X} \hat{p}(Y|x) \left(\frac{exp(\alpha.\hat{p}(Y|x))}{\sum_{z \in X} exp(\alpha.\hat{p}(Y|x))} \right) \tag{4.1}$$

When α is equal to 0, the operator behaves as an average pooling. When $\alpha 1$, it behaves as a softmax. When $\alpha \to \infty$, it approaches max pooling, and when $\alpha \leq 0$, it approximates mean pooling.

Auto-pooling has provided very good results for an audio event detection task. To our knowledge, it has not been used for music-related tasks. We do this here for the task of music similarity.

4.2.4 System architectures

In this chapter, the input representation used for all our architectures is the Constant-Q transform (CQT). For each track, we compute a CQT of 96 frequency bins and a hop size of 1024 (23 ms). We then convert it to power amplitude and log-scale the magnitudes. The input of our CNN is a patch of 512 time frames of CQT. 512 frames account for 11.88s. This duration was chosen as a good compromise between GPU memory efficiency and sufficient musical context. Since the annotations of our dataset are at the track level (see Section 4.3.1), we randomly select several of these

[2] The Distance-based sampling is inspired from Manmatha et al., 2017, who suggest to sample uniformly according to the distances between the embedding vectors. Here, we rather sample uniformly the ground truth distances given by $\mathcal{S}$. This is what was proposed by Wang et al., 2014 for intra-class negative mining (where $\mathcal{S}$ is computed using "golden features"). Since we have no classes, we extend the Distance-based sampling to all triplets.

TABLE 4.1: Details of the three architectures used. MP stands for Max Pooling, FC for Fully-Connected. In italic are the output sizes of the max-pooling layers.

AT Baseline	**TL**	**TL Autopool**
CQT *(input: F=96,T=512,C=1)*		
Conv2D (3,3)×128		Conv2D (3,3)×64
MP (2,4) *(F=48,T=128,C=128)*		MP (2,2) *(F=48,T=256,C=64)*
Conv2D (3,3)×256		Conv2D (3,3)×128
MP (2,4) *(F=24,T=32,C=256)*		MP (2,2) *(F=24,T=128,C=128)*
Conv2D (3,3)×512		Conv2D (3,3)×256
MP (2,4) *(F=12,T=8,C=512)*		MP (2,2) *(F=12,T=64,C=256)*
Conv2D (3,3)×1024		Conv2D (3,3)×512
MP (3,3) *(F=4,T=2,C=1024)*		MP (2,2) *(F=6,T=32,C=512)*
Conv2D (3,3)×2048		Conv2D (3,3)×1024
MP (4,2) *(F=1,T=1,C=2048)*		MP (6,1) *(F=1,T=32,C=1024)*
		FC (*d*) *(F=1,T=32,C=d)*
FC (*m*)	FC (*d*)	Autopool (1,32) (*d*)

(96×512) patches to represent a given track and assume that the annotations apply to each patch.

At test time, each track is represented by 8 randomly selected patches. When the network is used for auto-tagging, we pass each of them through our trained network to obtain the tag probability vector. We then simply use the average vector over the 8 tag probability vectors. When the network is used for embedding, we compute the average embedding vector over the 8 embedding vectors.

All models have been implemented using Keras framework (Chollet and others, 2015). We used an Adam optimizer (Kingma and Ba, 2015), a batch size of 42 patches and applied early stopping.

Baseline system: Auto-tagger (AT): The similarity function $\mathcal{S}$, used for ranking, relies on tag annotations. A naive approach is therefore to automatically estimate these tags from the audio and then apply $\mathcal{S}$ to those. Auto-tagging is a multi-label classification problem (output activations are sigmoids, and the loss is defined as the sum of binary cross-entropies). In preliminary experiments, we have shown that VGG-like architectures (Choi, Fazekas, and Sandler, 2016a) were the most suited to our task (see Appendix C.2 for details on the architecture search). Thus, we reproduce the *FCN-5* architecture proposed by Choi, Fazekas, and Sandler, 2016a. We adapt it to the shape of our inputs (96×512) and outputs (m tags). We remove the batch normalization and apply lower dropout rates. This system is referred to as *AT Baseline* in the rest of the chapter. We give its architecture in Table 4.1, column 1 and provide the details (dropout, activations) in our code. We train it with a learning rate of 10^{-4}.

Triplet loss system (TL): Our objective here is to estimate directly an embedding such that applying the Euclidean distance between the embeddings of two tracks (t_1, t_2) mimics $\mathcal{S}(t_1, t_2)$. The network we use to compute the embedding is similar to the *AT Baseline* one, but it differs in the output layer. The last fully-connected layer has now d units (the dimension of the embedding space) instead of the m units, and has linear activations instead of m sigmoid activations (see Table 4.1, column 2).

After a short grid search in a pilot experiment, we set d to 128. The embeddings are L2-normalized to the unit sphere. The margin parameter α of the triplet loss is set to 0.5 and the learning rate to 10^{-6}. Each mini-batch contains 42 triplets of patches. A given mini-batch represents one anchor track, one positive track and 42 negative tracks. Patches from these tracks are then randomly selected.

This network is referred to as *TL* in the rest of the chapter. We test it with the three sampling strategies presented in Section 4.2.2: *Neighbors, Random uniform* and *Distance-based*. N_p *is set to 15* in order to keep only the most similar tracks. N_n *is set to 250* to create a large, yet still tractable number of triplets. With these values, we obtain a total number of triplets of $N \times N_p \times N_n = 14,246 \times 15 \times 250 = 53,422,500$ in the Creaminal dataset.

Triplet loss system with Auto-pooling (TL Autopool): In the *AT Baseline* and *TL* networks, the time dimension is progressively removed by a succession of max-pooling layers. We test here the use of the Auto-Pooling layer proposed by McFee, Salamon, and Bello, 2018. Two main adaptations were necessary to use Auto-pooling in our setup. First, the max-pooling sizes of the *TL* network need to be adapted to carry some temporal information until the last layer. Second, the number of filters needs to be divided by two due to GPU memory constraints. This network is referred to as *TL Autopool* in the rest of the chapter (see Table 4.1, column 3). As for the *TL* network, we train it to output embeddings using the triplet loss. In the following, TL Autopool will only be tested using the Distance-based mining strategy.

4.3 Evaluation of the proposed method

4.3.1 Dataset

To test our proposal, we use the $N = 14,246$ tracks of the Creaminal dataset, described in Section 3.1.2. This dataset features tags annotated at the track level and an associated ground truth similarity function $\mathcal{S}$. Similar datasets also exist in streaming services such as Pandora.

Although in our case, $\mathcal{S}$ is a proprietary function, we shortly describe in this paragraph how it is computed from the set of manually annotated tags. $\mathcal{S}$ first expands the set of tags to include similar tags (for example, "Waltz" expands to "Classical" and "Dancing"). This expansion operation features a considerable number of manually tuned parameters. Weights are attributed to each tag, either manually by the annotators or during the expansion operation (tags added by $\mathcal{S}$ typically have lower weights than manually annotated tags). The weights of the tags are then combined in a non-linear way to compute the final matching score.

For our experiments, we split the dataset into a training, validation and test sets (60%, 20% and 20% respectively). The same artist cannot be represented in two different sets. The distribution of tags is approximately the same for training and testing. This dataset is private and cannot be shared but the proposals made here can be applied to other ones.

4.3.2 Evaluation metrics

To evaluate our systems, we used each track of the test set as a query and ask the systems to rank all the other tracks of the test set by decreasing similarity with the query. For the AT Baseline system, the similarity score is obtained by applying $\mathcal{S}$ to

the estimated tags. The likelihood of each tag, provided by the corresponding sigmoid unit, is used as a manually annotated weight. For both TL systems, the ranking is obtained by ascending order of Euclidean distance between the embedding of the query and these of the other tracks of the test set.

Let $R_k^{\mathcal{S}}(t) = [r_1(t), \ldots, r_k(t)]$ be the list $R^{\mathcal{S}}(t)$, truncated at rank k. Without loss of generality, we consider here that the *relevant* tracks to recommend for a given test query t are the five first tracks of the reference ranking: $R_5^{\mathcal{S}}(t)$. Thus, for each query track t, we ask our system to retrieve the 5 relevant ground truth recommendations $R_5^{\mathcal{S}}(t)$ among its k estimated recommendation $R_k^{\hat{\mathcal{S}}}(t)$. Since serendipity can be desired in music similarity applications, we want the system to recommend as many relevant tracks as possible in its top k $R_k^{\hat{\mathcal{S}}}(t)$, while not strongly penalizing irrelevant tracks. We therefore set k to 20. We then use four of the evaluation metrics proposed by (Urbano, 2013):

- *Mean Average Precision (MAP)*: evaluates if the relevant tracks appear in high position in $R_k^{\hat{\mathcal{S}}}(t)$. Higher is better.
- *Recall@k*: indicates which proportion of the relevant tracks appear in $R_k^{\hat{\mathcal{S}}}(t)$. Higher is better.
- *Reciprocal rank (RR)*: is the inverse of the rank of the first relevant track in $R_k^{\hat{\mathcal{S}}}(t)$. Since we consider only the top k, the reciprocal rank is set to 0 if the rank is higher than k. Higher is better.
- *Normalized Discounted Cumulative Gain (nDCG)*: this metric allows to have a *relevance scale* instead of binary relevance judgments (e.g., recommending $r_1(t)$ will produce a higher score than recommending $r_5(t)$ at the same rank in $R_k^{\hat{\mathcal{S}}}(t)$). Higher is better. Details of the nDCG that we used can be found in Appendix C.1.2.

4.3.3 Quantitative results

In Table 4.2, we compare the systems AT Baseline, TL (with the three mining strategies *Neighbors*, *Random uniform* and *Distance-based*) and TL Autopool. We indicate for each system the average and the confidence interval at 95% of the four metrics (expressed as percentages). We performed all experiments (training and evaluation) on a Nvidia GeForce GTX 1080 Ti GPU (see Appendix B.1 for more hardware details).

We first observe that the AT baseline is outperformed by all TL systems on all metrics. This shows that in our case, directly learning the ranking is clearly more efficient than learning the tags and applying $\mathcal{S}$ to their estimates.

We then see that among the various mining strategies of the TL systems, the Distance-based negative sampling performs best on all metrics. It should be noted that the Distance-based negative sampling was initially proposed by Manmatha et al., 2017 which uses the learned embeddings to compute the distance online. In our case, the distance can be calculated offline since we use the ground truth $\mathcal{S}$ instead of the embeddings for the distance computation. The Neighbors and Random uniform sampling strategies have similar performances, but the first has a better reciprocal rank while the latter has a better recall. This means that the Neighbors mining strategy favors a high ranking of the relevant tracks over retrieving as many of these as possible The Random uniform strategy, which exposes the network to a larger variety of negative samples, acts the other way around. The Distance-based strategy

TABLE 4.2: Comparison of the results of the AT Baseline vs TL systems (with various sampling strategies) vs TL Autopool. All metrics are percentages. Higher is better.

Model	MAP@20	Recall@20	RR@20	nDCG@20
AT Baseline	4.50 ± 0.34	12.57 ± 0.66	15.62 ± 1.07	11.30 ± 0.69
TL Neighbors	5.58 ± 0.41	12.73 ± 0.67	19.18 ± 1.23	13.41 ± 0.80
TL Random uniform	5.39 ± 0.38	15.01 ± 0.70	17.86 ± 1.12	13.50 ± 0.76
TL Distance-based	**5.98 ± 0.40**	**15.79 ± 0.73**	**19.89 ± 1.19**	**14.41 ± 0.78**
TL Autopool	**7.99 ± 0.51**	**17.74 ± 0.79**	**24.68 ± 1.34**	**17.95 ± 0.92**

gives the best of both worlds. We did not experiment with different values of N_p and N_n, however we expect these to play a role in the system's performance. Although a too large value of N_p could create noise in the triplets, we hypothesize that the larger the number of negatives N_n, the better the generalization performance will be. We did not increase N_n further than 250 for computational reasons.

The last row of Table 4.2 indicates the results of the TL Autopool system. We observe a boost in performances due to the added flexibility of Auto-pooling. This system is able to retrieve one of the top 5 most relevant tracks with almost a probability of 1/5 (Recall@20 = 17.74). It should be reminded that in our case, neither the query nor the reference tracks have been seen during training. The first relevant track is on average at rank 5.6 (inverse of RR@20=24.68), among approximately 2,900 test tracks. This makes our system a promising approach to efficient music retrieval in large datasets. Additionally, an informal listening test reveals that some of the recommended tracks, although judged "irrelevant" by our evaluation system, actually share important characteristics with the query[3].

4.3.4 Qualitative results

To further explore our results, we conducted a qualitative analysis of the MuSimNet's embedding space. Our hypothesis is that during training, the output units specialize in detecting different characteristics in the music. Then, at inference time, the tracks which embedding vectors take high values along the same dimension share the corresponding characteristics and are therefore similar. In this part, we consider only the system with Distance-based negative mining and Autopooling ("TL Autopool").

To verify our hypothesis, we took inspiration from Dieleman, 2014: we browsed the 128 units of the output layer. For each output dimension, we looked in the test set for the top 5 tracks that activated the most the corresponding unit. We listened to the generated playlists and tried to identify similarities between the tracks. Not all units did correspond to obvious characteristics, but we were nonetheless able to find interesting patterns for some of them.

[3]We provide listening examples for randomly selected queries at:
https://www.creaminal.io/#/access/642198129c5e26e47455897c84741d03

TABLE 4.3: Qualitative MuSimNet results: certain output units are specialized in detecting identifiable musical characteristics. The first column indicates the number of the considered unit, with a link to listen to the associated playlist. The second column denotes our manual annotation of the common characteristics of the 5 tracks which activate the most said unit. The third column denotes the tags of the Creaminal dataset which were common to all tracks of the unit's playlist, along with their number of occurrences. In italic, we highlighted tags that were similar to our annotations.

Dimension	Common characteristic	Creaminal tags
2 [Click to listen]	Calm, ethereal atmosphere, movie soundtrack, piano	*Acoustic: 5, Instrumental: 4,* Hypnotic: 3, Violins: 2, *Suspense: 2, Linear: 2, Adventure: 2, Dreamy: 2, Grand Spectacle: 2, Slow Motion Images: 2,* Nostalgic: 2, Repetitive: 2
4 [Click to listen]	Retro keys	Dreamy: 4, Hypnotic: 4, Instrumental: 3, *Electronic: 3,* Acoustic: 3, *Reverb: 2, Psyche: 2,* Laid-back/Cool: 2, Ethereal: 2, Repetitive: 2, Mainstream: 2
5 [Click to listen]	Americana Folk	*English: 5,* Electric: 5, Happy: 4, Quirky: 3, *Americana: 2,* Electric Blues: 2, Alternative Pop: 2, Naive / Simple: 2, Acoustic: 2, Calm: 2, Male Lead: 2
7 [Click to listen]	Female singers	English: 5, Acoustic: 5, *Female Lead: 4,* Mainstream: 4, Verse-Chorus: 3, Guitar: 3, Reverb: 3, Dreamy: 3, Electric: 2, Indie Rock: 2, Happy: 2, Naive / Simple: 2, Nostalgic: 2, 1-3: 2, Folk Acoustic: 2, *Female Choir: 2,* Calm: 2, Emotional: 2, Intimate: 2, Slow Motion Images: 2, *Feminine: 2,* Banjo: 2, Hopeful: 2
8 [Click to listen]	Hand claps	Acoustic: 4, Male Lead: 3, English: 3, Unifying: 3, *Onomatopoeia: 3,* Electronic: 2, Performance: 2, Instrumental: 2, Electric: 2, Quirky: 2
13 [Click to listen]	Guitar, soft atmosphere	*1-3: 5, Acoustic: 5,* English: 5, *Guitar: 5,* Mainstream: 5, *Intimidate: 5,* Female Lead: 4, Folk Acoustic: 4, *Calm: 4,* Emotional: 4, *Dreamy: 4, Slow Motion Images: 4, Ethereal: 4,* Reverb: 4, *Gracious: 4,* Vocal Jazz: 3, *Nostalgic: 3,* Jazz Pop/Light Jazz: 2, *Elegant: 2,* Verse-Chorus: 2, Whirling: 2

Table 4.3 shows examples of units which significantly detected musical characteristics. Top tracks activated by unit 2 sounded a lot like movie soundtracks, played by acoustic instruments such as pianos. This is confirmed by the manual annotations, which include several movie categories: "Suspense", "Adventure", "Grand Spectacle", "Slow Motion Images". The presence of vintage keyboard sounds, common to the tracks of playlist 4, was not directly identifiable via the tag annotations, which only described consequences of this feature: "Electronic", "Reverb". In this case, the listening test (following the method of Dieleman, 2014) proved to be necessary. For some units, the distinctive audio features detected by the MuSimNet are not reflected by the tag annotations, because they were not judged relevant enough by the annotators. This is for example the case with the handclaps detected by unit 8.

The main limitation of our approach is that it explains only the audio characteristics detected by one single unit. We hypothesize that other characteristics can be detected not by one single unit, but a combination thereof, making them invisible when exploring the embedding space only axis by axis.

4.4 Summary

In this chapter, we proposed a method to learn a similarity ranking using a triplet loss network and a dataset of reference rankings. This system has applications in music supervision for large music catalog management, and to expand track suggestions. We show that to estimate the pairwise similarity function $\mathcal{S}$, auto-tagging isn't an interesting workaround, even though $\mathcal{S}$ is originally computed from tag annotations. We show that using the triplet loss to directly learn the ranking provided by $\mathcal{S}$ gives better results. This result is consistent across all our metrics. Finally, we showed that Auto-pooling, proposed in the framework of audio event detection, also allows improvement in the case of music similarity. A qualitative review confirms that our music similarity system produces an embedding space in which certain dimensions correspond to distinctive audio characteristics.

Therefore, in the next chapter, we test reusing the MuSimNet as an audio feature extractor for V/M matching. We will consider only our best performing system, that uses the Distance-based triplet mining strategy and the Auto-pooling layer.

Chapter 5

Video/Music Recommendation: A Study of Design Choices

In Chapter 4, we saw that trained music embeddings are useful to perform similar music recommendation given a music query. Moving to our second objective, from this chapter, we study multimodal embeddings for the Video/Music recommendation task. Our problem is the following: given a query video clip, which music track from a music catalog is the most suitable to serve as a soundtrack? And inversely, given a music track, which video from a collection best illustrates its content? How to design an efficient system for this task, that can be trained without annotated data? Besides professional music supervision (Inskip, Macfarlane, and Rafferty, 2008), music recommendation for video has applications in automatic video editing (Shah, Yu, and Zimmermann, 2014). Video recommendation for music has applications in automatic MTV generation (Liao, Wang, and Zhang, 2009) and music recommendation (Zeng, Yu, and Oyama, 2018).

Real-world data is inherently multimodal. When we observe our environment, we usually gather information using several senses at the same time. Co-occurrence of events across our senses allows us to efficiently acquire a large amount of knowledge without the need for ground truth annotations. Similarly, it is possible to leverage multimodal data to learn this type of correspondence. Starting from this observation, an active research area has developed around multimodal learning paradigms (Ngiam et al., 2011; Wang et al., 2016; Aytar, Vondrick, and Torralba, 2017; Müller et al., 2019). One of the most popular approaches uses multimodal embedding spaces to associate audio data and video data, in order to perform a variety of tasks (Aytar, Vondrick, and Torralba, 2016; Owens et al., 2016; Arandjelovic and Zisserman, 2017; Schindler, 2019; Parekh et al., 2019). By learning to distinguish between matching and non-matching audio-video pairs, self-supervised systems can learn effective joint representations (or embeddings) of the audio and video. Thereupon, it is possible to solve the cross-modal recommendation task using only the content of the videos and music tracks. Such systems do not exploit mood tags (Sasaki et al., 2015; Li and Kumar, 2019), usage data or other prior information (Shah, Yu, and Zimmermann, 2014). They only require a large dataset of video clips to learn the matching of music and video.

However, common audio-video self-supervised systems are computationally intensive to train (2 days on 16 GPUs in parallel for the L3-Net of Arandjelovic and Zisserman, 2017). One way to alleviate this issue is to leverage pre-trained input features. It has been shown that pre-trained features associated with fine-tuning can match the performance of systems trained from scratch (Oquab et al., 2014). This allows to reduce the computational cost associated to the development of new systems, and to improve performances by exploiting a large diversity of datasets.

Reusing pre-trained features can also be critical in domains where access to training data is limited (e.g., due to copyright, or high cost of acquisition). In image and video processing, ImageNet features are commonly used as a starting point for other tasks (Yue et al., 2015; Balntas, Li, and Prisacariu, 2018; Wang et al., 2014). In music research, however, there is no widely accepted feature extractor that would be the equivalent of ImageNet features. Several systems, both unimodal and multimodal, can play this role (Gemmeke et al., 2017; Hershey et al., 2017; Kumar, Khadkevich, and Fugen, 2018; Pons and Serra, 2019; Cramer et al., 2019). Grollmisch et al., 2020 evaluated some of these on several audio classification tasks. We study here which audio embedding is the most appropriate for the cross-modal V/M recommendation task. Finally, we also study which loss and which feature aggregation method give the best results on this task.

Chapter contributions:

1. We build upon the VM-Net (Hong, Im, and Yang, 2018), a self-supervised network for V/M content matching, and challenge some of its design choices. We validate the choice of the cross-modal triplet loss originally proposed in the VM-Net compared to the binary cross-entropy loss commonly used in self-supervised learning.

2. We compare the performance of several open-source audio embeddings for this recommendation task, including MusiCNN, OpenL3 and AudioSet. We show that audio features already pre-trained on a cross-modal task (Cramer et al., 2019) perform best. To our knowledge, this is the first time these embeddings are used in the context of V/M recommendation.

3. We compare several methods to aggregate frame-level features into clip-level features for training. We show that an Auto-pooling layer or a simple Attention mechanism (Raffel and Ellis, 2015) gives better performance compared to the originally proposed statistical aggregation.

Experiments 1 and 2 were published at the Special Session on Representation Learning for Audio, Speech, and Music Processing of the IJCNN conference (Prétet, Richard, and Peeters, 2021a). Experiment 3 presents original results.

Chapter organization. In Section 5.1, we describe the reference system that we use as baseline (Hong, Im, and Yang, 2018). In Section 5.2, we challenge three design choices of our baseline system by changing the training loss, the input audio features, and the temporal aggregation method. In Section 5.3, the performances of these variants are evaluated on an independent test set. Some conclusions are suggested in Section 5.4.

5.1 Baseline Video/Music recommendation system

In this part, we briefly describe the Video/Music (V/M) recommendation system that we will consider as our baseline. This system, named VM-Net, has been proposed by Hong, Im, and Yang, 2018. The VM-Net learns music-video embeddings in a self-supervised way (without the need for external annotation). It is a two-branch network where music and video are independently projected into the same embedding space, such that the embeddings of associated music tracks and videos are

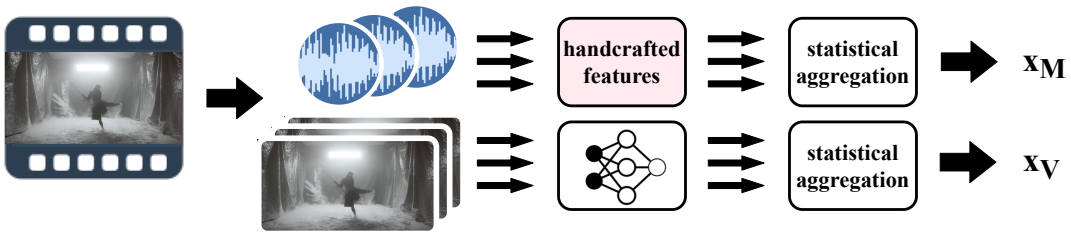


FIGURE 5.1: Generating the timeless input vectors of the VM-Net by statistical aggregation.

close to each other. It therefore allows cross-modal recommendation, hence Music-to-Video (M2V) or Video-to-Music (V2M) recommendation.

5.1.1 VM-Net inputs: Feature vectors

One specificity of the VM-Net is that the input of each branch is timeless, i.e., the time evolution of the track (either video or music) has been summed up as a timeless vector (see Figure 5.1). As opposed to other works (Arandjelovic and Zisserman, 2017), those inputs are not the raw data (raw audio waveform or spectrogram, or images) but audio and image features precomputed by another system. This allows to be efficient in terms of computational resource and to only learn the multimodal projection.

Video input: x^v is a timeless feature vector of dimension 1,024 which represents the whole clip duration. To construct this vector, ImageNet features (Abu-El-Haija et al., 2016) are computed for images of the videos sampled every second. These features correspond to the output of the penultimate layer of an Inception model trained on ImageNet, and post-processed for dimensionality reduction. The ImageNet features are then statistically aggregated using mean over the whole clip duration.

Music input: x^m is a timeless feature vector of dimension 1,140 which represents the whole clip duration. To construct this vector, handcrafted features (such as spectral centroid, MFCCs, and Chromas) are extracted at each time frame (1 frame$\simeq$21ms) and statistically aggregated using mean, variance and max.

5.1.2 VM-Net architecture: Fully-connected network

The architecture of the VM-Net is made of two independent branches: one for the music $e^m = f_m(x^m)$ and one for the video $e^v = f_v(x^v)$ (see Figure 5.2). The music branch f_m consists of 3 fully-connected layers of 2,048, 1,024, and 512 units respectively. The video branch f_v consists of 2 fully-connected layers of 2,048 and 512 units respectively. ReLU activations are applied after each layer, except the last one. Linear activations and batch normalization are applied after the last layer of each branch. A L_2 normalization layer is finally applied to ensure all embedding vectors have unit norm. This architecture accounts for 6,037,504 trainable parameters and 12,084,249 floating-points operations. The outputs of the two branches, e^m and e^v, are considered as the music and video embeddings. Preliminary experiments validated the original architecture.

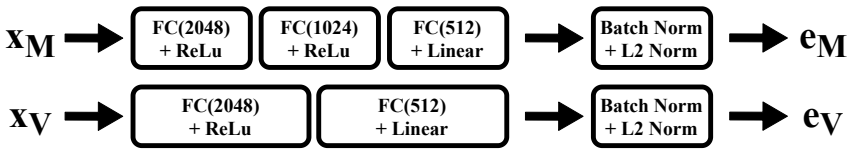


FIGURE 5.2: The VM-Net architecture relies on two branches of fully-connected layers.

5.1.3 VM-Net training: Self-supervised metric learning

The parameters of the VM-Net are trained using metric learning in a self-supervised way. The goal is to project the music and the video into the same multimodal embedding space, such that the music embedding e_c^m and video embedding e_c^v extracted from the same AV clip c are closer together than from any other sample. Closeness and distance here are defined in terms of Euclidean distance.

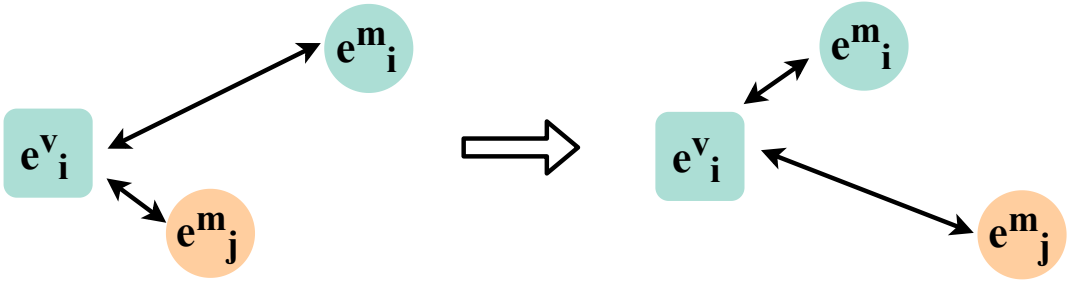
To obtain this property, we train the VM-Net with an extended triplet loss (see Chapter 2.1.2 for the generic triplet loss definition). In the VM-Net, the triplet loss is used not only to ensure that the music embeddings e_c^m and video embeddings e_c^v from the same clip c are close but also that, for all clips, the distances between their music embeddings e_c^m (resp. video embeddings e_c^v) remains similar to the distance between their music features x_c^m (resp. video features x_c^v). This leads to the definition of an extended triplet loss $\mathcal{L}_{VMNET}$ with four constraints grouped into two types: inter-modal ranking constraints ($\mathcal{L}_{VM}, \mathcal{L}_{MV}$) and soft intra-modal structure constraints ($\mathcal{L}_{MM}$ and $\mathcal{L}_{VV}$). These constraints are combined via a weighted sum: $\mathcal{L}_{VMNET} = \lambda_1\mathcal{L}_{VM} + \lambda_2\mathcal{L}_{MV} + \lambda_3\mathcal{L}_{VV} + \lambda_4\mathcal{L}_{MM}$.

For the **inter-modal ranking constraints** $\mathcal{L}_{VM}$ and $\mathcal{L}_{MV}$, the triplets are defined as such: a and p represent the same AV clip, while n represents a different AV clip (see Figure 5.3a). Let us denote by x_c^v the value of x^v for the clip $c \in \{1, \ldots, C\}$. The triplet loss is then computed in both directions, using either the music or the video as anchor: $\mathcal{L}_{VM}$ with $(a, p, n) = (x_i^v, x_i^m, x_{j\neq i}^m)$ and $\mathcal{L}_{MV}$ with $(a, p, n) = (x_i^m, x_i^v, x_{j\neq i}^v)$. This ensures that each sample in the multimodal embedding space gets closer from its paired sample than from any other sample.

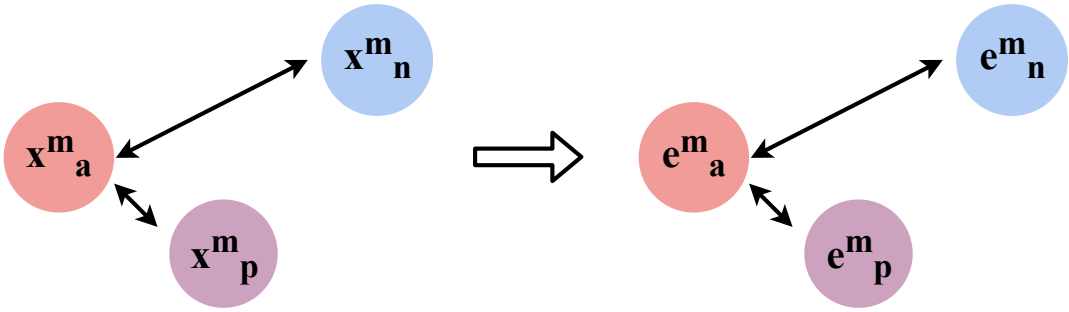
The **soft intra-modal structure constraints** are computed separately for the music ($\mathcal{L}_{MM}$) and for the video ($\mathcal{L}_{VV}$). We only describe it for the music case (see Figure 5.3b). The triplets are mined such that a, p and n are taken from three different clips but from the same modality (music in this case). For selecting a, p and n we use the values of the input features x^m (not the values of the embeddings e^m). We select them such that the feature vectors x_a^m and x_p^m are closer (in terms of Euclidean distance) than x_a^m and x_n^m. The triplet loss is then applied to the corresponding embeddings: $(a, p, n) = (e_a^m, e_p^m, e_n^m)$ for $\mathcal{L}_{MM}$. This ensures that the "modality-specific characteristics of the input features are preserved even after the embedding process." The process is the same for defining $\mathcal{L}_{VV}$ (video case) using $(a, p, n) = (e_a^v, e_p^v, e_n^v)$.

5.1.4 Datasets: HIMV-50K, MVD

The original VM-Net system is trained on a subset of the YouTube-8M dataset (Abu-El-Haija et al., 2016). We use the HIMV-50K, described in Section 3.2.1, an adaptation of the original training dataset. To evaluate the VM-Net, we use a cleaner dataset, the MVD (see Section 3.2.2).



(a) The $\mathcal{L}_{VM}$ inter-modal ranking constraint. Video e_i^v and music e_i^m taken from the same AV clip are brought together, while external music e_j^m is pushed away.



(b) The $\mathcal{L}_{MM}$ music intra-modal structure constraint. Relative distances between music input features are preserved during training.

FIGURE 5.3: The two types of constraints present in the $\mathcal{L}_{VMNET}$ loss. Squares represent the video modality, while circles represent the audio modality. Same colors represent the same clip.

5.2 Proposed experiments

5.2.1 Experiment 1: Triplet loss (TL) or Binary Cross-Entropy (BCE)

As shown by Arandjelovic and Zisserman, 2017 with the L3-Net, it is possible to learn relevant audio-video embeddings with a binary cross-entropy loss. In the first experiment, we then replace the $\mathcal{L}_{VMNET}$ triplet loss by a binary cross-entropy loss. To do so, we stack three fully-connected layers on top of the VM-Net with 1,024, 128 and 1 unit respectively. The output activation is now a sigmoid. The target of the network is set to 0 for non-matching audio-video pairs and 1 for matching pairs. We therefore train the VM-Net for a binary classification task. For the music branch, we still use the original handcrafted features.

5.2.2 Experiment 2: Input audio features

The VM-Net uses pre-trained ImageNet features for its video branch and handcrafted audio features for its music branch. This latter choice may be sub-optimal considering recent advances in feature learning using deep learning. We therefore gather four pre-trained music embeddings and use them as audio feature extractors for our multimodal embedding training (See Figure 5.4). We do not retrain them with the VM-Net, for more efficient use of computation resources (we refer to the respective papers for more details on how each of them was trained). We train each network with the $\mathcal{L}_{VMNET}$ triplet loss.

MuSimNet: We use the output of a pre-trained `MuSimNet` to compute the x^m. We proposed this system in Chapter 4 to estimate music similarity and we trained

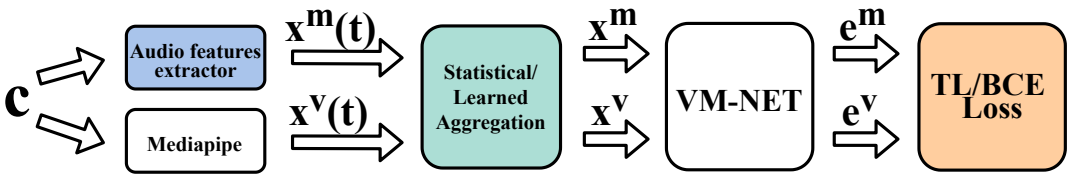


FIGURE 5.4: Experiment 1: Replacing the Triplet Loss by the BCE Loss (orange block). Experiment 2: Substituting handcrafted audio features for pre-trained audio features (blue block). Experiment 3: Switching from statistical aggregation to learned aggregation of the frame-level input features (green block).

it using a triplet loss paradigm (Prétet, Richard, and Peeters, 2020). For each track, we randomly select 24 segments of 12s duration and compute their corresponding embeddings. We then aggregate those over time using mean, variance and max, resulting in a single $3 \times 128 = 384$-dimensional feature vector for each music track.

MusiCNN: We compute the output of the penultimate layer of the music auto-tagging model MusiCNN (Pons and Serra, 2019). The open-source library `musicnn` was installed according to the instructions[1]. We compute the frame-level features at a rate of 1Hz and we aggregate those over time using mean, variance and max, resulting in one single 600-dimensional feature vector for each music track.

OpenL3: We compute the music embeddings provided by the OpenL3 model. OpenL3 has been trained to obtain embeddings usable for an audiovisual correspondence task. This is a task similar to ours. The open-source library `openl3` was installed according to the instructions[2]. We compute the frame-level features at a rate of 1Hz and we aggregate those over time using mean, max and var, resulting in one single 18,432-dimensional feature vector for each music track. We use the system trained on the "music" dataset.

AudioSet: We use the official audio features from the YouTube-8M dataset. These features are obtained from a VGG-inspired acoustic model trained on an auto-tagging task on the YouTube-8M dataset (Gemmeke et al., 2017). That this is the same dataset as the one we use for training the VM-Net. We compute the frame-level features at a rate of 1Hz and average those over time to obtain one single 128-dimensional feature vector for each music track.

5.2.3 Experiment 3: Learned temporal aggregation

In order to train the VM-Net, it is necessary to first compute frame-level features and then to aggregate them temporally in order to form a pair of timeless input vectors (see Figure 5.1). Hong, Im, and Yang, 2018 propose to use statistical metrics such as mean, variance and max. We expect this choice to be sub-optimal, provided that neural networks are able to learn temporal aggregations on their own using dedicated layers. Recurrent neural networks can be used to summarize the sequence of frames; however, because of vanishing gradient issues, they are not suited for very long sequences (of 250 frames on average in the HIMV-50K dataset). In our last

[1] https://github.com/jordipons/musicnn
[2] https://github.com/marl/openl3

experiment, we therefore test two recently proposed types of layers to perform the aggregation:

- **Auto-pooling:** In Chapter 4, we demonstrated the efficiency of the recently proposed Auto-pooling layer (McFee, Salamon, and Bello, 2018) for a music similarity task. This layer allows to interpolate between several pooling operators (such as min-, max-, and average-pooling) via a learned parameter.

- **Attention:** Attention mechanisms (Bahdanau, Cho, and Bengio, 2014) have been increasingly popular in language modeling, and many formulations were proposed (Pilehvar and Camacho-Collados, 2020). Attention layers are able to weight input elements according to a learned importance factor, allowing to model context and long-term dependencies. We reproduced a very simple self-attention mechanism, as proposed by Raffel and Ellis, 2015. Here, we assume that a linear combination of the frame-level input vectors x_t by learned attention weights α_t will provide more accurate aggregation than the original statistical features. In our implementation, the attention weights α_t for each time frame t are computed via a fully-connected softmax layer of parameters w and b and a hyperbolic tangent activation function:

$$\alpha_t = \frac{exp(tanh(w^T x_t + b)))}{\sum_{k=1}^{T} exp(tanh(w^T x_k + b))},$$

  and the final output of the layer is computed as a linear combination of the weights with the inputs:

$$y = \sum_{t=1}^{T} \alpha_t x_t.$$

To validate our hypothesis, we train the VM-Net by adding the Auto-pooling or Attention layer at the input level, before the fully-connected layers of each branch (See Figure 5.4). For each clip, we randomly select $T = 100$ frames at each epoch and pass them to the considered aggregation layer, which we train jointly with the rest of the network. We repeat the same frame sampling strategy at inference time. In this experiment, we train each network with the $\mathcal{L}_{VMNET}$ triplet loss and we consider the AudioSet audio features only (frame rate = 1 Hz).

5.2.4 Training details

We reimplemented the VM-Net in Keras from the provided Tensorflow code[3]. After a preliminary validation experiment, we validated our re-implementation and kept the original hyperparameters of the triplet loss $\mathcal{L}_{VMNET}$. We used the Adam optimizer with a learning rate of 10^{-6}, a dropout scheme with a probability of 0.5, and a batch size of 1,000. For all TL systems, we used early stopping to prevent overfitting. Stopping happened after 500,000 epochs on average. For the BCE system, preliminary evaluations on the validation data showed that early stopping was detrimental for the performances. Instead, we trained the VM-Net for 15,000 epochs.

To compute the handcrafted audio features, we used the original VM-Net code, which makes use of the Librosa library[4] (McFee et al., 2015). In experiment 1, we

[3] https://github.com/csehong/VM-Net
[4] https://librosa.org/doc/

only use these handcrafted audio features. The video ImageNet and audio AudioSet features are computed using the Mediapipe library[5].

All experiments are performed on a Nvidia GeForce GTX 1080 Ti GPU (see Appendix B.1 for more hardware details). Running the evaluation script takes approximately 30 seconds for the TL systems, and 5 minutes for the BCE system.

5.3 Evaluation and results

5.3.1 Evaluation metrics

Because of its higher quality, we only evaluate the VM-Net on the MVD dataset, used as a test set. The training is done using the HIMV-50K dataset.

Tasks. We evaluate our systems for two tasks:

- *Music-to-Video (M2V)*: from a music query, retrieve the corresponding video.
- *Video-to-Music (V2M)*: from a video query, retrieve the corresponding music.

In both cases, there is a single correct item to be retrieved among the 1,000 of the MVD dataset. Since this is a different dataset from HIMV-50K, none of the test clips (query nor targets) were seen during training.

For the M2V task, given a music query q, we first compute a ranked list of recommendations for q. To do so, we compare the pairwise distances $\{\delta(e_c^v, e_q^m)\}_{c\in\{1,\dots,C\}}$ between the query music embedding e_q^m and all video track embeddings from the test set $\{e_c^v\}_{c\in\{1,\dots,C\}}$. This ranking distance δ depends on the experiment. When using the triplet loss, δ is taken as the Euclidean distance between e_c^v and e_q^m (smaller distance means more similar). When using the BCE loss, δ is taken as the output of the final sigmoid layer $P(y = 0|x_c^v, x_q^m)$ (larger likelihood means more similar). We then rank all videos by increasing distance δ.

For the V2M task, we swap the audio and video modalities.

Performance measures. For a given query q, if the corresponding item is in the top k of the ranked list, we set the Recall at k ($R@k$) to 1, otherwise to 0. For each query, $R@k$ is hence binary. We repeat this operation using all 1,000 music tracks of the MVD dataset as the query. The final metric is the average of $R@k$ over the 1,000 test AV clips, displayed as percentages. Higher is better. We do not display confidence intervals for $R@k$, as it is a binary metric, and its distribution cannot be considered Gaussian. We use $k \in \{1, 10, 25\}$.

We also compute the *Rank* at which the target video appears in the ranked list. If it is displayed first, we set the *Rank* to 1, if it is the last recommendation, the *Rank* is equal to 1,000. Lower is better. We repeat this operation using all music tracks of the test set as the query q^v. The final metric is the mean *Rank* over the 1,000 test clips. We display the confidence interval at 95% for the *Rank*.

5.3.2 Results of the Experiment 1

Table 5.1 shows the results of Experiment 1 on the comparison of two training loss functions. Since the test set consists of 1,000 clips, we denote as "Chance" the recall

[5] https://github.com/google/mediapipe

TABLE 5.1: Results of Experiment 1: comparing different training loss functions. The Triplet Loss performs better than the BCE loss in terms of *R@k* but worse in terms of *Rank*.

	Music to Video				Video to Music			
	R@1	R@10	R@25	Rank	R@1	R@10	R@25	Rank
Chance	0.10	1.0	2.50	500	0.10	1.0	2.50	500
TL	**3.10**	**11.40**	**19.40**	205 ± 27	**2.70**	**11.60**	**19.50**	209 ± 27
BCE	1.60	8.60	18.50	$\mathbf{188 \pm 12}$	2.10	9.80	18.50	$\mathbf{185 \pm 11}$

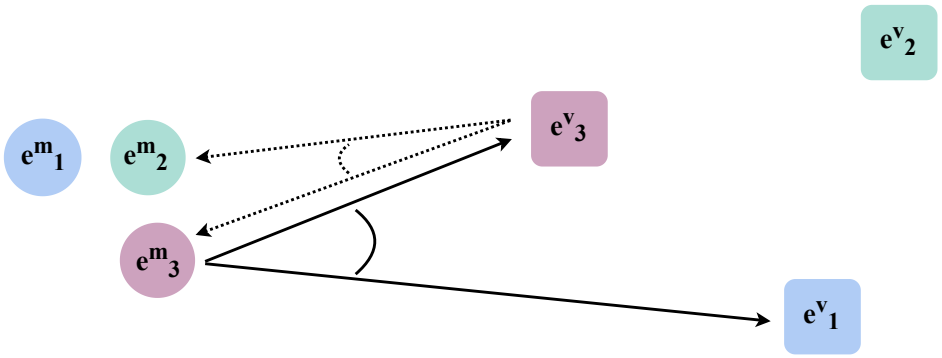


FIGURE 5.5: Illustration of why a difference in dispersion within each media may explain a difference between the M2V and V2M scores (TL case).

expectancy of a random recommendation system. With the triplet loss, the VM-Net trained in 18 hours and with the BCE in 20 hours, so the training time of both systems is similar. On Table 5.2, we observe that replacing the $\mathcal{L}_{VMNET}$ loss by the BCE seems to degrade the performances in terms of *R@k*, but improves them in terms of *Rank*. We hypothesize that the triplet loss, which allows to perform metric learning, is more suited to our recommendation problem than the BCE, which trains a projection for a discriminative task.

In the BCE experiment, the ranking distance δ represents the model's logistic regression: $\delta(x_c^v, x_q^m) = P(y = 0|x_c^v, x_q^m)$. Therefore, the BCE system has a priori no reason to return the same results for M2V ($\delta(x_q^v, x_c^m)$)) and V2M ($\delta(x_c^v, x_q^m)$). By contrast, the TL system projects each sample on a single point in the embedding space, and in this case, δ is taken as their Euclidean distance. The difference between the results obtained for M2V and V2M with the TL system may be explained by the distribution of the data in the embedding space. We computed the average dispersion of e^v and e^m and obtained 0.628 for music and 0.940 for video. Since the e^v are more spread, it is easier to distinguish among them given a music query than it is to distinguish the e^m given a video query (see Figure 5.5).

5.3.3 Results of the Experiment 2

Table 5.2 shows the results of Experiment 2, which is about comparing different audio input features. The two first rows show that the MuSimNet features perform comparably to the handcrafted features. The fact that the MuSimNet features, which have been trained for a music similarity task, do not outperform the handcrafted ones can be explained by the fact that features that represent music similarity may not help to represent V/M similarity. The MusiCNN features, which were trained on a music tagging task, perform a little better than the MuSimNet features. Here, we

TABLE 5.2: Results of Experiment 2: comparing different audio input features (loss: TL). Conclusion: the handcrafted features are outperformed by pre-trained embeddings such as AudioSet or OpenL3.

	Music to Video				Video to Music			
	R@1	R@10	R@25	Rank	R@1	R@10	R@25	Rank
Chance	0.10	1.0	2.50	500	0.10	1.0	2.50	500
Handcrafted	3.10	11.40	19.40	205 ± 27	2.70	11.60	19.50	209 ± 27
MuSimNet	1.20	12.20	24.60	172 ± 25	1.60	12.60	21.00	176 ± 25
MusiCNN	3.30	16.20	29.50	149 ± 23	2.70	15.10	28.60	148 ± 23
AudioSet	2.70	19.30	35.90	108 ± 18	3.00	**18.00**	**31.90**	$\mathbf{116 \pm 18}$
OpenL3	**4.10**	**23.50**	**39.90**	$\mathbf{101 \pm 18}$	**3.20**	17.40	31.60	124 ± 20

TABLE 5.3: Results of Experiment 3: comparing different input features aggregation methods (loss: TL, audio features: AudioSet). Conclusion: the statistical aggregation is outperformed by the learned aggregation layers ; Attention performs best.

	Music to Video				Video to Music			
	R@1	R@10	R@25	Rank	R@1	R@10	R@25,	Rank
Chance	0.10	1.0	2.50	500	0.10	1.0	2.50	500
Statistics	2.70	19.30	35.90	108 ± 18	3.00	18.00	31.90	116 ± 18
Auto-pooling	3.70	21.90	36.70	109 ± 19	2.90	20.30	37.10	109 ± 18
Attention	**4.10**	**24.10**	**38.70**	$\mathbf{103 \pm 18}$	**4.00**	**22.50**	**38.30**	$\mathbf{105 \pm 18}$

assume that the VM-Net was able to take advantage of the generic music knowledge encapsulated in the MusiCNN features.

Finally, the two best performing embeddings are Audioset and OpenL3. These two embeddings were trained on a much larger dataset than the MuSimNet and MusiCNN. Note that the Audioset features were trained on a classification task on the YouTube-8M dataset, so the same content as for the video branch. Note also that the OpenL3 features were trained on an audiovisual correspondence task, so a task similar to ours. Additionally, the OpenL3 features were trained on music videos, which is closer to our application scenario.

The last row of Table 5.2 shows that when using OpenL3 features, the VM-Net achieves a $R@25 = 39.90$ for music-to-video. This means that for more than one third of the music queries, the VM-Net was able to retrieve the exact corresponding video in its top 25 recommendations, out of 1,000 videos. Remember that all test samples (queries and database) are taken from MVD, a different dataset than the HIMV-50K used for training. Finally, we observe that the dimensionality d of the audio feature vector does not seem to have any impact on the recommendation performance. Indeed, the best performing features are the OpenL3 features ($d = 18,432$) while the second best performing features are the Audioset features ($d = 128$). In comparison, the original handcrafted features have a dimension of 1,140. However, d seems to have an impact on the training time. The VM-Net trained with the audio handcrafted features ($d = 1,140$) in 18 hours, the MuSimNet features ($d = 384$) in 18 hours as well, the MusiCNN features ($d = 600$) in 19 hours, the AudioSet features ($d = 128$) in 6 hours, and the OpenL3 features ($d = 18,432$) in 42 hours.

5.3.4 Results of the Experiment 3

Table 5.3 shows the results of Experiment 3, on the temporal aggregation method. Adding the automatic aggregation layers resulted in a noticeably increased training time, from 18 hours to 10 days. However, both of them seem to allow slightly better results, especially in terms of $R@10$ and $R@25$. The Attention system, which outperforms the Auto-pooling one, achieves a $Rank = 105$ for video-to-music. This means that the VM-Net was able to retrieve the exact corresponding music at rank 105 on average, out of 1,000 videos. This experiment confirms our intuition that the temporal variations of the music and video matter, and that all frames do not have the same importance for recommendation.

5.4 Summary

In this work, we studied Video/Music cross-modal recommendation. We built upon a recent V/M retrieval system named VM-Net, which originally relies on an audio representation obtained by a set of statistics computed over handcrafted features. We validated the Triplet Loss design choice from Hong, Im, and Yang, 2018. We demonstrated that using feature learning, especially the audio embeddings provided by the pre-trained OpenL3 network, allows to largely improve the obtained recommendations. This may be explained by the fact that OpenL3 features were trained on a *task similar to ours*, on a *large dataset* of *musical videos*. The other audio features we considered (MuSimNet, MusiCNN and AudioSet) do not present these three characteristics simultaneously. Please note that this is consistent with the results obtained by Grollmisch et al., 2020 on several audio tasks. Finally, we showed that switching the statistical aggregation of the features to a learned aggregation (Auto-pooling or Attention layer) allows improving further the recommendations.

Chapter 6

Is there a "Language of Music-Video Clips" ? A Qualitative and Quantitative Study

In Chapter 5, we showed that it was possible to directly recommend music tracks for videos based on their respective content. While using OpenL3 features and the Attention layer enhanced the recommendations, there is still room for improvement. The main restriction of the VM-Net is that it performs the recommendation at the full clip level, without considering the sequential evolution of music and video events. In order to design a relevant music supervision system, we need to go beyond global content recommendation and consider synchronization-aware recommendations. To do so, we first need to collect domain knowledge about which temporal considerations should be integrated into such a system, and how. In this chapter, we attempt at bridging this knowledge gap by performing a fine-grained analysis of how the synchronization of music and video is usually performed. We hypothesize that a better understanding of professionally produced music videos and an analysis of their "language" will help design better models for Video/Music recommendation.

Temporal structure (at the beat, bar or functional segment level) is one of the distinctive characteristics of music. For this reason, its automatic estimation has received a lot of attention in the MIR community (Downie, 2005). Temporal structure in video (cuts, scenes, chapters) has similarly received a lot of attention in the computer vision community, for example with the goal of creating video summaries or automatic video chaptering (Aner and Kender, 2002; George Awad et al., 2021). Our cross-modal analysis will be using these structural elements.

Similar analyses could be performed on any type of video that features a musical soundtrack - e.g. commercials, movies, performance recordings (Antoniadis and Bevilacqua, 2016). We focus here on the special case of Official Music Videos (OMVs). We call OMV an audiovisual document where the audio part consists of a music track, and which aims at promoting said track and its performing artists. As a result, the music track is generally the only source of audio in OMVs. This makes OMVs good prototypes for a study on music-video synchronization. We do not consider user-generated videos, because we assume that analyzing professionally produced OMVs is more likely to provide reusable insights.

In the specific case of OMVs, the editing team will often arrange the video rushes based on the structure of the music track (Schindler and Rauber, 2016). In some cases, the music track can also be adapted from the studio version for narrative purposes. Therefore, music and video structure are de facto associated. However, the level of synchronicity is not always the same, depending on the considered OMV.

This is not only due to artistic choices but also depends on the music genre and video genre, as we will see in our study.

Chapter contributions: In this chapter, we study the relationship between music and video temporal organization using a qualitative and quantitative approach. Case studies on specific music videos have already been conducted from a musicological perspective (Burns and Hawkins, 2019). We propose here to scale this type of analysis on a larger corpus, and we correlate the V/M structure and V/M genres. Our results were presented at the ISMIR conference (Prétet, Richard, and Peeters, 2021b).

1. The *qualitative* study is based on a set of interviews with three renowned specialists of Official Music Videos. We interview them in order to find out if and how they consider the relationship between music and video structure in their work.

2. The *quantitative* study is based on a detailed analysis of music and video structural events in OMV using MIR and computer vision tools. The OMV dataset we use corresponds to a subset of the Harmonix dataset (Nieto et al., 2019). We study specifically the relationship between the duration of music and video segments and between the positions of their respective boundaries. We highlight the dependency of those according to the OMV music and video genre (for which we annotated the data).

Chapter organization. This chapter is organized as follows. Section 6.1 describes the qualitative study and summarizes the interviews of three music video experts: Jack Bartman (composer), Alexandre Courtès (director) and Maxime Pozzi (editor). Section 6.2 describes the quantitative study: the dataset creation (6.2.2), the analysis of the music and video segment duration and position (6.2.3). Section 6.3 summarizes this work.

6.1 Qualitative analysis: Interviews

6.1.1 Methodology

In order to gather intuition on the synchronization of music and video, we conducted a series of semi-structured face-to-face interviews. We selected three music video experts from different professions: composition, direction and editing. Following Inskip, Macfarlane, and Rafferty, 2008, we selected the respondents using a snowball sampling technique.

Interviews were performed using the Zoom video conferencing software, lasting up to one and a half hours. The interviews were transcribed manually by the researcher, and transcripts were sent back to the respondents for validation. All interviews were initially performed in French and then translated into English by the researchers. Areas of discussion included the participant's day-to-day workflow and technical tools, their interactions with the other professions of the industry, and their opinion on example music videos prepared by the researcher. Transcriptions for each interview can be found in Appendix D.

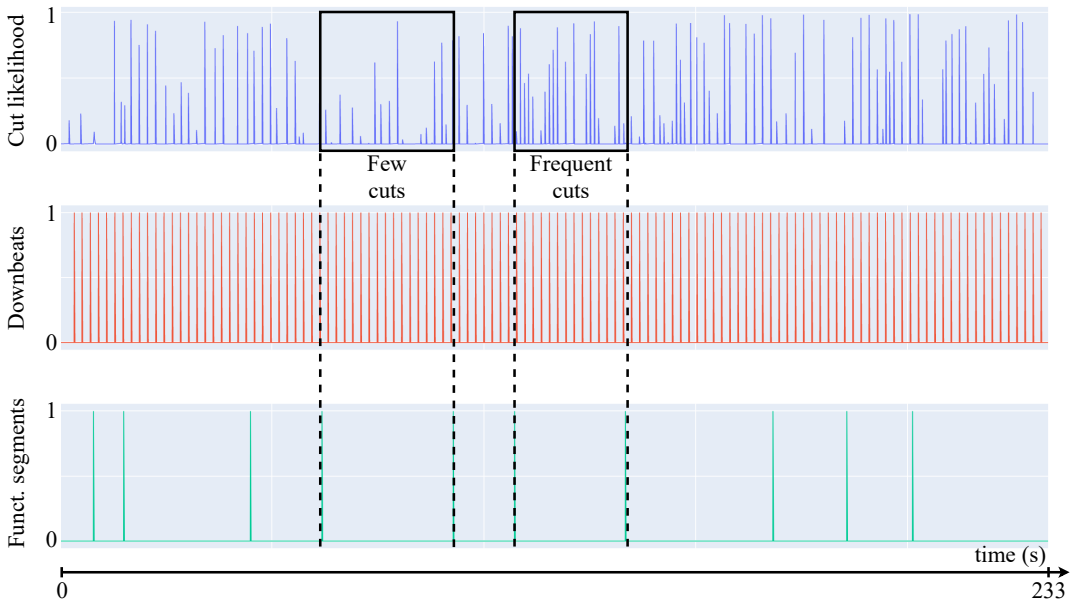


FIGURE 6.1: Audiovisual structure of Katy Perry, *Firework*, full clip. Horizontal axis: time. Cuts: TransNet estimates. Downbeats: Madmom estimates. Music functional segments: OLDA estimates.

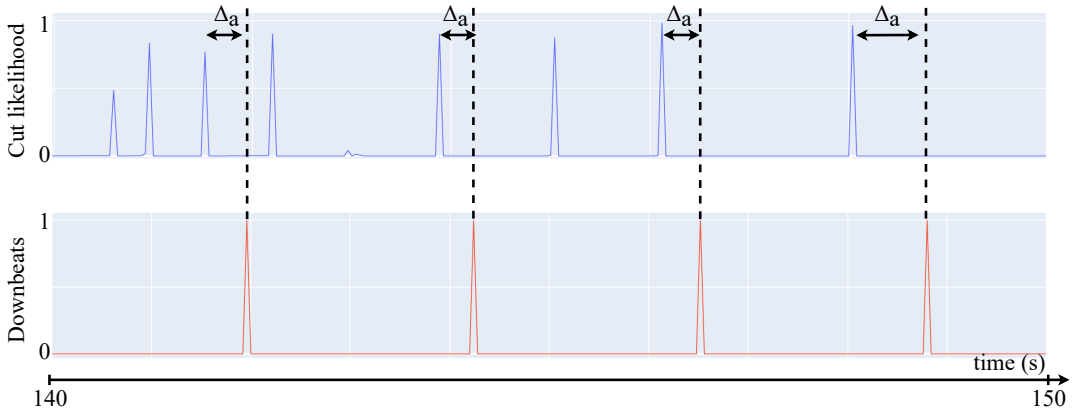


FIGURE 6.2: Audiovisual structure of Adele, *Rolling in the deep*, timestamps 02:20 to 02:30. Horizontal axis: time. Cuts: TransNet estimates. Downbeats: Madmom estimates. Δ_a: anticipation of cuts with respect to downbeat.

6.1.2 Interviews

Jack Bartman, Composer

As a composer (for commercials such as Nike, Apple or UbiSoft), Bartman has to adopt both a global and a precise local approach: the content of the music has to match the visual atmosphere, and its temporal structure must be aligned both globally at clip level and locally at frame level. In some cases, the video editing follows the structure of the music. But in other cases, typically for advertisement, it is the opposite, and the composer has to adapt the music to an existing movie. Most of the time, when music has to be edited, the slicing operation is privileged.

> "Slicing can happen at unconventional moments, like the first or last beat of a bar! I simply add a sound effect to make it work."

Time stretching and accelerations can be employed too, but are far less usual. Bartman stresses that synchronizing cuts to audio events is especially important around the emotional climaxes of the video. Finally, for some projects, an exact synchronization is not the golden rule:

> This year, I worked on a short movie about psychological aspects of the Covid-19 lockdown. After getting used to an imperfectly synchronized mockup soundtrack, the director did not want to use the final version, as the mockup would better suit the intended *madness* atmosphere.

Alexandre Courtès, Director

As a director (such as for U2, Phoenix, Cassius, Franz Ferdinand or Jamiroquai), Courtès generally has a lot of freedom when it comes to the temporal organization of a music video. Directors often come up with their own concept and they have little constraint about the content of the video. At a larger temporal scale, their mission is to emphasize the music climaxes by the appropriate video content.

> "The music video will often show a performance, so it is similar to a musical comedy: it has to feature costumes, chapters, sets, acts."

Directors are not responsible for placing the cuts, but they can introduce diversity in the video transitions (using for example explosions, large objects passing in front of the camera, etc.)[1].

> "Cuts have to follow the music's rhythm, even though they might not always co-occur with beats."

Maxime Pozzi, Editor

As an editor (such as for Rihanna, Taylor Swift, Foals or Woodkid), Pozzi has to combine both a local, frame-level approach to the design of a global emotional trajectory.

> "Editors and musicians have a similar job, we all want the same thing: rhythm, narration, climaxes."

[1] Schindler and Rauber, 2019 explain how shot transitions are used in a complex and artistic way in music videos. They provide an extensive list of effects and transition types that are rare in cinematic productions but common in OMVs.

For chorus and verses, the editing will follow the rhythm and typically accelerate near climaxes. During bridges, it will often be slower and poetic. This can be illustrated for example by Katy Perry's *Firework* music video (Figure 6.1). In this clip, we can see some functional segments where cuts happen very frequently (several times in each bar) and segments where they happen less frequently, for example on the downbeats only.

Editing can be used as an element of narration. For example, in Adele's *Rolling in the deep* music video, starting at timestamp 02:20, the cuts are systematically placed on the backbeat (just before the downbeat) (see Figure 6.2).

> "Off-beat cuts are used to create dynamics: to surprise the viewer and illustrate the music's emotional climax. It makes the video direction appear more "indie" as well, this can be required by the performing artists."

6.1.3 Summary

These three interviews provide us with a series of intuitions and hypotheses about the way audio and video are synchronized in music videos. First, musical structure such as choruses and verses are taken into account when directing a music video. Second, music events such as rhythm, beat and downbeat are taken into account when editing a music video. Finally, according to the desired atmosphere, the music and video structural events can be more or less perfectly synchronized.

6.2 Quantitative Analysis

6.2.1 Methodology

In the following, we conduct a set of quantitative experiments on how the Structural Events (SE) of the music and of the video are synchronized in time. We therefore first collect a dataset of OMVs, along with music and video genre annotations (Section 6.2.2). For each of them, we use MIR tools to estimate music SE and computer vision tools to estimate video SE. More specifically, we estimate the downbeat positions, functional segments and shot boundaries from the OMVs of our dataset. In our first experiment, we study the correlation between the duration of the shots and the various musical SEs (beat and bar duration). In our second experiment, we study the temporal co-occurrence of the shot boundaries and the various musical SEs (bar and functional segment boundaries). We analyze the results of those for each music genre and each video genre.

6.2.2 Dataset

For our quantitative study, we consider a subset of 548 OMVs from the Harmonix dataset (Nieto et al., 2019), described in Section 3.2.3.

Annotations into structural events

By analyzing OMVs, Schindler and Rauber, 2016 observe that the music videos present characteristic editing styles (number of shots per second, types of transition) for certain music genres or moods. But they do not quantify this correlation. Here, we consider two types of Structural Events (SE): those based on the music content -audio-, and those based on video content -image frames over time (see Figure 6.3).

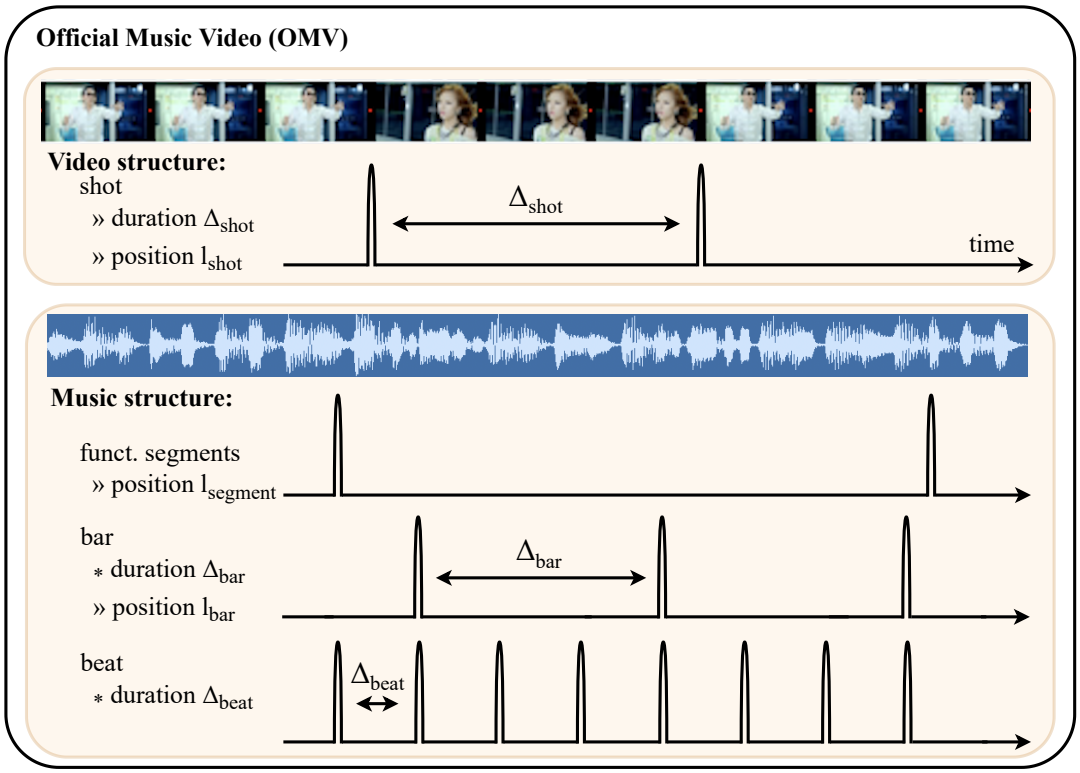


FIGURE 6.3: Schematic view of the different audiovisual structural events considered: shots (Δ_{shot}, l_{shot}), functional music segments ($l_{segment}$), bars/downbeats (Δ_{bar}, l_{bar}) and beats (Δ_{beat}, l_{beat}). Harmonix annotations are denoted by a $*$ symbol; computer vision and MIR annotations by a » symbol. Illustration music video: Psy, *Gangnam Style*.

TABLE 6.1: Summary of the annotations used for the quantitative analysis.

Music		
genre		Harmonix annotations
funct. segments positions	$l_{segment}$	OLDA/MSAF
bar duration	Δ_{bar}	Harmonix annotations
bar/downbeat positions	l_{bar}	Madmom
beat duration	Δ_{beat}	Harmonix annotations
Video		
genre		Manual annotations
shot boundary probability	$f_{shot}(t)$	TransNet
shot boundary positions	l_{shot}	TransNet
most common shot duration	$\Delta_{shot}^{\max}$	

Music SE. We consider three types of music SEs. At the smallest time scale, we consider the beats and downbeats; at the largest temporal scale, we consider the functional music segment boundaries (between the intro, verses, bridges, choruses). Harmonix features a set of manual annotations into functional segments, downbeat and beat. However, these annotations correspond to studio versions of the tracks which can, in some cases, be largely different from the version used in the OMV. For this reason, we only used the annotations into bpm and meter of the Harmonix dataset to get the beat duration $\Delta_{beat} = \frac{60}{bpm}$ and bar duration Δ_{bar} (which is computed as a multiple of the beat duration Δ_{beat} using the provided time signature). The downbeat positions and functional segment boundaries had to be estimated from the music tracks. We used two open-source algorithms, which we described in Section 2.1.3. For the downbeat positions, we used the algorithm of Böck, Krebs, and Widmer, 2016, implemented in the Madmom library (Böck et al., 2016). We ran the Madmom algorithm on each music track and we kept only the first beat of each bar to obtain the downbeat positions. In the following, we denote by l_{bar} the list of downbeat positions for a given track. For the functional music segments, we used the implementation of OLDA from the MSAF library (Nieto and Bello, 2016). In the following, we denote by $l_{segment}$ the list of boundary positions between the segments for a given track. For our dataset, the average duration of functional music segments is 19.73 s. and the average bar duration is 2.30 s.

Video SE. We consider only the least ambiguous video SE, the shot boundaries (or cuts). To estimate the position of boundaries between shots, we used the TransNet system (Souček, Moravec, and Lokoč, 2019), which we introduced in Section 2.1.3. The associated library is freely available on GitHub[2]. The TransNet output is a continuous function of time $f_{shot}(t) \in [0, 1]$ representing the likelihood of a boundary at time t. f_{shot} has a sampling rate of 25 Hz.

Also, for each OMV, we compute the histogram of its shot duration. We do so by first estimating the list of shot boundary positions l_{shot} by thresholding $f_{shot}(t)$ with $\tau = 0.5$. The resulting shots have an average duration Δ_{shot} of 4.76s. We then compute the histogram of these durations. We denote by $\Delta_{shot}^{\max}$ the position of the maximum of this histogram (in seconds).

We sum up the various SE notations in Table 6.1.

[2] https://github.com/soCzech/TransNet

Annotations into genre

We consider both the genre associated to the music and the one associated to the video.

Music genre. While still controversial in its exact definition (Hennequin, Royo-Letelier, and Moussallam, 2018), music genre is an effective way to describe musical content. For this reason, it has been and is still a widely studied topic. It has dedicated challenges (Defferrard et al., 2018), and large datasets featuring hundreds of categories (Bertin-Mahieux et al., 2011; Gemmeke et al., 2017; Defferrard et al., 2017). For our experiments, we use the music genre annotations provided by the Harmonix dataset metadata.

Video genre. Video genre classification is a much less studied topic. Existing studies focus on small sets of video genres (You, Liu, and Perkis, 2010; Varghese and Ramachandran Nair, 2019; Choroś, 2018). Only Gillet, Essid, and Richard, 2007 and Schindler, 2019 studied the case of OMVs and there is no consensus on the taxonomy of OMV genres. There is also no annotated dataset for this task.

We merge [Gillet, Essid, and Richard, 2007] and [Schindler, 2019] to obtain a set of 5 video categories and a corresponding single-label dataset. Maxime Pozzi, a professional music video editor, validated our taxonomy during our preliminary interview (see the verbatim in Appendix D.3). We then manually annotated all 548 video clips of Harmonix into the five following video genres:

- `Performance videos (P)`: The artist or band are presented performing the song. 74 videos; example: Iron Maiden, *Powerslave*.
- `Concept/Abstract videos (C)`: The video illustrates the music metaphorically via a series of abstract shots related to the semantics or atmosphere of the song. 227 videos; example: Lady Gaga, *Poker Face*.
- `Narrative videos (N)`: The music video has strong narrative content, with identifiable characters and an explicit chronology. 160 videos; example: Taylor Swift, *Teardrops on My Guitar*.
- `Dance videos (D)`: Artists present a rehearsed dance choreography in sync with the music. 62 videos; examples: Sean Paul, *Get Busy*.
- `Other (O)`: Other types of music videos, including lyrics videos, animated music videos, etc. 25 videos; example: Train, *Hey, Soul Sister*.

Correlation of music and video genres. As can be seen on Figure 6.4, the video genre is usually correlated to the music genre in our dataset. For example, Reggaeton music is very likely to be illustrated by Dance videos, while Hip Hop music tends to be accompanied by Conceptual visuals. Country music favors the Narration format.

6.2.3 Experiments

We hypothesize that the music structural events play an important role in the placement of cuts during video editing. We check this assumption by measuring:

- if their segment duration are correlated in Section 6.2.3;
- if their position co-occur in Section 6.2.3.

According to Gillet, Essid, and Richard, 2007, the performance of alignment-based music-video recommendation systems is strongly correlated to the video genre. We therefore differentiate our results by music and video genre.

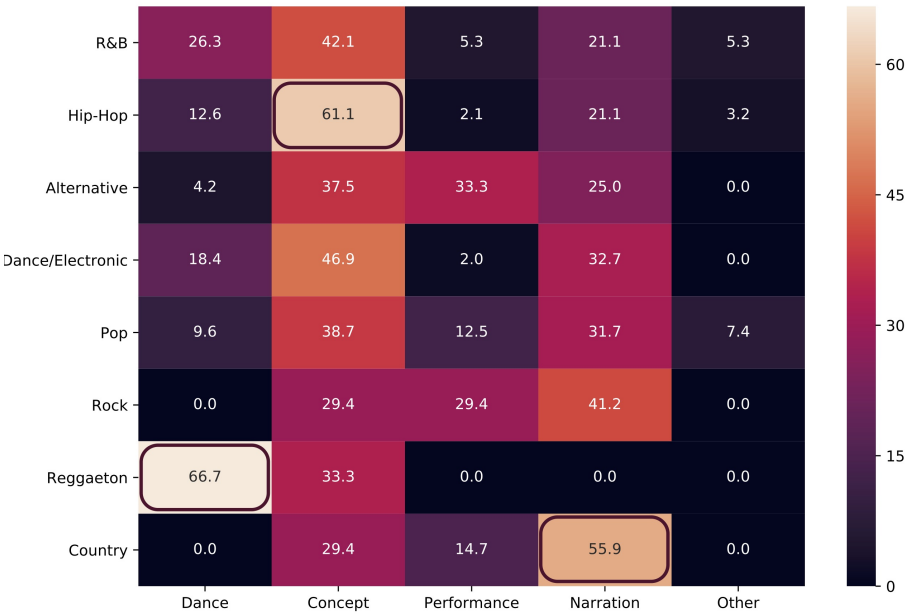


FIGURE 6.4: Distribution of video genres per music genre (as the percentage of all clips for each music genre).

Comparison between events duration

Our first experiment aims at evaluating to which extent the musical and video segments have similar durations.

To measure this, we compare $\Delta_{shot}^{\max}$ (the most common shot duration) with the beat duration Δ_{beat} and bar duration Δ_{bar} obtained from the Harmonix annotations. When $\Delta_{shot}^{\max}$ is close to Δ_{bar}, this indicates that a systematic change of shots occurs with the same period as the bar changes. This however does not mean that the changes occur simultaneously (we study this in Section 6.2.3).

This is for example the case of "Heartless" by Kanye West (see Figure 6.5 [top]) where the large peak at $\Delta_{shot}^{\max}$=2.72 s can be explained by the tempo at 88 bpm; or "Firework" by Katy Perry (see Figure 6.5 [bottom]) where the large peak at $\Delta_{shot}^{\max}$= 1.93 s can be explained by the tempo at 124 bpm.

In our dataset, a synchronization at the bar level ($0.5\Delta_{bar} < \Delta_{shot}^{\max} < 1.5\Delta_{bar}$) occurs for one fifth of the clips (95 music videos). Synchronization may also occur at other levels: at the beat level Δ_{beat}, or the pattern level $\Delta_{pattern}$ (usually an even multiple of the bar duration). In our dataset, a synchronization at the beat level ($0.5\Delta_{beat} < \Delta_{shot}^{\max} < 1.5\Delta_{beat}$) occurs for two thirds of the clips (329 music videos). However, synchronization at pattern level $\Delta_{pattern} = 4\Delta_{bar}$ almost never occurs (2 music videos).

In Table 6.2, we indicate for each music genre and video genre, the number of tracks for which the $\Delta_{shot}^{\max}$ correspond to Δ_{bar} or Δ_{beat}. The Harmonix Set proposes a taxonomy of 20 genres, but some of these were strongly under-represented in our subset (e.g. `Classic Rock`, `Word`, `Prog`). We only focus here on the most represented genres, i.e. which appear at least 10 times. We observe a strong correspondence between $\Delta_{shot}^{\max}$ and Δ_{bar} for the music genres `Country`, `Dance/Electro` and `Rock` (one

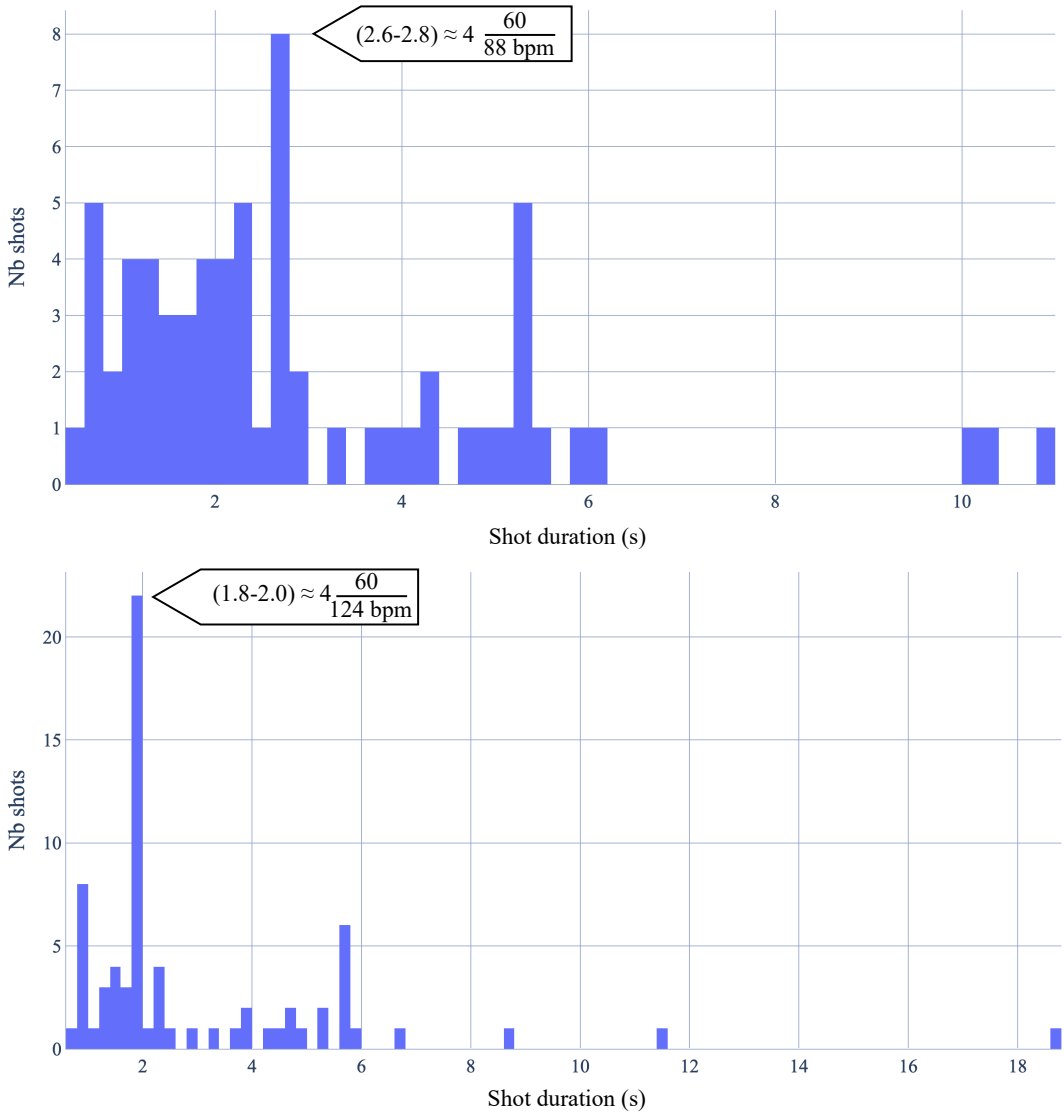


FIGURE 6.5: [top] Histogram of shot duration in the music video of *Heartless* by Kanye West. The tempo is 88 bpm. [bottom] Histogram of shot duration in the music video of *Firework* by Katy Perry. The tempo is 124 bpm.

TABLE 6.2: Agreement of musical segment duration (bar Δ_{bar} and beat Δ_{beat}) and dominant shot duration $\Delta_{shot}^{\max}$ according to the **music genre** [top table] and according to the **video genre** [bottom table]. Highest values are highlighted in bold, lowest values in italic.

	$\Delta_{shot}^{\max} \simeq \Delta_{bar}$		$\Delta_{shot}^{\max} \simeq \Delta_{beat}$	
Music Genre	# tracks	%	# tracks	%
Alternative	2	8.3	19	**79.2**
Country	10	**29.4**	16	47.1
Dance/Electro	12	**24.5**	28	57.1
Hip-Hop	12	12.6	69	72.6
Pop	40	14.8	158	*58.3*
R&B	1	*5.3*	13	68.4
Reggaeton	1	8.3	9	**75.0**
Rock	4	**23.5**	10	*58.8*

	$\Delta_{shot}^{\max} \simeq \Delta_{bar}$		$\Delta_{shot}^{\max} \simeq \Delta_{beat}$	
Video Genre	# tracks	%	# tracks	%
Concept	33	14.5	148	**65.2**
Dance	11	17.7	40	**64.5**
Narration	28	17.5	93	58.1
Performance	19	**25.7**	41	55.4
Other	4	*16.0*	8	*32.0*

fourth of the tracks). We observe a strong correspondence between $\Delta_{shot}^{\max}$ and Δ_{beat} for the music genres `Alternative` and `Reggaeton` (three quarters of the tracks). This may imply, for example, that music video professionals favor more dynamic editing styles (using shorter shots on average) for `Reggaton` than for `Country` music. We observe a strong correspondence between $\Delta_{shot}^{\max}$ and Δ_{bar} for the video genre `Performance` (one fourth of the tracks). On the contrary, we observe a low correspondence between $\Delta_{shot}^{\max}$ and Δ_{beat} for the video genre `Other` (one third of the tracks). It is likely that music videos in the `Other` category favor experimental editing styles, with shots of more diverse duration.

As we see, there is a strong relationship between the video and musical events duration. This however does not mean that the changes occur simultaneously. We study this in the next paragraph.

Comparison between events position

Our second experiment aims at evaluating to which extent the musical events l_{seg}, l_{bar} and video events $f_{shot}(t)$ happen simultaneously. To measure this, we compute for each music boundary i ($t_i \in l_{seg}$ or $t_i \in l_{bar}$) a score $S_i \in [0,1]$. S_i is defined as the integral over time of the shot boundary likelihood $f_{shot}(t)$ tampered by a non-normalized Gaussian window $w(t)$. $w(t)$ is centered on t_i, with $\sigma = 2$ (such that the effective duration of the window is approximately 0.5s at a frame rate of 25Hz) and with $w(0) = 1$.

$$S_i = \int_t w(t - t_i) f_{shot}(t) dt, \quad \forall t_i \in \{l_{seg}, l_{bar}\} \tag{6.1}$$

A large value of S_i indicates that the t_i position (the music structural event) corresponds to a large probability of shot boundary. We then average S_i for all music

TABLE 6.3: Shot transition intensity S around music boundaries (functional segments boundaries l_{seg} and bar boundaries l_{bar}) according to the **music genre** [top table] and according to the **video genre** [bottom table]. Mean values and confidence intervals at 95% are displayed. Highest values are highlighted in bold, lowest values in italic.

Music Genre	$S(l_{seg})$	$S(l_{bar})$	# tracks
Alternative	0.22 ± 0.08	0.23 ± 0.02	24
Country	0.20 ± 0.06	0.21 ± 0.02	34
Dance/Electro	*0.18 ± 0.05*	0.21 ± 0.02	49
Hip-Hop	0.19 ± 0.03	0.25 ± 0.01	95
Pop	**0.36 ± 0.02**	0.21 ± 0.01	271
R&B	**0.29 ± 0.10**	**0.31 ± 0.03**	19
Reggaeton	0.24 ± 0.11	**0.28 ± 0.04**	12
Rock	*0.18 ± 0.07*	*0.19 ± 0.03*	17
Video Genre	$S(l_{seg})$	$S(l_{bar})$	# tracks
Concept	0.20 ± 0.02	**0.23 ± 0.01**	227
Dance	**0.18 ± 0.04**	**0.24 ± 0.01**	62
Narration	**0.18 ± 0.03**	**0.23 ± 0.01**	160
Performance	*0.15 ± 0.04*	0.16 ± 0.01	74
Other	*0.11 ± 0.06*	*0.11 ± 0.02*	25

boundaries i to get S. S might be considered as a measure of precision, since it provides information on *how many music boundaries are explained by a video boundary*. S is also close to the measure proposed by Cemgil et al., 2000 to evaluate the performances of beat-tracking algorithms. It should be noted that the number of video boundaries is larger than the number of music boundaries (as seen in Figures 6.1 and 6.2). A large value of S indicates that the shot boundaries are located at the same positions as the music structural events l_{seg} or l_{bar}. We compute S separately using the t_i from l_{seg} or from l_{bar}. To check if the amount of music-video event synchronization depends on the music and video genre, we average S over all tracks of a given genre (music or video).

Table 6.3 [top part] shows the co-occurrence scores S aggregated over music genres. We observe variations of the values of S according to the music genre. For `Pop`, $S(l_{seg})$ is large (0.36) indicating that many shot transitions occur at the functional segment boundaries positions. For `R&B` and `Reggaeton`, $S(l_{bar})$ is large (0.31 and 0.28) indicating that many shot transitions occur at the downbeat positions. We also observe that the values of $S(l_{seg})$ and $S(l_{bar})$ vary according to the music genre with very small values for `Dance/Electronic`, `Hip-Hop` and `Rock`. This comes as a surprise especially for `Dance/Electronic`, because in the previous experiment, we observed a strong correspondence between the duration of shots and bars for this music genre. This shows that even though bars and shots have similar duration, their boundaries might not always co-occur.

Table 6.3 [bottom part] shows the co-occurrence scores S aggregated over video genres. We observe variations of the values of S according to the video genre. We see that the `Dance` video genre has a large value of $S(l_{bar})$ (0.24), which is not surprising given that video labeled as `Dance` actually show people dancing on the beat. We also observe large values of $S(l_{bar})$ for the `Concept` and `Narration` video genres with consistent synchronization on the downbeats. For the `Performance` video genre (the band is playing in front of the camera), we do not observe such a large correspondence ($S(l_{bar}) = 0.16$). For the `Other` video genre, the low values

($S(l_{bar}) = S(l_{seg}) = 0.11$) are not surprising, given that some videos are very experimental and may feature complex video transitions, which may be difficult to detect by the TransNet.

6.3 Summary

In order to design an efficient recommendation system for music supervision, we aim at going beyond global content recommendation and at considering structure adequation as well. As a first step toward this improvement, we attempted to better define the structure of music videos. Experts and experiments support our assumption that structure is crucial for editing music videos.

According to the professionals, official music videos are edited by taking into account the music structure. Although some experts mentioned that synchronization was often a matter of taste and intuition, we were able to bring out some trends.

We analyzed the concordance of music and video structural boundaries and correlated those to the music and video genres. We showed that the co-occurrence of music and video structural events would vary according to the music and video genres. These elements can be reused to design or improve automatic V/M recommendation systems. For example, if the task is to recommend an illustration video for a Pop or R&B track, the system is expected to favor candidates that allow high synchronization of the structural events.

In the next chapter, we will see how we can improve our V/M recommendation system by taking into account the sequence of music and video segments.

Chapter 7

Music-Video Recommendation using a Temporal Alignment of Segments

In Chapter 5, we studied the VM-Net, a self-supervised neural network for Video / Music recommendation introduced by Hong, Im, and Yang, 2018. The VM-Net is able to learn content matching between video and music from a large amount of non-annotated music video clips. Compared to other similar works, this system has interesting specificities such as an extended triplet loss for video and music consistency and a lightweight architecture thanks to the use of high-level features as inputs. In Chapter 5, we showed that the performances of the original VM-Net could be largely improved upon by replacing the original handcrafted audio features with pre-trained music embeddings (Gemmeke et al., 2017). However, another important limitation on the original approach remains; the whole duration of the video and of the music are summed up as timeless embedding vectors. As a result, the temporal relationship between the music and video representation vectors is not exploited. Like other V/M recommendation systems based on a matching by semantic association (Arandjelovic and Zisserman, 2017), the VM-Net does not allow for representing variations of content over time.

This is in contradiction with the way music video clips and music tracks are constructed (as a sequence of shots and verse-chorus segments). Our study on music video clips in Chapter 6 actually confirmed the intuition that a high-performing V/M recommendation system should take into consideration the structure of the music tracks and query videos. As seen in Section 2.2.4, some systems use the music structure for music video generation (Wang et al., 2007; Wang, Chng, and Xu, 2006; Hua, Lu, and Zhang, 2004). In our case however, the target music track is not known a priori, and has to be recommended. This raises several questions: How can we define "structure" in musical videos? Which temporal scale should we consider to model the temporal evolution? How can we train the network at shorter time scales? What is the best way to evaluate it?

In this chapter, we introduce a new way to perform cross-modal recommendation of music tracks to be used as soundtracks for videos. Inspired by the music supervision use case and domain knowledge, we propose an improved system that takes into account the music and video structure. We assume the following constraints for an efficient music recommendation system:

- querying should be fast enough to scale to large catalogs;
- the video query cannot be edited while the music tracks can (only to some extent: typically by slicing them into meaningful segments, see Section 6.1.2).

Our idea relies on the fact that embedding vectors can not only represent the full AV clip but also a segment thereof, depending on the level of aggregation. We slice the music and video clips into semantic segments, then train the VM-Net *at segment level*, and finally perform the ranking of the music tracks according to a segment alignment cost. Compared to other V/M matching systems based on sequence comparison (Lin, Wei, and Wang, 2015; Wang et al., 2012; Lin et al., 2017; Shin and Lee, 2017), we do not employ emotion annotations. We maintain the Self Supervised Learning metric learning paradigm to take advantage of the large (but unannotated) HIMV-50K training dataset, and we do not restrict ourselves to recommending music tracks that have similar emotional content as the video. Finally, we take into account the domain-specific constraints to propose evaluation scenarios that are more realistic from a music supervision point of view.

Chapter contributions: We introduce the Seg-VM-Net, a self-supervised Video-to-Music (V2M) retrieval system that makes use of a structure-aware segmentation of the clips. We divide each music video clip into segments, train a multimodal neural network to obtain one representation per segment in a joint embedding space, and compute alignment scores between sequences of segments to perform the ranking of the music tracks. An overview of the proposed system is given on Figure 7.2. We show that this approach significantly improves performance and robustness compared to previous clip-level approaches. To our knowledge, this is the first Video-to-Music recommendation system that combines the three following characteristics:

1. a self-supervised multimodal embedding space dedicated to Video/Music recommendation;
2. a Video-to-Music recommendation performed via a dynamic time warping alignment cost between two sequences of segments;
3. a semantic definition of segments, capable of handling units of variable duration.

The resulting system, the Seg-VM-Net, is the subject of a patent application (Prétet et al., 2021) and a journal paper (Prétet et al., 2022).

Chapter organization. The extension of the VM-Net from a clip-level to a segment-level recommendation system (the Seg-VM-Net) is exposed in Section 7.1. In Section 7.2, we propose a way to evaluate the segment-level system. In Section 7.3, we demonstrate with a series of experiments, how we can improve upon the performance of the VM-Net and how we make it more suitable for the music supervision task. Lastly, Section 7.4 concludes our study.

7.1 Proposed Video-to-Music recommendation system

In this study, we consider the VM-Net as our baseline recommendation system. We introduced the VM-Net in Section 5.1, and the relevant datasets (HIMV-50K for training and MVD for evaluation) in Chapter 3. One of the drawbacks of the original VM-Net system is the lack of considerations of the temporal evolution of the music and video modalities. For each AV pair $c = (c^v, c^m)$, the VM-Net sums up both modalities as timeless vectors $e_c = (e_c^m, e_c^v)$. This accounts for a massive information loss, given that the average duration of an AV clip in the training dataset HIMV-50K

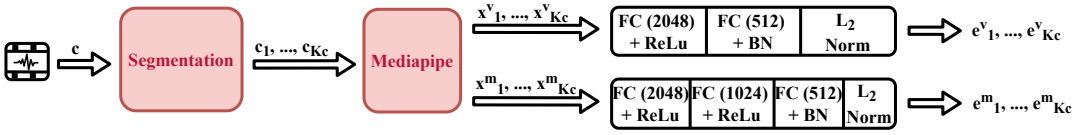


FIGURE 7.1: General overview of the studied Seg-VM-Net recommendation system (training perspective). "FC" stands for Fully-Connected and "BN" for Batch Normalization. We highlight in red the differences with respect to the original VM-Net, as described by Hong, Im, and Yang, 2018.

is 3 min 54s. In this work, we introduce the Seg-VM-Net, an extension of the VM-Net which takes into account this temporal evolution.

While it is possible to consider the evolution at the frame level (for example 1 frame = 1 second), in this work, for computational reasons, we consider the evolution at the *segment* level. We therefore consider a AV clip c as a temporal succession of non-overlapping segments $c = \{c_1, \ldots, c_{K_c}\}$, as illustrated on Figure 7.1. Each segment c_k consists in a pair of a music and a video segment: $c = \{(c_1^m, c_1^v) \ldots, (c_{K_c}^m, c_{K_c}^v)\}$. In this work, segments represent continuous temporal regions where the content is considered homogeneous either from the video content viewpoint or from the audio content viewpoint. For example, segments can be defined as the sequence of shots in the video part c^v or the sequence of verse/chorus/bridge parts in the music part c^m. This naturally leads to several possible definitions of such a segmentation which we will detail in the following.

7.1.1 Input music and video features x^m, x^v

The original VM-Net uses handcrafted audio features, computed with Librosa (McFee et al., 2015). In Section 5.2.2, we showed that this choice was sub-optimal considering recent advances in feature learning. Therefore, we replace the original handcrafted audio features of the VM-Net by the AudioSet pre-trained music embeddings, as illustrated on Figure 7.1. As a reminder, the AudioSet deep feature extractor model is a VGG-inspired autotagger pre-trained on a preliminary version of the YouTube-8M dataset. We select the AudioSet features because they perform better in the V2M scenario, compared to the OpenL3 features (see Table 5.2). For the video, we keep the originally proposed ImageNet features, described in Section 5.1.1.

Implementation. We compute the music and video features using the Mediapipe library (Gemmeke et al., 2017). We compute the frame-level features at a rate of 1Hz and average those over time, resulting in one single timeless vector for each segment. We do not train the Mediapipe models with the Seg-VM-Net; instead we use them as frozen feature extractors. Indeed, we do not have access to datasets as large as the one used to train them, and therefore we doubt we could match their performance. We do not finetune the feature extractors either, for more efficient use of computation resources.

7.1.2 Segmenting AV clips

As mentioned above, the original VM-Net considers a AV clip c as a pair of timeless vectors (x_c^m, x_c^v). In our work, we consider c as a temporal succession of non-overlapping temporal segments $c = \{c_1, \ldots, c_{K_c}\}$, each segment being a pair of a music and a video segment: $c = \{(c_1^m, c_1^v) \ldots, (c_{K_c}^m, c_{K_c}^v)\}$.

We define the *segments* as excerpts of AV clips that are homogeneous according to one of the two modalities. Such segmentation can therefore be obtained in various ways: for example using the sequence of shots in the video part c^v or the sequence of verse/chorus/bridge parts in the music part c^m. Segments can therefore have variable duration.

We denote by S a specific method to obtain such a segmentation[1]. Once S is applied to one modality, we make the hypothesis that we can use the same segment boundaries for the other modality without severe loss of homogeneity[2]. Since our training and test datasets are not annotated in music nor video structure, S is defined by a segmentation algorithm either based on the music or the video content. We experiment with four different segmentation algorithms S: three music structure estimation methods and one video shot estimation method. We describe each method ($S_{Foote}, S_{SF}, S_{OLDA}$ and $S_{TransNet}$) with more details in Section 2.1.3.

S_{Foote}**:** We estimate the music segment boundaries using the classic novelty-based approach introduced by Foote, 2000. Homogeneity is the main criteria to define segments.

S_{SF}**:** We estimate the music segment boundaries using the *Structural Features* (SF) algorithm introduced by Serra et al., 2012, in which repetition is taken into account.

S_{OLDA}**:** We estimate the music structure using the supervised *Ordinal Linear Discriminant Analysis* (OLDA) method introduced by McFee and Ellis, 2014. Then, the obtained features are clustered by merging similar successive segments.

$S_{TransNet}$**:** We estimate the shot transitions using the supervised TransNet CNN introduced by Souček, Moravec, and Lokoč, 2019.

Implementation. For the three music structure estimation algorithms (Foote, SF and OLDA), we use the implementation available in the open-source music segmentation framework MSAF (Nieto and Bello, 2016). The resulting music segments last on average 20 seconds. For the visual shot segmentation, the open-source Python library of the TransNet shot detector was installed according to the instructions[3]. The average duration of the shots in our dataset is 6.6 seconds.

For each segmentation algorithm S, we segmented the AV clip by considering only one modality and then applied the obtained segment boundaries to the other modality. Since the segmentations differ according to the choice of S, we trained the Seg-VM-Net for each choice of S using the corresponding segmented version of HIMV-50K. In Experiment 1 (Section 7.3.1) and Experiment 2 (Section 7.3.2), we compare the performances obtained for recommendation using the different segmentation approaches S.

7.1.3 Training using segments

For a given choice of S, each AV clip c is represented as a sequence of AV segments: $c \xrightarrow{S} \{c_1, \dots c_{K_c}\}$. For each segment c_k we compute the segment-level input feature vectors $x^m_{c_k}$ and $x^v_{c_k}$ by aggregating the frame-level music and video features over the duration of the segment c_k using their mean values.

[1] This should not be confused with the $\mathcal{S}$ function used as ground truth for music similarity in Chapter 4.

[2] This is because in the HIMV-50K dataset, the videos are often edited following the music tracks.

[3] https://github.com/soCzech/TransNet

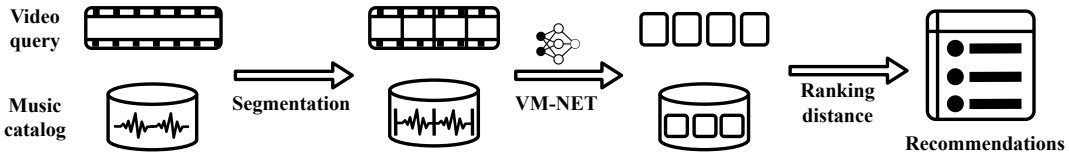


FIGURE 7.2: General overview of the studied Seg-VM-Net recommendation system (inference perspective).

For each AV clip c, we obtain a pair of segment-level feature vectors: $x_c = \{(x^m_{c,1}, x^v_{c,1}) \ldots, (x^m_{c,K_c}, x^v_{c,K_c})\}$. We then train the Seg-VM-Net using the $\mathcal{L}_{VMNET}$ triplet loss (see Section 5.1) but matching segments rather than matching clips. This is achieved using the following triplet mining strategy that we explain for $\mathcal{L}_{VM}$. For a given video anchor a chosen as $x^v_{i,l}$ (where i denotes the clip and l the segment number):

- the positive sample p can only be the corresponding music segment from the same AV clip (same timestamp as the anchor): $x^m_{i,l}$
- the negative samples n can only be sampled from other music tracks at any timestamp: $x^m_{j\neq i,*}$

Note that we experimented with other mining strategies (such that allowing negatives n to be sampled from the same AV clip but at different time-stamps: $n = x^m_{i,k\neq l}$) but none had a positive impact on performance. The same mining strategy is used for $\mathcal{L}_{MV}$, with $(a, p, n) = (x^m_{i,l}, x^v_{i,l}, x^v_{j\neq i,*})$. Compared to Hong, Im, and Yang, 2018, we choose not to use intra-modal structure constraints, as preliminary experiments showed they were not beneficial in our setup. We also select the hyperparameters $\lambda_1 = \lambda_2 = 1$ and $\alpha = 0.1$.

Given that each choice of S leads to different segmentations of the HIMV-50K training set, we train a different projection function f_S for each S. Since we represent each segment as a pair of timeless feature vectors, we can use the exact same training procedure as described in Section 5.1.

7.1.4 Video/Music recommendation using segments

To perform Video/Music recommendation using our segment-level system, we first segment the query q using the same segmentation algorithm S as for training. We then project each segment through the corresponding f_S to get the embedding vectors: $e^m_{q,k}$ and $e^v_{q,k}$ with $k \in \{1, \ldots K_q\}$. Each AV clip is thus represented as a sequence of pairs of embedding vectors: $e_c = \{(e^m_{q,1}, e^v_{q,1}) \ldots, (e^m_{q,K_q}, e^v_{q,K_q})\}$.

To find the most relevant music tracks for a given video query, we rank the music tracks based on a distance δ between the two sets of embedding vectors: one representing the video query $e^v_{q,*}$, the other representing the candidate music track of the catalog $e^m_{c,*}$ with $c \in \{1, \ldots, C\}$ where C is the size of the catalog. To do so, we propose and test two families of methods: one based on cluster aggregation strategies (as used in hierarchical clustering), and one based on alignment costs. We illustrate the process on Figure 7.2.

Cluster aggregation. We propose and test three simple ranking distances δ considering two unordered sets of segments. We illustrate these different methods in Figure 7.3, and discuss their respective performance in Experiment 1 (Section 7.3.1).

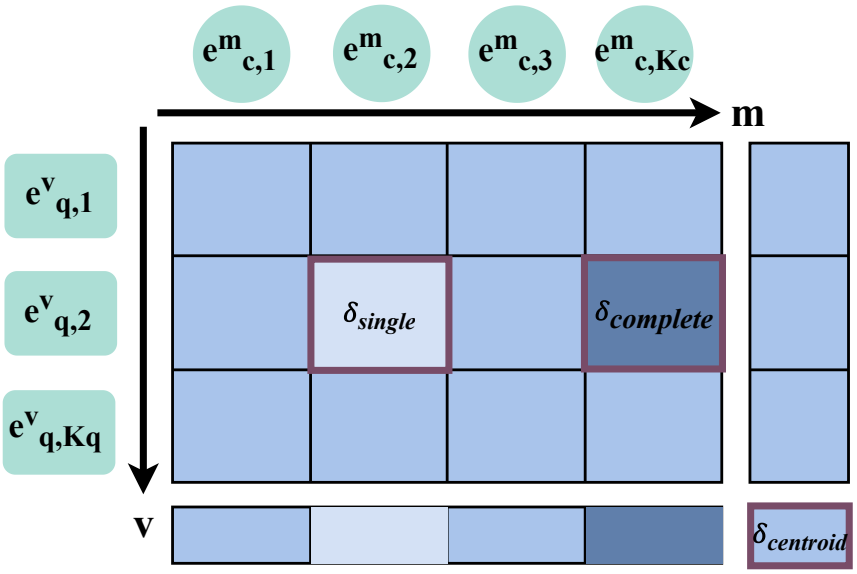


FIGURE 7.3: Experiment 1: Illustration of the proposed segment aggregation methods δ_{single}, $\delta_{complete}$ and $\delta_{centroid}$. The darker the color of the cell, the larger the distance between the two segment embeddings.

Centroid aggregation: The distance $\delta_{centroid}$ between a music track c and a video query q is the Euclidean distance between the centroid of the embedding vectors of all of their respective segments.

$$\delta_{centroid}(c,q) = ||\frac{1}{K_c}\sum_{i=1}^{K_c} e^m_{c,i} - \frac{1}{K_q}\sum_{j=1}^{K_q} e^v_{q,j}||^2. \tag{7.1}$$

Single linkage: The distance δ_{single} is the single linkage clustering between the two sets of embedding vectors.

$$\delta_{single}(c,q) = \min_{i,j} ||e^m_{c,i} - e^v_{q,j}||^2. \tag{7.2}$$

Complete linkage: The distance $\delta_{complete}$ is the complete linkage clustering between the two sets of embedding vectors.

$$\delta_{complete}(c,q) = \max_{i,j} ||e^m_{c,i} - e^v_{q,j}||^2. \tag{7.3}$$

Alignment costs. A more sophisticated method is to consider the temporal succession of segments within each AV clip. By doing so, we leverage the natural temporal ordering of the segments within each AV clip. The ranking distances δ can then be expressed as alignment scores between two temporally ordered sequences of segments. In Experiment 2, we investigate several ways of computing this alignment cost. We illustrate the concept of each alignment cost on Figure 7.4. We discuss their respective performance in Section 7.3.2.

NW-DTW: The distance δ_{NW-DTW} is a Dynamic Time Warping (DTW) cost between the two sequences of embedding vectors. The DTW algorithm used is inspired from the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970). The

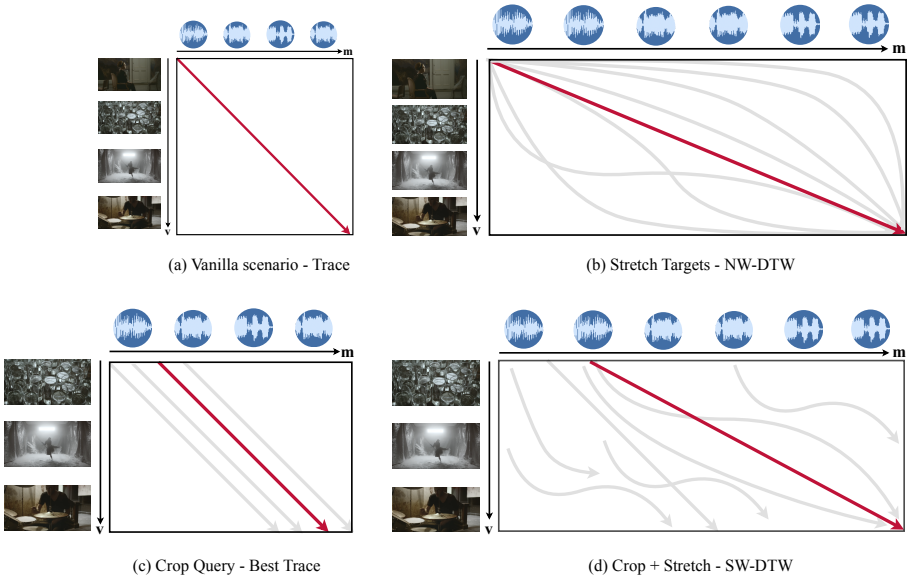


FIGURE 7.4: Schematic representations of the pairwise distance matrix between the segment embeddings: $D_{i,j} = ||e^m_{c,i} - e^v_{q,j}||^2$. Each subfigure represents an evaluation scenario and an example ranking distance suitable to find the optimal path (such that δ is minimal) in D. For each evaluation scenario, we display in red an example of ground truth path. We display in grey examples of paths which can be found by the represented alignment distance.

NW-DTW we used is detailed in Algorithm 1 for the case of similarity (the distance is then the opposite) where $m[i] = e^m_{c,i}$ and $v[j] = e^v_{q,j}$. We used an *indel* cost of 0.05, determined via a grid search.

SW-DTW: The distance δ_{SW-DTW} is a DTW cost between the two optimal *subsequences* of embedding vectors. The DTW algorithm used is inspired from the Smith-Waterman algorithm (Smith and Waterman, 1981). Contrary to NW-DTW, the SW-DTW variant allows to compute the alignment cost based on local sequence alignment rather than on a full path. The SW-DTW we used is detailed in Algorithm 2 for the case of similarity (the distance is then the opposite) where $m[i] = e^m_{c,i}$ and $v[j] = e^v_{q,j}$. We used an *indel* cost of 0.01, determined via a grid search.

Trace: δ_{trace} is the sum of the coefficients on the main diagonal of the pairwise distance matrix:

$$\delta_{trace}(c,q) = \sum_{i=1}^{min(K_c,K_q)} ||e^m_{c,i} - e^v_{q,i}||^2. \tag{7.4}$$

Best (B.) Trace: δ_{btrace} is the minimal trace over all diagonals of the pairwise distance matrix:

$$\delta_{btrace}(c,q) = \min_{k\in[0,K_q-K_c]} \sum_{i=1}^{K_c} ||e^m_{c,i} - e^v_{q,i+k}||^2 \text{ if } K_c < K_q; \tag{7.5}$$

$$\delta_{btrace}(c,q) = \min_{k\in[0,K_c-K_q]} \sum_{i=1}^{K_q} ||e^m_{c,i+k} - e^v_{q,i}||^2 \text{ if } K_c > K_q. \tag{7.6}$$

Algorithm 1: Needleman-Wunsch DTW

Result: NW-DTW(m, v)
$indel \leftarrow 0.05$;
for $i \leftarrow 0$ **to** $len(m)$ **do**
  $X[i, 0] \leftarrow -indel * i$;
end
for $j \leftarrow 0$ **to** $len(v)$ **do**
  $X[0, j] \leftarrow -indel * j$;
end
for $i \leftarrow 1$ **to** $len(m)$ **do**
  for $j \leftarrow 1$ **to** $len(v)$ **do**
    $sim \leftarrow m[i] \cdot v[j]$;
    $X[i, j] \leftarrow$
     $\max(X[i-1, j] - indel, X[i, j-1] - indel, X[i-1, j-1] + sim)$;
  end
end
return $X[len(m), len(v)]$;

7.2 Evaluation

We evaluate the Seg-VM-Net on the Music Video Dataset (MVD), as described in Section 5.1.4. The real use case scenario would consist in finding the most appropriate music track (here denoted as target) t^m from a catalog given a video query q^v; this independently of the respective duration of t^m and q^v. However, the evaluation of this real use case scenario would imply human evaluation which cannot be done at scale. To allow performing the evaluation at scale (i.e. using 1,000 different queries), we create two evaluation scenarios.

In both cases, to obtain a ranked list of recommendations for a given video query, we compute the pairwise distances $\{\delta(c^m, q^v)\}_{c\in\{1,\dots,C\}}$ between the query video embedding e_q^v and all music track embeddings $\{e_c^m\}_{c\in\{1,\dots,C\}}$. This ranking distance δ depends on the experiment. We then rank all music tracks by increasing distance δ. Please note that all distances studied in this chapter are symmetrical, and could therefore be used to retrieve videos from a music query. For simplification purposes, and in line with the music supervision use case, we choose here to focus on the V2M case.

7.2.1 Vanilla evaluation scenario

In the vanilla evaluation scenario, each video part of the 1,000 clips of the MVD dataset is in turn chosen as a query q, the goal being to retrieve the corresponding music part: $q^v = c^v, t^m = c^m$. Although convenient (as it allows a direct evaluation of our systems), this scenario is obviously not realistic, because since q^v and t^m are coming from the same AV clip c, they share the exact same boundaries $\{c_1, \dots c_{K_c}\}$. Their segments are therefore perfectly aligned (See Figure 7.4 (a)). To solve this issue, we propose a second evaluation scenario, denoted by “realistic".

Algorithm 2: Smith-Waterman DTW

Result: SW-DTW(m, v)
$indel \leftarrow 0.01$;
for $i \leftarrow 0$ **to** $len(m)$ **do**
  $X[i, 0] \leftarrow 0$;
end
for $j \leftarrow 0$ **to** $len(v)$ **do**
  $X[0, j] \leftarrow 0$;
end
for $i \leftarrow 1$ **to** $len(m)$ **do**
  for $j \leftarrow 1$ **to** $len(v)$ **do**
    $sim \leftarrow m[i] \cdot v[j]$;
    $X[i, j] \leftarrow$
     $\max(0, X[i-1, j] - indel, X[i, j-1] - indel, X[i-1, j-1] + sim)$;
  end
end
return $\max(X)$;

7.2.2 Realistic evaluation scenario

In the music supervision task, the best soundtrack is not known a priori. It may also have a different number of segments than the video query, thus requiring some editing. Therefore, the vanilla evaluation scenario only accounts for very specific cases. A relevant recommendation system for music supervision should be able to find the best music track from the catalog, even if some transformations are required in order to obtain the optimal fit. In other words, we need our recommendation system to be robust against video queries that are not perfectly aligned to the candidate music tracks.

In Experiment 3, we test the robustness of the different ranking distances δ using more realistic evaluation scenarios. Since we do not have realistic ground truth data, we apply three types of perturbations to the queries q^v and targets t^m in order to imitate real-life use cases:

"Stretch Targets" : we repeat each segment of the target music tracks:
$\{c_1^m, ..., c_{K_c}^m\} \rightarrow \{c_1^m, c_1^m, ..., c_{K_c}^m, c_{K_c}^m\}$ (See Figure 7.4 (b)).

"Crop Query" : we remove the first two segments of each video query:
$\{c_1^v, c_2^v, c_3^v, ..., c_{K_c}^v\} \rightarrow \{c_3^v, ..., c_{K_c}^v\}$ (See Figure 7.4 (c)).

"Crop+Stretch" : we apply simultaneously the "Crop Query" and "Stretch Targets" perturbations (See Figure 7.4 (d)).

We use these simple transformations in line with the music supervision needs. Indeed, usual video queries (e.g., commercials) are often shorter than full-length music tracks. One common edit on the music tracks is therefore to crop them to select the most relevant excerpt. Swapping segments or time stretching would be much less common, according to the professionals we interviewed in Chapter 6.1.

As before, we then compute all pairwise alignment costs with the modified queries or targets. We discuss the performance of our system against each evaluation scenario in Section 7.3.3.

TABLE 7.1: Results of Experiment 1 in terms of Mean *Rank* (lower is better): comparing segmentation methods S with simple segment aggregation techniques δ in the vanilla evaluation scenario. The music segmentation method Foote and the Centroid aggregation method perform best.

	$\delta_{centroid}$	δ_{single}	$\delta_{complete}$
S_{Foote}	**94 ± 17**	120 ± 18	324 ± 37
S_{SF}	100 ± 17	125 ± 18	337 ± 36
S_{OLDA}	**94 ± 16**	128 ± 19	347 ± 36
$S_{TransNet}$	113 ± 1	172 ± 22	283 ± 31
Baseline 1		193 ± 25	
Baseline 2		118 ± 19	
Chance		500	

7.2.3 Performance measures

We use the same performance measures as in Section 5.3.1: the average Recall at k ($R@k$, $k \in \{1, 10, 25\}$) and average *Rank* of the target over the 1,000 video queries. In both evaluation scenarios (vanilla or realistic), there is a single correct music track to be retrieved among the 1,000 of the dataset. None of the test clips (queries nor targets) were seen during training.

7.3 Results

7.3.1 Experiment 1: Basic segment aggregation

In the first experiment, we compare the performances of the original *clip-level* VM-Net (represented by the baseline systems of Hong, Im, and Yang, 2018 and Prétet, Richard, and Peeters, 2021a) to our proposed *segment-level* Seg-VM-Nets. All evaluations are performed using the vanilla scenario.

We consider two baseline systems trained at *clip level*. The two only differ by the choice of the input audio features. The Baseline 1 is the VM-Net trained at clip level with the original handcrafted audio features obtained from Librosa, as in Hong, Im, and Yang, 2018. The Baseline 2 uses the pre-trained AudioSet features, as we proposed in Chapter 5. These AudioSet features are the ones we use in all our segment-level systems.

For the segment-level systems, we compare the use of the various segmentation algorithms S: based on the music structure (S_{Foote}, S_{SF}, S_{OLDA}) or based on the video structure ($S_{TransNet}$). S is used to define the segments which are used both for training the Seg-VM-Net (leading to different embedding projections $e = f_S(x)$), and at inference time to match the segments of two media. For the inference, we compare the use of the three cluster aggregation methods: $\delta_{centroid}$, δ_{single} and $\delta_{complete}$. In this experiment, the matching function δ does not take into account the temporal organization of the segments.

Computation time. The training time of the Seg-VM-Net for this experiment varies between two and six days. All test scripts took less than one minute to process the 1,000 queries of the MVD.

Table 7.1 and Figure 7.5 show the results of Experiment 1.

The Centroid segment aggregation method is beneficial on almost all metrics, however it is not the case for the other ranking distances studied. Among the

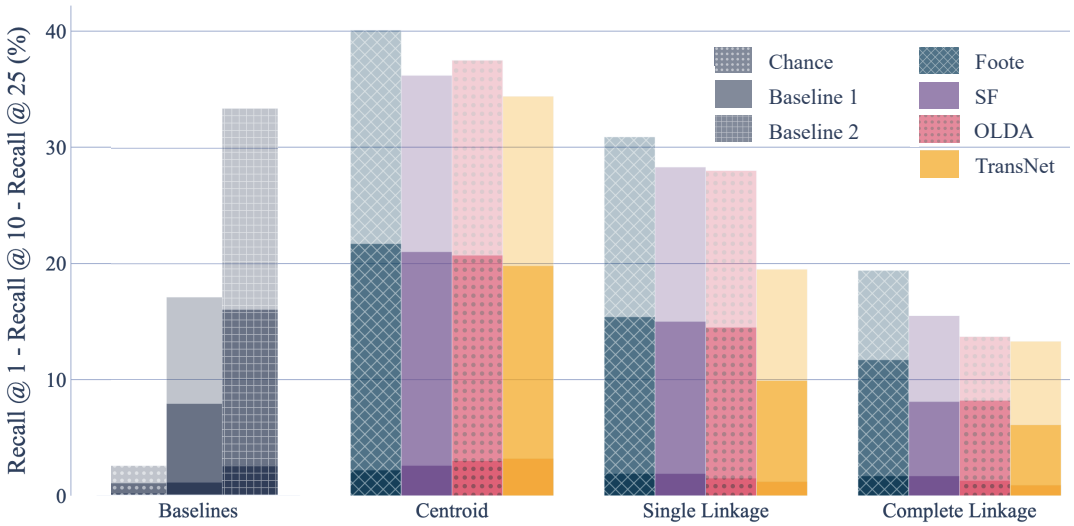


FIGURE 7.5: Results of Experiment 1 in terms of Recall: comparing segmentation methods S with simple segment aggregation techniques δ in the vanilla evaluation scenario. The three Recall metrics (R@1, R@10 and R@25) are stacked with decreasing opacity. Higher is better. The Foote segmentation performs best within each alignment method. Best viewed in color.

various aggregation methods, $\delta_{centroid}$ perform best, while $\delta_{complete}$ performs worst. $\delta_{centroid}$ trained on segments defined by S_{Foote} performs better than the clip-level baselines on all metrics. To understand this, let us be reminded that the clip-level VM-Nets are trained on input vectors x_c^m / x_c^v which represent the whole AV clip. On the opposite end, the segment-level Seg-VM-Nets are trained on input vectors which represent segments $x_{c,k}^m / x_{c,l}^v$ of the AV clips of *variable duration*. In $\delta_{centroid}$, the resulting embeddings are then averaged over the clip to produce the recommendation. As a result, for $\delta_{centroid}$, short and long sequences have the same weight for the recommendation. We expect this to be an advantage in the case where shorter sequences are especially meaningful.

We observe a variation of the performance of the Seg-VM-Net according to the segmentation strategy S of the AV clips. The segmentation S_{Foote}, which is based on homogeneity of consecutive audio frames, seems to perform best for all the cluster aggregation methods. $S_{TransNet}$ results in poor performance in almost every setting. Compared to the three audio-based methods, $S_{TransNet}$ tends to produce many more segments per clip (i.e. the average shot duration is shorter than the average music segment duration). While most music boundaries are reflected by cuts when editing a music video, we do not expect every cut to have a signification in terms of music. This may explain why $S_{TransNet}$ has poor performance.

Experiment 1 therefore supports the intuition that the segment-level embeddings carry extra representation power, but a better ranking distance δ is required in order to take full advantage of it.

7.3.2 Experiment 2: Segment alignment

In the second experiment, we focus on the segment-level systems (Seg-VM-Net). We compare the performance obtained using the five sequence alignment algorithms δ: δ_{NW-DTW}, δ_{SW-DTW}, δ_{trace} and δ_{btrace}. We also still compare the segmentation algorithms S_{Foote}, S_{SF}, S_{OLDA} and $S_{TransNet}$. All evaluations are again performed

TABLE 7.2: Results of Experiment 2 in terms of Mean *Rank* (lower is better): comparing segment alignment methods in the vanilla evaluation scenario. The Trace alignment method performs best both in terms of metrics and inference duration.

		Mean Rank	Time
	Chance	500	00'00"
	Baseline 1	193 ± 25	00'10"
	Baseline 2	118 ± 19	00'10"
δ_{NW-DTW}	S_{Foote}	56 ± 13	11'59"
	S_{SF}	56 ± 13	10'09"
	S_{OLDA}	46 ± 12	13'41"
	$S_{TransNet}$	60 ± 13	48'29"
δ_{SW-DTW}	S_{Foote}	68 ± 14	19'40"
	S_{SF}	70 ± 15	16'20"
	S_{OLDA}	62 ± 14	21'14"
	$S_{TransNet}$	99 ± 17	1h13'56"
δ_{trace}	S_{Foote}	44 ± 12	00'55"
	S_{SF}	44 ± 11	01'00"
	S_{OLDA}	$\mathbf{32 \pm 10}$	**00'53"**
	$S_{TransNet}$	43 ± 11	01'45"
δ_{btrace}	S_{Foote}	55 ± 13	01'45"
	S_{SF}	55 ± 13	01'32"
	S_{OLDA}	43 ± 12	01'35"
	$S_{TransNet}$	73 ± 16	03'51"

using the vanilla scenario. Table 7.2 and Figure 7.6 show the results of Experiment 2.

Segment alignment metrics. It seems clear that the segment-level approach combined with alignment metrics allows for a large improvement compared to the clip-level systems but also compared to the cluster aggregation methods (Experiment 1). According to Figure 7.6, it seems that this time S_{OLDA} allows for the largest improvement. The highest R@25 is achieved using δ_{Trace} and S_{OLDA}: the Seg-VM-Net then reaches a R@25 $= 78.40$ (for more than three out of four queries, the Seg-VM-Net was able to retrieve the matching music track in its top 25). Table 7.2 indicates that for this configuration, the Seg-VM-Net achieves a Mean *Rank* of 32 (the matching video is on average at rank 32, out of 1,000 videos). In comparison, the Baseline 2 clip-level system had a much higher Mean *Rank* (118). The improvement is also very noticeable compared to the $\delta_{centroid}$ Seg-VM-Net (see Experiment 1) which had a $Rank = 96$.

The fact that δ_{trace} provides the best results in the vanilla evaluation scenario is not surprising, since in this scenario, the sequences of segments of the target music and query video perfectly match each other. For the vanilla scenario, there is no gain to use the more elaborated δ_{NW-DTW}, δ_{SW-DTW} and δ_{btrace}. Performing a global alignment (δ_{NW-DTW}) or allowing an offset to the trace (δ_{btrace}) both result in degraded performance. The subsequence alignment method (δ_{SW-DTW}) produces the worst performance.

Computation time. In this experiment, the time for inference can vary a lot according to the choice of δ; we provide each inference time in the table of results. We performed all retrieval operations on an Intel Core i7-6850K CPU with 64G of

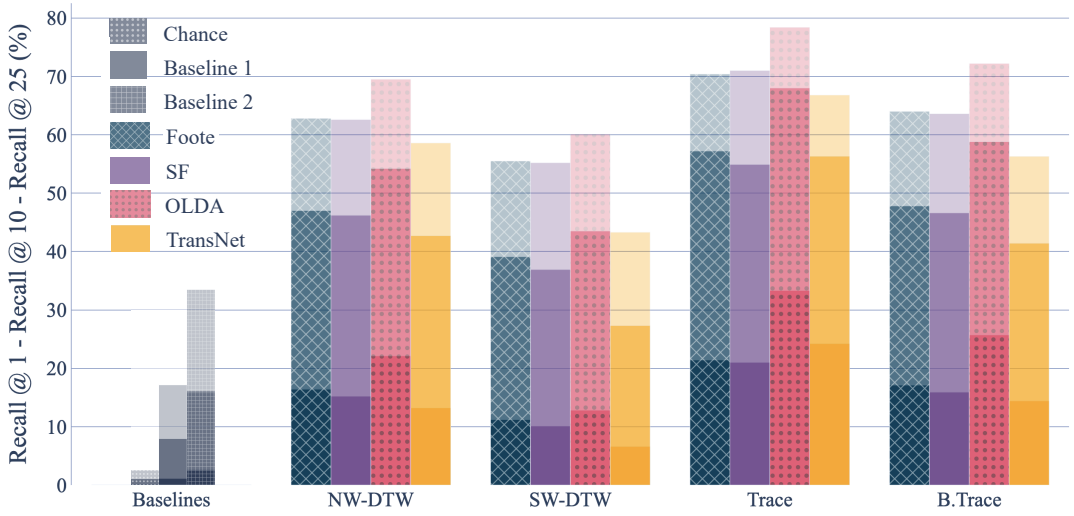


FIGURE 7.6: Results of Experiment 2 in terms of Recall: comparing segment alignment methods (vanilla evaluation scenario). The three Recall metrics (R@1, R@10 and R@25) are stacked with decreasing opacity. Higher is better. The OLDA segmentation performs best within each alignment method. Best viewed in color.

RAM (see Appendix B.1 for more hardware details). Overall, the inference duration of δ_{trace} and δ_{btrace} seem reasonable to us, as it represents the processing of 1,000 queries. In a realistic use case, the user will enter one query at a time, thus expecting a response in about 0.1s for the system trained on S_{OLDA} segments with the δ_{trace} alignment cost. Using the $S_{TransNet}$ segmentation results systematically in longer inference time, because of the higher number of segments to align. Both DTW alignment methods (δ_{NW-DTW} and δ_{SW-DTW}) result in high computation times. This was expected, as their complexity is quadratic. Only allowing an offset (δ_{btrace}) results in a slight increase of computation time compared to δ_{trace} (1'35 instead of 53" for OLDA), with performance similar to the more costly δ_{NW-DTW}.

7.3.3 Experiment 3: Realistic evaluation

While Experiments 1 and 2 were achieved using the vanilla evaluation scenario, we now test the realistic evaluation scenario where query and targets do not (necessarily) have the same number of segments. To mimic the real-case scenario, we artificially modified either the query or the targets (using "Crop Query", "Stretch Targets" and "Crop+Stretch").

We then again compare the various pairwise ranking distances $\delta_{centroid}$, δ_{NW-DTW}, δ_{SW-DTW}, δ_{trace} and δ_{btrace} but using only the best segmentation algorithm S_{OLDA}.

Table 7.3 and Figure 7.7 show the results of Experiment 3. As can be seen, the perturbation of the queries and targets results in a degradation of the performance of all systems, compared to the vanilla evaluation procedure (see Table 7.2). The $\delta_{centroid}$ and δ_{trace} systems are particularly impacted by this degradation. Following our intuition, δ_{btrace} is the most robust against the "Crop Query" perturbation. Additionally, δ_{NW-DTW} and δ_{SW-DTW} are the most robust against the "Stretch Targets" and "Crop+Stretch" degradations. This was expected, as the DTW alignment algorithms are the only ones that can handle regression coefficients other than 1 (See Figure 7.4). The δ_{NW-DTW}, which is constrained to use all segments from the first to the last one, performs best in terms of R@10, R@25 and Rank. The δ_{SW-DTW}, which

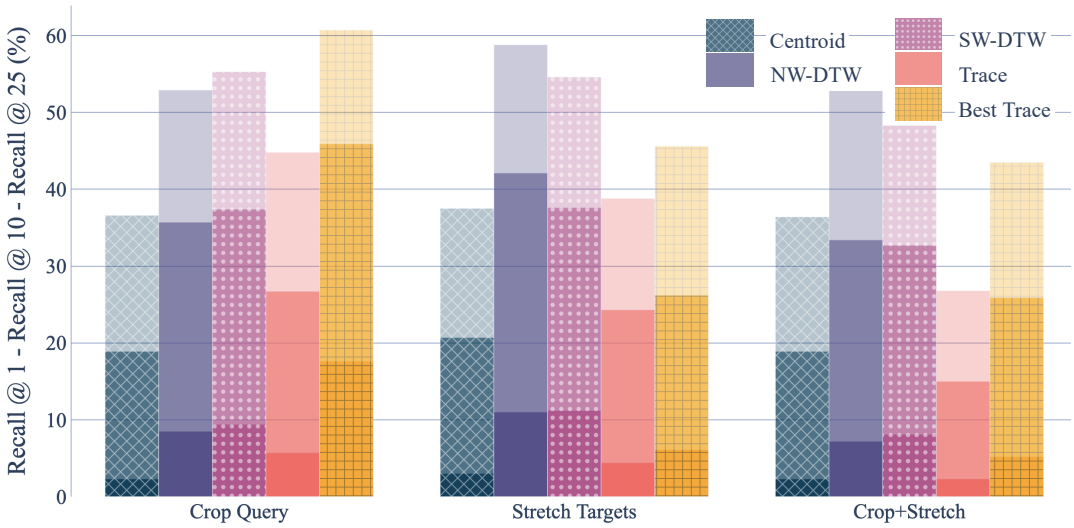


FIGURE 7.7: Results of Experiment 3 in terms of Recall using the realistic evaluation scenario and S_{OLDA}. The three Recall metrics (R@1, R@10 and R@25) are stacked with decreasing opacity. Higher is better. δ_{btrace} performs best in the "Crop Query" scenario, while the δ_{NW-DTW} and δ_{SW-DTW} perform best in the scenarios involving Stretch. Best viewed in color.

can ignore segments at the beginning and end of the media, performs best in terms of R@1.

7.3.4 Qualitative analysis

Validating our segment assumption. The assumption that motivated our Seg-VM-Net is that the content of c^v or c^m is not homogeneous over time and therefore can hardly be represented by the timeless embeddings vectors e^v_c or e^m_c. We therefore proposed to represent c as a succession of homogeneous segments $\{c_1, \dots c_{K_c}\}$ and their respective embeddings. Assuming that the content is not homogeneous over time is equivalent to say that the similarity between $e^v_{c,l}$ and $e^m_{c,l}$ is higher than between $e^v_{c,l}$ and $e^m_{c,k\neq l}$. We illustrate on Figure 7.8 an example of pairwise similarity matrix X between the music $e^m_{c,*}$ and video segments $e^v_{c,*}$ of a same AV clip c. We did not perturb the video, nor the music part of the clip (vanilla scenario). The vertical axis represents the sequence of music segments $\{e^m_{c,1}, ..., e^m_{c,K_c}\}$, while the horizontal axis represents the sequence of video segments $\{e^v_{c,1}, ..., e^v_{c,K_c}\}$. Each coefficient $X_{i,j}$

TABLE 7.3: Results of Experiment 3 in terms of Mean *Rank* using the realistic evaluation and S_{OLDA}. Lower is better. Scenario "Crop Query": The δ_{btrace} system is the most robust against the cropping of the query. Scenarios "Stretch Targets" and "Crop+Stretch": δ_{NW-DTW} is the most robust against the perturbation of the target music tracks.

	Crop Query	Stretch Targets	Crop+Stretch
$\delta_{centroid}$	96 ± 16	94 ± 16	96 ± 16
δ_{NW-DTW}	75 ± 16	$\mathbf{55 \pm 12}$	$\mathbf{65 \pm 13}$
δ_{SW-DTW}	72 ± 15	68 ± 15	79 ± 16
δ_{trace}	92 ± 17	90 ± 16	146 ± 22
δ_{btrace}	$\mathbf{60 \pm 14}$	83 ± 15	89 ± 16

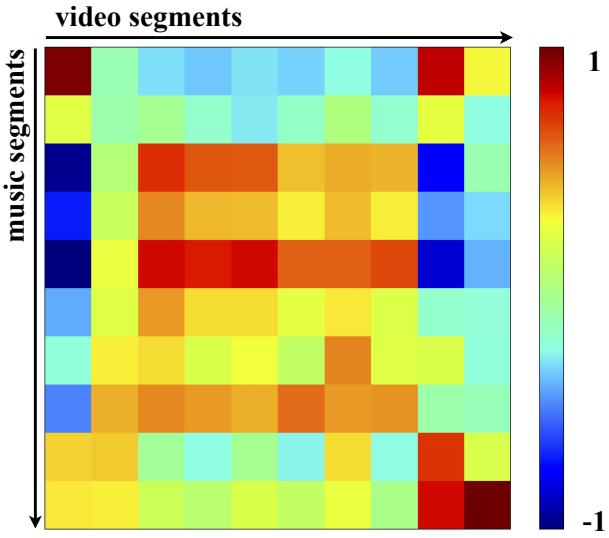


FIGURE 7.8: Pairwise similarity matrix X of segment embeddings for the track RnB_11 of the MVD (S_{OLDA} segmentation). Warm colors at location (i, j) indicate high similarity between $e^m_{c,i}$ and $e^v_{c,j}$, while cold colors indicate low similarity.

of the pairwise similarity matrix is computed as the scalar product between $e^m_{c,i}$ and $e^v_{c,j}$. We display in red the coefficients of high similarity (close to 1) and in blue the ones of low similarity (close to -1). An ideal system would produce very high values on the diagonal and very low values elsewhere, since only the music and video segments extracted at the same timestamp are associated according to our training criteria. However, given that the example AV clip was taken from the test set, we only observe an approximation of this pattern. Certain segments in the middle of the video seem to be interchangeable, but the very first and last music and video segments are strongly associated. We assume that the beginning and the end of this AV clip are very distinctive and play a crucial role in the recommendation. In this vanilla scenario, we expect the diagonal of X to contain the highest coefficients in case of a match, and therefore we expect δ_{trace} to be the most appropriate ranking distance (schematized on Figure 7.4 (a)).

Illustration of the results. To better understand the system's performance, we provide in Figure 7.9 some examples of recommendations obtained by our best[4] system on the MVD (vanilla evaluation scenario). We give 11 examples of video queries q^v and the suggested music tracks. We only display the first 3 recommendations provided by the system (from left to right). As proposed by Hong, Im, and Yang, 2018, we select one key frame (picture) of the AV clip to represent both the music track and the video. Below each picture, we provide the name of the AV clip as provided in the MVD. With blue boxes, we highlight the hits, i.e. when the Seg-VM-Net retrieved the correct track in the displayed top 3. For the first query, the correct music track was retrieved in the first position, hence its $R@3$ is one. To the right of the figure, we display the histogram of the distances between the query and all cross-modal samples. We highlight in red the histogram bin corresponding to the ground truth

[4]We used S_{OLDA} and δ_{trace}.

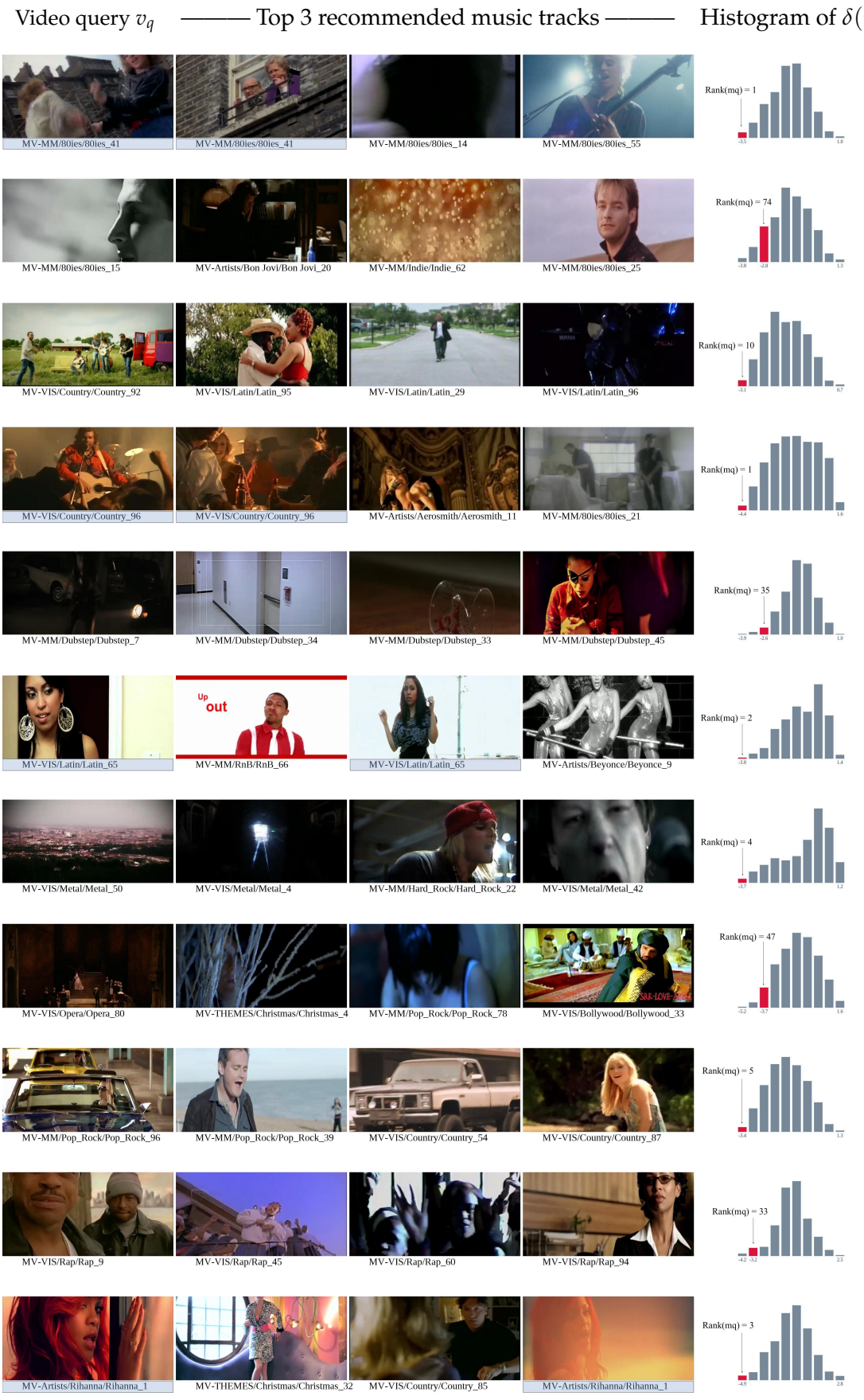


FIGURE 7.9: Qualitative results of the δ_{trace} system trained with S_{OLDA} on three video queries from the MVD.

sample q^m, along with its rank in the list of recommendations. Note that the exact matching sample of the query was not retrieved in most examples, hence the $R@3$ for these queries is zero. As we see, when the Recall is zero, the recommended music tracks are from a similar music genre to the video query, which still makes sense in terms of applications. This shows the limitations of our current Recall metric.

7.4 Summary

In this study, we present a simple, yet efficient Video-to-Music recommendation system: the Seg-VM-Net. We do so by improving a self-supervised system, the VM-Net. Inspired by the music supervision use case, we train our model at segment level, using segments defined by the music and video structure. We then rank the tracks according to a segment alignment cost. With these adaptations, we largely outperform the clip-level baselines and the basic segment aggregation systems. This demonstrates that while both the aggregation and alignment systems employ the exact same training weights, the neural networks can have very different retrieval performance, depending on the way embedding vectors are used at inference time. Additionally, the resulting method makes use of pre-trained features and a lightweight network architecture, thus requiring a highly reasonable training time. Depending on the complexity of the application scenario, we demonstrate many possible ranking distances, which provide various tradeoffs between retrieval time and robustness of the performance. For example, the Best Trace alignment cost is fast to compute and offers robustness to basic transformations of the query (cropping). It therefore is the best suited for large catalogs. The two DTW alignment costs make the system robust against other transformations of the queries and targets (stretching), while requiring additional computation time. They can be used in scenarios involving medium-size catalogs and little constraints on the allowed media transformations. These different options make the Seg-VM-Net flexible and suitable for applications in music recommendation, music supervision or automatic video editing.

Chapter 8

Conclusion

8.1 Summary of contributions

In this thesis, we studied the topic of music supervision, i.e. recommending music to be used as a soundtrack for videos. We briefly review here the main contributions of our work.

8.1.1 Music similarity

In Chapter 4, we designed a system for music recommendation by similarity with respect to a music query. The MuSimNet requires tags and an oracle similarity metric for training, but only the audio tracks for inference, allowing to bypass the use of tags (manual or estimated) for new tracks. This system represents similarity as a Euclidean distance in an embedding space, for fast and scalable retrieval. The main challenge in the conception of the MuSimNet was to have it learn the embedding space directly from reference ranked lists. We thus trained it using an innovative triplet mining algorithm, which we compared and validated on a large-scale experiment against an auto-tagging based approach. The results obtained highlight the efficiency of our system, especially when associated with the recently proposed Auto-pooling layer. In music supervision, where a large part of the time is dedicated to manually searching music tracks and editing metadata, an audio-based music similarity recommendation system is an important asset.

8.1.2 Video/Music recommendation

The next step in our study was to prototype automatic Video/Music recommendation systems to assist music industry professionals. While existing systems are already able to recommend music based on the video content, the temporal synchronization of both modalities is rarely taken into account. Ideally, our recommendation system should not only be effective and fast, but also take into account the domain-specific constraints of synchronization between the audio and video events.

In Chapter 5, we studied a basic system for music recommendation from video queries, the VM-Net. We showed that using audio representation learning such as pre-trained audio embeddings (MuSimNet, OpenL3, MusicCNN or AudioSet) largely improved recommendations. We validated the use of metric learning against a classification-based approach. We also demonstrated that learning the feature's temporal aggregation (with Attention or Auto-pooling) was a simple way to improve the VM-Net's retrieval performance.

In Chapter 6, we analyzed which aspects of the structure were decisive when editing music videos. To do so, we first interviewed music video professionals to obtain their insight, and then assessed these on a corpus study. We showed that

cuts, downbeats and functional segment boundaries were important for the editing of OMVs. More interestingly, the level of synchronization would vary according to the music and video genres. The paper's presentation video received the Best Video Award at ISMIR 2021.

These two studies resulted in the introduction of a new V/M recommendation system in Chapter 7. The Seg-VM-Net relies on a semantic segmentation of the query and target media, a training at segment level and a ranking based on the alignment of the segment sequences. It offers better performance and flexibility compared to the VM-Net, while keeping the training phase straightforward. By using different alignment costs, it is possible to make the system robust against various application scenarios. The Seg-VM-Net and the perspectives it offers in terms of cross-modal synchronization were awarded by a Grand Prix at the i-Lab 2021 innovation challenge (corresponding to a funding of 600k€ for the start-up Bridge.audio).

8.2 Discussion

8.2.1 Music similarity

Limitations. Several axes are worth exploring when it comes to improve the MuSimNet in terms of training methods, architecture, or input representations. During our preliminary work on triplet mining for music similarity, we experimented with incremental margin parameters, as in Zhang et al., 2019. However, this did not seem to have any impact on the performance of the system.

The evaluation method remains also an open question. Ideally, we aim at validating our recommendation system through formal perceptual evaluation, and online user testing.

Short-term perspectives. In future works, we aim at increasing the network training efficiency by mining useful triplets on the fly at each iteration with in-batch triplet mining (Hermans, Beyer, and Leibe, 2017; Mishchuk et al., 2017; Doras and Peeters, 2019). The performance might also be improved by appending a graph neural network component, as has been proposed by Korzeniowski, Oramas, and Gouyon, 2021 for artist similarity.

Other perspectives include the use of more sophisticated audio input representations (such as Harmonic Constant-Q Transform -Bittner et al., 2017-, dominant melody, pitch class profiles), or a combination thereof (Doras et al., 2020).

8.2.2 Video/Music recommendation

Limitations. As seen in Chapter 5, there is room for potential improvement in the VM-Net performance by fine-tuning the audio feature extractors (MuSimNet, MusiCNN, AudioSet and OpenL3) along with the VM-Net.

When conducting experiments for Chapter 7, we tried more exhaustive triplet mining strategies, such as using segments adjacent to the anchor as negatives. However, none seemed to have a positive impact on the metrics.

The main limitation of the Seg-VM-Net is its coarse temporal resolution, as the music structural segments last on average for 18.6 seconds. While we could probably further improve clip-level retrieval performance by learning the frame-to-segment feature aggregation, as we did for the clips in Chapter 5, this would still not allow to perform a fine-grained alignment of the two modalities.

Short-term perspectives. Future work will thus have to include frame-level temporal alignment scores to our ranking distance, as Gillet, Essid, and Richard, 2007 did (one video frame is usually 40 milliseconds). A fine synchronization precision was also achieved by Gowing et al., 2011 for multimodal alignment of dance movement. The corresponding challenge would require increasing the temporal precision while keeping the training and inference costs reasonable, especially for the DTW systems. In this regard, we may have to preselect a reduced set of tracks to rank using a faster method such as the Centroid aggregation.

So far, the evaluation protocol of the V/M recommendation systems requires to retrieve the exact same ground truth video or music from the pair, which is a severe metric. In a real case scenario, where there is no ground truth to be retrieved, we assume that some degree of serendipity is allowed. A subjective human evaluation of the obtained recommendations (by users or a panel of experts) would therefore help evaluate the quality of the recommendations. Finally, we want our system to be able to generalize to any type of input videos, e.g. movies or commercials. This will require to first put up a dataset which includes such videos.

Currently, to use the Seg-VM-Net in a real-life scenario (where music boundaries are not available a priori), it is still necessary to first segment the video queries. This can be done either manually (involving scalability issues) or automatically, as we have shown with the TransNet's segmentation into shots. However, we would like to integrate a more diverse and meaningful segmentation of the video. It is already possible to automatically detect effects such as fades and wipes (Zhang, Kankanhalli, and Smoliar, 1993; Zabih, Miller, and Mai, 1995), and chapters (Yamauchi et al., 2006). To refine the video processing part of our system, it will be essential to take these into account.

8.3 Future directions

Finally, we discuss the long-term perspectives of our study. Music supervision is a wide topic for both MIR and computer vision research, and we only provided a brief introduction. We are convinced that several mid-and long-term research directions will arise from this rich domain.

8.3.1 Beyond Video and Music data

In this dissertation, we only rely on the music and video signals to perform the recommendation. This has the advantage of adapting to any music database, including very large and non-annotated catalogs.

However, when facing a real-life music supervision project, many constraints other than content and structure matching come into consideration: the client's vision, budget, the release date and the general popularity of the track, availability of the rights in the diffusion country, adequacy with the brand's identity (for advertisement), etc (Inskip, Macfarlane, and Rafferty, 2008).

Including other matching criteria such as release date, artist popularity (Bauer and Schedl, 2019) or harmonic complexity (Weiss and Muller, 2015) could strengthen our systems.

Finally, lyrics information can be exploited to match music to music and music to video. Lyrics convey mood, but also various semantic data which can be relevant for the recommendation. We can imagine using lyrics data to supervise our cross-modal embedding training. The main difficulty will be to collect said lyrics data,

but as lyrics transcription systems become more and more efficient, we expect to be soon able to extract those from audio (Vaglio et al., 2021).

8.3.2 Video/Music synchronization

In the long term, we want to be able to directly propose an edit of the music tracks to the video query. Given each recommended music track, and the segmented video, the system will suggest one or several edits of the track on the video. To do so, the Seg-VM-Net has to be robust against more diverse transformations of the queries and targets, and to integrate an automatic synchronization module.

In music supervision, synchronization is articulated around so-called *sync points*, which are salient events in the image, carrying specific emotions or energy (see Appendix A.2). Sync points are expected to be emphasized by the music. The detection of sync points in video queries is therefore a crucial step. Very often, the type of mood, energy and movement around the sync point will even condition the choice of music tracks. Yet, to our knowledge, there is no literature on automatic sync point detection. More concerning is the absence of labeled datasets for the sync point detection task. Future works will have to tackle this lack of data first, before we are able to integrate a music and video salient event detection module to our system. As a first step, it is possible to have the user give an estimation of the number of sync points to be found in the video. This can be a simple function of the duration: on average, sync points occur every 15 to 30 seconds. Close topics that will be worth studying for sync point detection are video summarization (Sargent et al., 2016; Baraldi, Grana, and Cucchiara, 2017) and captioning (Chen et al., 2018). Finally, if we are able to reliably detect sync points in the silent video, we can consider using those to automatically segment the queries.

8.3.3 Application to other tasks

Once trained, V/M embedding spaces such as the one of the Seg-VM-Net can be reused in many contexts. From a MIR point of view, video data can facilitate singer and instrument recognition. From a computer vision point of view, music data can enhance singer and instrument diarization, especially in off-screen scenarios. In both cases, multimodal embedding vectors can be reused as input features for classification tasks.

In the optimistic case where we successfully integrate a sync point detection module and an automatic edition module to our system, we can tackle other interesting application scenarios such as automatic soundtrack generation for video games. In this case, the real-time processing is another challenging factor to consider.

Appendix A

Case study: Creaminal, a music supervision agency

A.1 Introduction to Creaminal

Creaminal is a French music supervision agency. Their office is located in Paris and they employ about 25 people. The core of the team is made of musicological experts called music supervisors. They are dedicated to recommending, synchronizing and producing music for videos such as advertisements, movies, series or others. To this aim, they have built over the years a large catalog of annotated music tracks. Creaminal's catalog is comprised of more than 35,000 tracks (as of September 2021) and is growing every day. Most tracks are annotated as Electro or Pop, and a substantial number of movie soundtracks are present as well. A backoffice software allows the employees to annotate the tracks and to retrieve them via several types of searches. A player allows to listen to the tracks.

A.2 Insight on the music supervision work: Interview

As a preliminary work to this dissertation, we interviewed Stéphanie Sfeir, senior music supervisor at Creaminal, to know more about her work process. We provide below a translation of the transcript of the discussion (that was originally conducted in French).

How do you proceed when searching for a music track to be synchronized on a commercial?

> I analyze the content of the video: the story, the narration, or lack thereof.
>
> I check the product that is presented: cars typically require mainstream music (electro or vintage), perfumes will be very dependent on the target (men or women), etc. For this part, it's ok to not take the semantics literally: counter-intuitive choices can be made.
>
> I analyze the target of the commercial: age, genre.
>
> I analyze the structure of the video and the salience points (sync points). Finding the sync points (when an important event occurs in the video) is both easy for a human and crucial for synchronization. The music will typically have to emphasize them via a build-up, an explosion or another structural audio feature. In general, commercials have 3 parts: an intro, one or two sync points, and a pack shot (outro). Each time the product is put forward in the video, the music should react with an audio event. However, if the track is already well-known, its structure must be respected and edits should be light and seamless.

I check my brief: sometimes the client provides examples, sometimes "no-gos". Sometimes I am given a free rein. The client's brief provides filters that I have to accommodate with my own vision of the movie.

The rhythm, style, mood and movement of the video are equally important. The style will often be constraint by the brief or the brand's DNA (general communication strategy).

What do you prepare for the client after you got the brief and the movie?

It depends if we are paid for the search or not. If we are paid (most of the time), we directly propose some mockups (edits of our recommended tracks on the video). Otherwise, we only prepare playlists.

A.3 Description of the Creaminal dataset

In Chapter 4, we use data extracted from Creaminal's internal catalog to train the MuSimNet (music similarity model). We introduce the Creaminal dataset introduced 3.1.2. Although we cannot release the audio, nor the annotations of this dataset, we provide a more detailed description (in terms of genre, number of tags and duration) in this part.

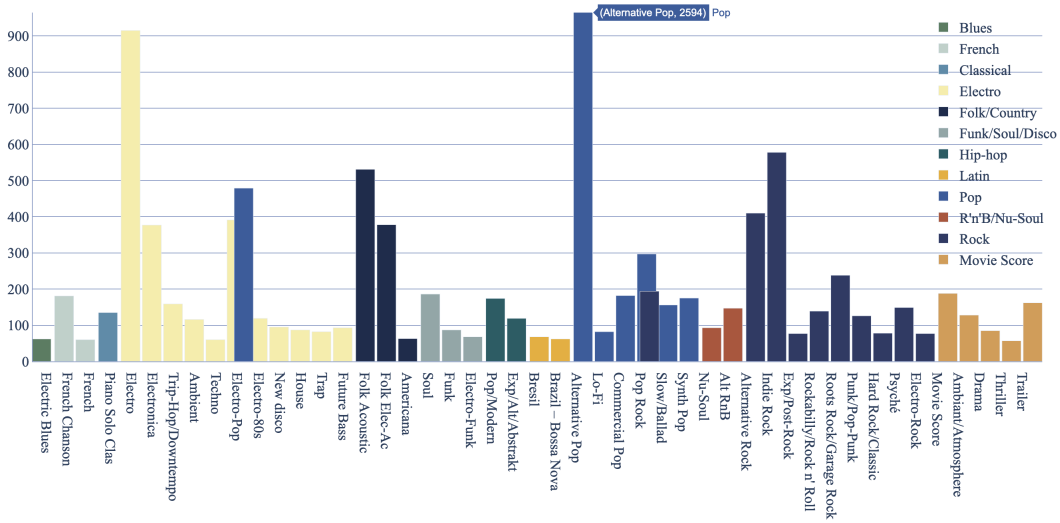


FIGURE A.1: Histogram of music genres in the Creaminal dataset. Only the 50 most represented genres are displayed. Genres are sorted into 12 main families. The genre 'Alternative Pop' being largely over-represented, its bar was cropped for readability. Two bins (Electro-Pop) and (Pop Rock) were annotated from different main families, hence their appearing twice.

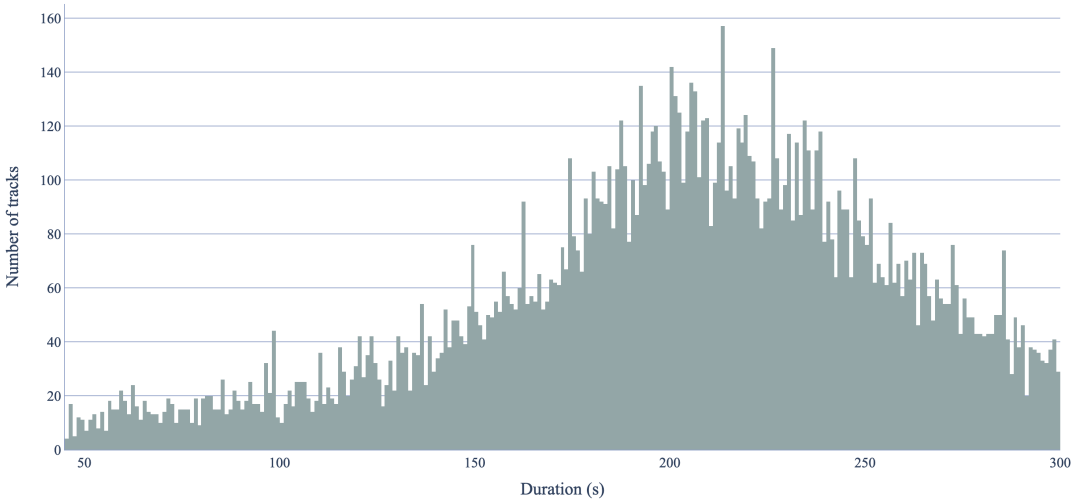


FIGURE A.2: Histogram of track durations in the Creaminal dataset (in seconds). Most tracks last between 3 and 4 minutes.The median duration is 206 seconds which corresponds to 3 minutes 26 seconds.

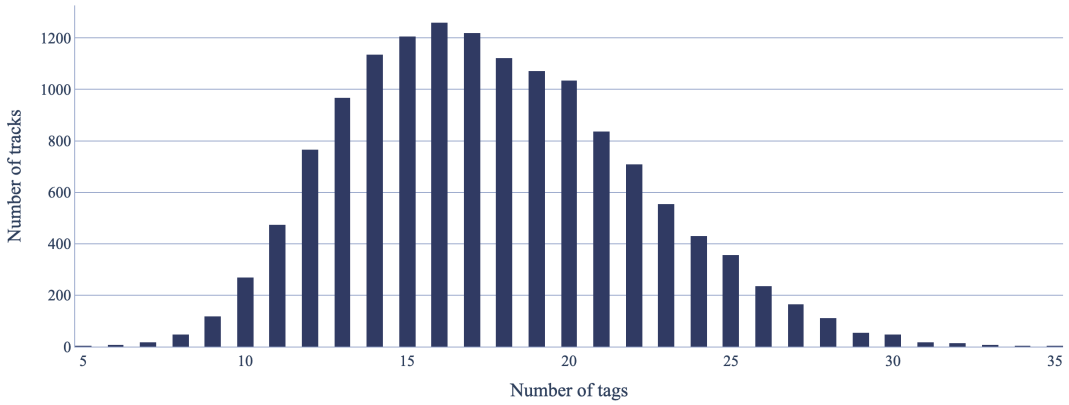


FIGURE A.3: Histogram of the number of tags per track in the Creaminal dataset. Only tracks that have between 5 and 35 tags are considered. Most tracks have between 10 and 25 tags. The median number of tags is 17.

Appendix B

Computational resources

B.1 Hardware experimental setup

All computations presented in this thesis were executed on a private server hosted at Creaminal's main office, Paris. This option was preferred against cloud-based solutions in order to safely store proprietary data from the Creaminal dataset. The computer was installed in a dedicated, air conditioned room. A VPN was configured to access the machine remotely. This equipment has the following configuration:

- 4 GPUs (GeForce GTX 1080 Ti);
- 64Gb of RAM;
- 2Tb of SSD RAID 5 storage;
- 4Tb of extra hard drive storage;
- 12-core Intel Core i7-6850K CPU running at 3.60GHz.

All experiments were conducted using CUDA v9.2.148 and Tensorflow v1.11.0 (as a backend for Keras v2.2.4). Python 3.6.9 was used to run most scripts, except for certain libraries which required earlier or more recent versions.

B.2 Carbon footprint

As the IPCC (Intergovernmental Panel on Climate Change) releases each year more and more alarming assessment reports, it is our responsibility as scientists to at least be aware of our impact on climate. In the machine learning community, neural network training is often pointed to as an energy intensive, and sometimes inefficient way of developing algorithms (Hsueh, 2020). Since this statement was made, several Python libraries were proposed to estimate the carbon footprint of machine learning experiments.

We believe that an important step towards reducing carbon emissions is to systematically report emissions in scientific publications. We therefore used the CodeCarbon emissions tracker to estimate the carbon footprint of our research (Lottick et al., 2019; Lacoste et al., 2019). As reported in Section B.1, all experiments were conducted on a private server located in Paris, France, which has a carbon efficiency of 0.42 kgCO_2eq/kWh. For the results presented in this manuscript, we estimated that a total of 2,543 hours of computation was performed (respectively 1,272h for the MuSimNet of Chapter 4, 983h for the VM-Net of Chapter 5 and 288h for the Seg-VM-Net of Chapter 7). Total emissions are therefore estimated to be 449.5 kgCO_2eq, which is equivalent to driving just above 10,000 kilometers in an average car, according to the U.S. Environmental Protection Agency.

It should be noted that this estimation does not include the carbon emissions of the air conditioning required to cool the server room. Air conditioning can be a significant source of carbon emissions, sometimes exceeding 40% of the total power consumption in typical data centers (Zhang et al., 2017). We did not have a way to measure the energy consumption of the air conditioning system, but we do not expect this to change the order of magnitude of the total emissions. We did not include either the (much higher) cost of developing each model and training all the models which performance was too low to be included in our results. We estimate this cost at 5 times the ones of training the final models. Finally, we estimate that the cost of evaluating the models is negligible compared to training them and therefore should not change the order of magnitude of the total carbon emissions.

Appendix C

Music Similarity: Supplementary material

C.1 Details on the evaluation metrics

We describe here the formula used in Chapter 4 to compute the MAP and nDCG metrics. Let $R_k^{\mathcal{S}}(t) = [r_1(t), \ldots, r_k(t)]$ be the reference tracklist $R^{\mathcal{S}}(t)$, truncated at rank k. Without loss of generality, we consider here that the *relevant* tracks to recommend for a given test query t are the five first tracks of the ranking: $R_5^{\mathcal{S}}(t)$. Thus, for each query track t, we ask our system to retrieve the 5 relevant ground truth recommendations $R_5^{\mathcal{S}}(t)$ among its k estimated recommendation $R_k^{\hat{\mathcal{S}}}(t) = [\hat{r}_1(t), \ldots, \hat{r}_k(t)]$.

In both MAP and nDCG, the position of the relevant tracks in $R_k^{\hat{\mathcal{S}}}(t)$ matters.

C.1.1 MAP formula

In the MAP metric, all relevant tracks $[r_1(t), \ldots, r_k(t)]$ have the same importance. The MAP only evaluates if the relevant tracks appear in high position in $R_k^{\hat{\mathcal{S}}}(t)$.

$$MAP@k(t) = \sum_{i=1}^{k} P(i) * C(i), \tag{C.1}$$

where:

- $P(i) = \frac{\mathbf{card}(R_5^{\mathcal{S}}(t) \cap R_i^{\hat{\mathcal{S}}}(t))}{i}$ is the precision at i;
- $C(i) = \begin{cases} \frac{1}{5}, \text{ if } \hat{r}_i(t) \in R_5^{\mathcal{S}}(t) \\ 0, \text{ otherwise} \end{cases}$ is the change in recall at i.

C.1.2 nDCG formula

In the nDGC metric, the relevant tracks $[r_1(t), \ldots, r_k(t)]$ have different levels of relevance. We use a linear relevance scale, which means that the relevance score of a target track $r_n(t)$ for the query track t decreases linearly as n increases.

$$nDCG@k(t) = \frac{\sum_{i=1}^{k} g(\hat{r}_i(t))/d(i)}{max\_nDCG}, \tag{C.2}$$

with:

- A linear gain function g such that $g(r_1(t)) = 5, g(r_2(t)) = 4, \ldots, g(r_5(t)) = 1, g(r_{[n>5]})(t) = 0$;
- A logarithmic discount function d such that $d(i) = log_2(i+1)$;

- The normalization factor max_nDCG is the best score which is possible to make in the case where $[r_1(t), \ldots, r_5(t)] = [\hat{r}_1(t), \ldots, \hat{r}_5(t)]$:

$$max\_nDCG = \sum_{i=1}^{k} \frac{g(r_i(t))}{log_2(i+1)}. \tag{C.3}$$

C.2 Preliminary results: MuSimNet architecture search

In this part, we present the experiment that led to the choice of a VGG-like architecture for the MuSimNet. We experiment with various popular CNN architectures and compare their performance on the music similarity task. All architectures were trained using the Triplet Loss (TL) paradigm and the Distance-based negative mining strategy.

C.2.1 Presentation of the different architectures

Based on the recent literature on music-related tasks, we selected four architectures of CNNs, which we present now. We adapt those to the shape of our inputs (96×512) and outputs (d units, the dimension of the embedding space). d is set to 128 after a short grid search experiment.

VGG-like: We reproduce the *FCN-5* architecture proposed by Choi, Fazekas, and Sandler, 2016a. This computer vision inspired architecture has been successfully used on auto-tagging tasks. We adapt it here by removing the batch normalization and applying lower dropout rates. We give the resulting architecture in Table C.1. This VGG-like architecture has 25,334,912 trainable parameters.

TABLE C.1: Details of the VGG-like architecture used. ELU means Exponential Linear Units (Clevert, Unterthiner, and Hochreiter, 2016).

Layer Parameters	Activation, Dropout	Output Shape
Conv2D (3,3)×128	ELU	F=96,T=512,C=128
MaxPooling (2,4)		F=48,T=128,C=128
Conv2D (3,3)×256	ReLu, d=0.1	F=48,T=128,C=256
MaxPooling (2,4)		F=24,T=32,C=256
Conv2D (3,3)×512	ReLu, d=0.2	F=24,T=32,C=512
MaxPooling (2,4)		F=12,T=8,C=512
Conv2D (3,3)×1024	ReLu, d=0.25	F=12,T=8,C=1024
MaxPooling (3,3)		F=4,T=2,C=1024
Conv2D (3,3)×2048	ReLu, d=0.3	F=4,T=2,C=2048
MaxPooling (4,2)		F=1,T=1,C=2048
Fully-Connected *(d)*	Linear	F=1,T=1,C=*d*
L2 Norm		F=1,T=1,C=*d*

MusiCNN: This musically-motivated architecture was introduced by Pons et al., 2017 for music auto-tagging. It features long filters, in order to best capture rhythmic (horizontal) patterns and harmonic (vertical) patterns present in the spectrograms. The point of such an architecture, when dealing with music signals, is to train fewer filters, but highly specialized, and more interpretable.

TABLE C.2: Details of the musically-motivated architecture used. ELU stands for Exponential Linear Units (Clevert, Unterthiner, and Hochreiter, 2016). The outputs of the initial block are combined by concatenating all feature maps along the channel axis.

Layer Parameters	Activation, Dropout	Output Shape
Conv2DBlock (64,1)×15 + (64,3)×12 + (64,5)×5 + (32,1)×15 + (32,3)×12 + (32,5)×5 + (1,10)×15 + (3,10)×12 + (5,10)×5 + (1,20)×15 + (3,20)×12 + (5,20)×5	ELU	F=96,T=512,C=128
MaxPooling (2,4)		F=48,T=128,C=128
Conv2D (3,3)×64	ReLu, d=0.1	F=48,T=128,C=64
MaxPooling (2,4)		F=24,T=32,C=64
Conv2D (3,3)×128	ReLu, d=0.2	F=24,T=32,C=128
MaxPooling (2,4)		F=12,T=8,C=128
Conv2D (3,3)×256	ReLu, d=0.25	F=12,T=8,C=256
MaxPooling (3,3)		F=4,T=2,C=256
Conv2D (3,3)×512	ReLu, d=0.3	F=4,T=2,C=512
MaxPooling (4,2)		F=1,T=1,C=512
Fully-Connected *(d)*	Linear	F=1,T=1,C=*d*
L2 Norm		F=1,T=1,C=*d*

To adapt this idea, we replaced the first layer of the VGG-like architecture with a convolution block composed of vertical and horizontal filters of various sizes and shapes. The feature maps produced by each filter are then concatenated along the channel axis. We kept the other layers as it, only reducing the number of filters. We give the details of this architecture in Table C.2. It represents a total of 1,683,980 trainable parameters.

ResNet: We instantiate the VGG-like architecture and add skip-connections around each convolutional block to form residual blocks. This type of skip-connection (also called shortcuts) was first introduced with the ResNet for image recognition (He et al., 2016). Residual networks have shown to be efficient in terms of number of parameters and training time. They were notably used in the Triplet MatchNet (Lu et al., 2017) for learning music similarity on the MagnaTagATune dataset. We used a similar architecture, shown in Table C.3. In this case, the feature maps are not downsampled using pooling layers. Instead, the steps made by the filters after being applied on a coefficient are larger than one, resulting in a **strided** convolution. We indicate the vertical and horizontal stride of each layer in the third column. The final architecture has 6,372,928 trainable parameters.

Inception: Inception networks are popular in computer vision since they were introduced in GoogLeNet Szegedy et al., 2015 for image recognition. Inception modules concatenate the output of filters of different sizes, including 1x1 convolutions which aim at reducing the number of feature maps. They allow to design very deep networks while keeping a reasonable computational cost.

We used the Inception V3 model implemented in the Keras framework[1]. We simply removed the activation in the last fully-connected layer to output embeddings

[1] https://github.com/keras-team/keras-applications/blob/master/keras_applications/inception_v3.py

TABLE C.3: Details of the ResNet architecture used. The output of each block and of the corresponding shortcut layer are combined via an element-wise addition operation. All activations are ReLu and no dropout is used.

Layer Parameters	Shortcut Parameters	Stride	Output Shape
Conv2D (3,3)×64		(2,2)	F=47,T=255,C=64
Conv2D (3,3)×64 Conv2D (3,3)×64	Conv2D (3,3)×64	(2,3)	F=23,T=85,C=64 F=23,T=85,C=64
Conv2D (3,3)×128 Conv2D (3,3)×128	Conv2D (3,3)×128	(2,4)	F=11,T=21,C=128 F=11,T=21,C=128
Conv2D (3,3)×256 Conv2D (3,3)×256	Conv2D (3,3)×256	(2,4)	F=5,T=5,C=256 F=5,T=5,C=256
Conv2D (3,3)×512 Conv2D (3,3)×512	Conv2D (3,3)×512	(2,3)	F=2,T=1,C=512 F=2,T=1,C=512
AveragePooling (2,1)			F=1,T=1,C=512
Fully-Connected *(d)*		Linear	F=1,T=1,C=*d*

rather than classification scores. We obtained a network with 22,012,832 trainable parameters.

C.2.2 Evaluation results

We trained each system on the Creaminal dataset as described in Section 4.3.1. The margin parameter α of the triplet loss is set to 0.5 and the learning rate to 10^{-6}, for a batch size of 42. We evaluate each architecture with the method described in Section 4.3.2 for the TL system.

The results displayed in Table C.4 show that the VGG-like architecture seems to have the best performance on each metric. This is why we selected this architecture for our experiments in Chapter 4. However, it is worth mentioning that the MusiCNN architecture achieves similar results, despite having 15 times fewer parameters. Adding residual connections (ResNet) or Inception blocks (Inception V3) does not seem to help in our case.

TABLE C.4: Comparison of the results of TL system with various architectures. The last columns indicates the number of trainable parameters. All metrics are percentages, with confidence intervals at 95%. Higher is better.

Model	MAP@20	Recall@20	Reciprocal rank	nDCG@20	Size
VGG-like	**5.98 +/- 0.40**	**15.79 +/- 0.73**	**19.89 +/- 1.19**	**14.41 +/- 0.78**	25 M
MusiCNN	5.82 +/- 0.39	15.62 +/- 0.71	19.45 +/- 1.17	14.10 +/- 0.77	1.6M
ResNet	5.16 +/- 0.38	12.60 +/- 0.65	18.08 +/- 1.19	12.61 +/- 0.76	6.3M
Inception	4.53 +/- 0.34	13.69 +/- 0.67	15.43 +/- 1.03	11.56 +/- 0.68	22M

Appendix D

Language of Clips: Interview transcripts

In the context of our "Language of Music-Video Clips" study (Chapter 6), we interviewed three experts of the music to image industry. We propose here a complete transcript of each interview. All interviews were conducted in French, then translated. We used the Zoom video conferencing software. We asked each participant questions about Video/Music matching, focusing on three main aspects:

- The content (semantic) adequacy;
- The structure synchronization and temporal organization;
- Their personal relationship to the different technologies available in the music for the screen industry.

Questions were prepared ahead of each interview, some of them being common to all participants and some of them specific to certain professions. In addition to the prepared questions, we reported additional questions found during the interviews, as well as moments of free expression of the participants.

D.1 Jack Bartman, Composer (January, 27th, 2021)

Jack Bartman is a professional music composer who mostly composes music for screen. His main projects include commercials (Ducati, Nike, Apple, UbiSoft), short movies (The Walls between Us, Mab An Tarz), and documentaries (Samaritain, Stony Paths, Inner Wars).

Hello Jack and thank you for your time. We are currently studying the relationship between audio and video in musical videos (commercials, official music videos, movies, ...). Can you introduce yourself and how the music and video content interact in your projects?

> I often work with professional video editor Maxime Pozzi (see Section D.3). For him, music is very important for editing; when he's not satisfied with the music on a project, he calls me. He is capable of suggesting very unexpected and strong creations to his clients and they like it. Besides my work with Maxime, I often compose for commercials, when creative professionals fall in love with my music. For this case, what drives me is a general mood, an atmosphere. Trends matter too: sometimes you need to ride the wave, and sometimes you need to challenge them.

How is it like to work with directors and editors?

For commercials, I create custom-made music tracks. The projects are usually short and intense with a frenetic pace. Sometimes, I don't even get to see the images. I only get a briefing from the editor, which can sound like this: "The duration is 35 seconds, there is a woman walking down the street, a climax at 25 seconds and here are some tracks to get inspiration from". I compose using a set of keywords and my knowledge of the target brand. The mockups then go back and forth several times in short periods of time.

For movies, the process takes longer, and the production rhythm is more reasonable. I like to compose while the movie is being edited, and send tracks to the director and editor along the way. Following their preferences, some sequences will be edited on the music, while for others, the music will support the image.

D.1.1 V/M matching by semantic association

How do music and video influence each other in terms of genre, mood and so on?

Music and video content can be associated, but can also be used as contrasting elements. For example, in the commercial Crocodile Inside from Lacoste, there is a strong contrast between the music and the video content. The effect is surprising.

Are there any other constraints to take into account when composing music for video?

Yes: trends evolve quickly, and they need to be taken into account. This is applicable both for global trends and for the brand's identity.

D.1.2 V/M matching by synchronization

What role do you play when it comes to the dynamics of the video?

There are many ways to highlight change in the image: rhythm, sound design... For example, I can insert effects like "woosh" on cuts, to emphasize the video structure. It's my personal "trick" to support abrupt cuts. I will typically introduce those at the transition between two musical phrases, such that the music can still be listened to on its own. Other than that, I don't do the sound design (clothes sounds, glass sounds, etc).

Do you sometimes have issues synchronizing rhythmic sequences?

Yes! This year, I worked on a short movie about the psychological aspects of the Covid-19 lockdown. The tempo increases progressively and it took me quite a long time to synchronize everything properly. But after getting used to an imperfectly synchronized mockup soundtrack, the director did not want to use the final version, as the mockup would better suit the intended *madness* atmosphere.

To what extent does the image follow the structure of your music? Or on the opposite, to what extent do you have to follow the visual structure? Which situation happens the most?

Both scenarios exist. For commercials, I often have to adapt. Cuts matter, but only at some key moments. In the other case, when working with Maxime, I know that he likes to place the visual rupture right before or after the audio event, as it can sometimes work better.

How do you edit your music when you have to adapt to an existing movie?

Most of the time, I will use cuts. Slicing can happen at unconventional moments, like the first or last beat of a bar! I simply add a sound effect to make it work. Sometimes I do some time stretching, or I increase the tempo, it works too, but it's far less frequent.

When the brief is not precise enough, do you make a "catch-all" track (where the climax can be perceived at several moments) or do you take the risk with a strong structure from the beginning?

I will take the risk, even if I have to modify it afterward or make several submissions.

D.1.3 Technological considerations

What is the impact of technological tools from a composer's perspective? What were the major revolutions for you?

I am one of the only composers to use Reason, even though this software is not adapted at all for music for screen. I know how to use Logic and Protools as well, but I'm just far less efficient with those. Recently, a Reason plugin for video was released: it was obviously a mini revolution for me. Another recent option is to use Reason as a Logic plugin. However, this becomes very CPU intensive.

Have you ever used AI tools? Is there any AI software that you'd like to have?

Difficult question!

I have already used online automatic mastering tools such as Landr. The results are not fantastic, but it's convenient when time is short. More generally, if an AI could manage compression and equalization for me, that would be nice. Source separation tools can also be useful.

There are also creative AIs, like the ones that create chords from a single key. It's very powerful but a little sad as well. If the assistance is too efficient, then you can't learn by yourself, especially if you're self-taught like me.

My conclusion is that the most useful AIs won't make the most interesting tasks. It's best used for uninteresting work, if it can save time.

Today in AI, there are many embedding representations to compare presets, loops, samples which have opaque names. DJs enjoy AI systems to navigate presets with some serendipity. Do you use these as well?

Yes, the Native classification is very ergonomic for this. I don't use samples very often, but sometimes when the delay is very short, I navigate randomly in my samples and take whatever sounds good. It results in

a simple, spontaneous and original piece. I think people tend to always collect larger and larger libraries, but don't take the time to explore them and this results in sub-optimal exploitation of their resources. Searching and experimenting is crucial because sometimes what matters is simply using what we already have.

Thank you Jack, it was very informative!

D.2 Alexandre Courtès, Director (December, 9th, 2020)

Alexandre Courtès has been directing movies since the late 90s. His filmography includes official music videos (Phoenix, Kylie Minogue), feature films (The Incident, Les Infidèles) and commercials (Puma, Lexus, Paco Rabanne). In 2005, he received a Grammy Award for the clip "Vertigo" by U2.

Hello Alexandre and thank you for your time. We are currently studying the relationship between audio and video in musical videos (commercials, official music videos, movies, ...). Can you introduce yourself and how the music and video content interact in your projects?

I have directed a large number of official music videos. In this case, it's mostly in terms of rhythm that the music determines the video content. At editing time, cuts have to sync with the beat.

D.2.1 V/M matching by semantic association

When directing an OMV, what do you choose to show and why? To which extent can this choice depend on the music content in terms of genre, mood, etc?

It's left to our personal taste and appreciation: we can show horror pictures on classical music, there are no predefined rules. It's the director's job to decide.

There are different types of music videos (narration, performance, visual effects ...) and even directors who are specialized in one specific category. The type of OMV can be a requirement of the artist, or a suggestion from the director. Personally, I've seen every scenario: sometimes I am given a free rein, sometimes it's specific demands. But in most cases, directors arrive on the project with their ideas.

The song lyrics can provide a theme, and ideas to the director.

The music genre can have an impact too, typically for hardcore music, the rhythm will be wild, and often the editing will follow.

Music on its own creates an emotion, and so does the video. The music/video association creates a third emotion, or makes one emotion distort the other one.

Are there any similarities between OMVs and commercials from the direction point of view?

In both cases, we get to arrange several layers of material: image, color, actors, music ... For OMVs, the emotion carried by the visuals has to highlight the music. For commercials, the emotion has to highlight the product. It's the same exercise, but the other way around.

D.2.2 V/M matching by synchronization

To which extent does the video structure follow the music structure in OMVs?

> There are no general structure guidelines like we have for short movies. However, the music video will often show a performance, so it is similar to a musical comedy: it has to feature costumes, chapters, sets, acts.

How do you create dynamics in the video?

> Cuts have to follow the music's rhythm, even though they might not always co-occur with beats.
>
> In OMVs, there are other types of visual transitions to take into account: explosions, sudden appearance of large objects (eg, cars) in front of the camera, etc.
>
> For more precise details, you should ask someone specialized in editing.

Is there a notion of climax/salience like we have in commercials?

> Yes, we have this notion for OMVs too, but it's the music that decides when the climax happens. For movie soundtrack, it's the opposite (image determines the music's climax).

D.2.3 Technological considerations

What is the impact of available technologies on the directing and editing work? What were the most important technological revolutions for you?

> There were several important revolutions:
>
> - First, digital editing: it changed perspectives. Historically, editing was a meticulous job and thus placed under women's responsibility. When the first OMVs were produced for MTV, we were not on film anymore, but editing still required huge machines.
> - Then came computer-generated images, which made visual effects more accessible (for example with After Effect).
> - Finally, computers became mainstream. Today, you don't need to go to the studio anymore to edit movies, you can do everything from home. This is of course very convenient.

Have you ever used AI tools? Are there any AI-powered tools you'd like to use?

> No, I have never used AI tools but I'd like to. I am currently developing software to help film Formula One, and there should be some AI involved, but I can't say more about it for now.

Thank you for answering our questions!

D.3 Maxime Pozzi, Editor (January, 23rd, 2021)

Maxime Pozzi is a movie editor and photographer. He worked on big U.S. productions during what he calls the "golden age" of official music videos (early 2000s). In particular, he edited official music videos (Woodkid, Labrinth, Rihanna, Taylor Swift), feature films (Slalom, Savage State), short movies (Ellis, Paris Loves Us, Les Bosquets), and commercials (Lancôme, Nike, Armani).

Hello Maxime and thank you for your time. We are currently studying the relationship between audio and video in musical videos (commercials, official music videos, movies, ...). Can you introduce yourself and how music can impact video editing in your projects?

> Editors and musicians have a similar job, we all want the same thing: rhythm, narration, climaxes. To go further, we can even say that by editing music and image, we will manipulate the viewer's emotions. For OMVs, the goal is to promote the music. We create a video at the crossroads between commercial and fiction. There are many things to show in a short time (3 to 4 minutes). As an editor, we help create a world, and often tell a story. Intuition is important, as there are no universal rules. It helps me to listen to the music several times before starting.

In your opinion, are there different categories of OMV? Does it make sense to classify these into Dance, Performance/Concert, Concept/Abstract, Narration and so on? What do you think of the "Concept" category which can be heterogeneous (ie, include both urban music and Lady Gaga videos)? Does the editing technique vary according to these categories?

> Yes, there are indeed different possible approaches for making an OMV.

Playback: The artist is shown performing the song. This requires very precise synchronization of image with sound, and generates a lot of rushes. Playback is almost always present on OMVs, and it happens that the full video consists of playback. Otherwise, I have to find a balance between showing the performing artist and the other shots.

It can be challenging when singers are not perfectly in sync with the audio. They can easily be one or two seconds late, and up to ten seconds! We do our best to resynchronize everything, but sometimes it's just impossible to fix completely.

Narration: In *full story* OMVs, there are few singing shots. The goal is to tell a story.

Dance: There are often dance shots in OMVs, because music and dance go hand in hand. Dance can be used in several ways:

- in support, to showcase the artist or establish an atmosphere;
- in counterpoint, like in Sia's videos where dance is a full character to the story.

I edited a huge amount of dance videos, for example the short movie Les Bosquets featuring the NYC Ballet. You need to know how to "listen to the dance" in order to be able to edit it properly: taking the choreography as it, without adaptation, almost never works. In OMVs, the shooting is often imperfectly prepared, and dancers may lag behind. You have to resynchronize everything to outperform reality.

Mixtures thereof: There is frequently an alternation between story and playback, especially in US productions. As spectators, it seems obvious that we want to see both a nice story and our favorite artist singing.

The delicate case is when individual shots include both playback and story, like in Safe and Sound by Taylor Swift. This one was shot on a 35mm film. There are "in" moments that are part of the story while the singer is still performing. It was a real challenge for me to synchronize the different spaces well. Another example is the video "In Your Likeness" by Woodkid, where the music is very slow, it's a melancholic ballad. There is a story/playback tradeoff to find and the final result must be very fluid, smooth, seamless.

Concept: Different cultures have different approaches to "Concept" videos.

The *French approach* is minimalist, it leaves a lot of space for the music to unfold, but the final result can be boring.

The *U.S approach* represents the vast majority of OMVs. It consists in representing several sets alternatively. It's often a mixture of several genres because American artists are expected to be all-round: they sing, dance, model ... This style is often found in rap, hip-hop and urban music in general. Sets depict a mood, cuts are synchronized with onsets, the illustration is very straightforward. When the music is mainstream, it can afford a rough illustration.

There are many other approaches, for example *Scandinavia* has its own style and aesthetics.

What are the differences and common points between OMVs and commercials or movies?

> In the case of music for screen (commercials, full-length movies, short movies, documentaries), the approach is quite different. According to Michel Chion, music can be added to the soundtrack in two ways, either as a part of the scene (a radio playing), or without visual justification, as a support for the narration.
>
> The editor is central in the soundtrack creation, they make the transition between the composer and the director, while having their own vision too. Music and sound edits are the bulk of the work. We propose large amounts of soundtracks, way too many, and in the end we have to get rid of the majority of them.
>
> Nevertheless, both universes are linked, as they influence each other. Lars van Trier directed shots with ambitious slow-motion and this launched a worldwide slow motion trend in commercials. The OMV of Fever by Dua Lipa and Angèle uses the codes of the cinematographic industry (music integrated into the scene, embedded camera, dialogues, playback in and off, B-roll ...). The movie LaLaLand re-launched the American musical trend in OMVs and commercials. The aesthetic of the movie Grand Budapest Hotel (strong symmetry, pastel colors, feel-good jazz music) had a visible influence on OMVs and commercials. Conversely, Woodkid's spectacular OMVs had a big influence on commercials.

D.3.1 V/M matching by semantic association

When editing an OMV, what do you choose to show and why? To which extent can this choice depend on the music content in terms of genre, mood, etc?

In OMVs, editing makes the link between the music and the narration. On the audio side, the OMV track is rarely the same as on the radio. There can be additional dialogues, sound design, audio effects. For example the "underwater" filter is almost systematic in Justin Bieber's OMVs.

For movies, we first work with *influences*: we pick tracks from the chosen composer's catalog or from other movie soundtracks to make the first mockups. Once we have the final tracks, we make several versions, and we almost always end up swapping tracks that were intended for different sequences. For the western Savage State, the dialogue with the composer took months. There is a battle scene with women against men, and I asked for all the stems (more than 40 !) in order to adapt the mix to the shots (showing the men or women sides). In the movie Slalom, I combined energetic tracks for sports scenes and a more naturalistic atmosphere for the scenes about sexual abuse.

Are there any other constraints to take into account?

Sometimes, the mix of the track is not finished when the OMV is being made, so we have to edit several versions. This process can last up to one year for a Rihanna track!

D.3.2 V/M matching by synchronization

To which extent does the video structure follow the music structure?

Some elements are to all OMVs. Songs are generally composed of verses, choruses and bridges (or breaks). For verses and choruses, the visual dynamics follow the music's emotional climaxes. For example if the drum beat accelerates, the editing will follow. For bridges, which are transitions without lyrics, the editing is often slowed down, with a focus on the poetic aspect, the mood. For me, these are the funniest moments: I can play with gaze, with the atmosphere, make breaks with longer shots, use B-roll (e.g., backstage shooting), etc. This technique is particularly noticeable in Miley Cyrus' videos.

In a song, there are on average 3 choruses. Each time, we try to introduce a new element, typically a change of scenery. The edit shouldn't reveal everything from the beginning.

For full-length movies, we want to avoid the music to be always present, as this tends to stifle the atmosphere and duplicate the words of the image. Instead, we try to treat the music as an additional character, which can give clues from time to time. Famous examples include Inception and Elevator to the Gallows. There are codes to respect, for instance, it's hard to imagine a chase scene without music.

How do you create dynamics in the video? For example, we noticed an anticipation of cuts with respect to the downbeat in some OMVs.

In the music, there are often offsets and counterpoints. We emphasize those by playing on the looks, inserting shots on the musicians or effects like explosions.

In the Rolling in the deep video, off-beat cuts are used to create dynamics: to surprise the viewer, and illustrate the music's emotional climax. It's a common trick to draw back the viewer's attention, especially near the end of the video. It makes the video direction appear more "indie" as well, this can be required by the performing artists or the director. The effect is unsettling and creates suspense: the stalling effect makes the climax more intense, which is on point since the topic of the song is to overcome one's anxiety. The editing here is used as an element of narration and progression: it gets less and less rigid. On the other hand, there is little evolution of the sets, and the duration of each shot remains more or less constant. A similar process is used in the movies of the production company Canada, for example in Invisible Light by the Scissor Sisters.

Anticipation mechanisms of one modality with respect to the other one are quite complex, although intuitive. In OMVs, cuts are often observed before downbeats, maybe because downbeat is more steady and won't create surprise on its own, or because we have a long persistence of vision. In long movies, however, it's the opposite: music anticipates threats that appear later on screen.

In commercials, we generally have a 3-minute music track and a 30-second video. I quickly structure the music, and a specialized studio makes the rest. The final result does not depend only on me since they need several declinations (20 seconds, 8 seconds ...). For example, for Holiday Celebration by Lancôme, I created an intro, a first soft climax, then a rhythmic passage until the pack shot. The challenge is to be very precise so that people don't notice the edits on the music.

Are there any other constraints to take into account?

Artists often have special requests, and whatever happens, the final edit must satisfy them. Sometimes, they want the music to be like the metronome of the image, and both temporal structures must be in perfect adequacy (we call this style "mickey-mousing"). On the contrary, some artists don't like the cuts to be perfectly synchronized with the beat, they want something more original, that looks "indie".

D.3.3 Technological considerations

What is the impact of available technologies on editing techniques? What were the most important technological revolutions for you?

The Avid software, which is now the leader for full-length movies and TV. There is an online collaborative mode with an amazing AI which handles file synchronization and versioning. It's not a simple copy system, but a complex algorithm that softly combines the work of several editors. This is done using `.mxf` metadata, a unified standard for images, sound, music stems, etc. Refreshes occur almost every minute, and this allows to collaboratively edit from several countries. The technological achievement is impressive given that the considered amount of data is typically 1 to 2 Terabytes a day! Of course, to manipulate so much data, you will need a powerful machine. But with a recent Mac Pro, you can now easily work from home.

Have you ever used AI tools? Are there any AI-powered tools you'd like to use?

> Editing requires intuition, and therefore it cannot be fully automated. However, it requires smart tools to quickly process huge volumes of data. I use the Avid file synchronization AI on a daily basis. There are also quite convenient AIs in Adobe Premiere and Final Cut Pro, to sort rushes and synchronize the audio from different sources (camera, boom arm).
>
> AIs are thereby assistants, they will be used by every expert in the future, given that they are configurable and customizable. For now, their main limits are the lack of accuracy, the errors that are painful to fix, and the lack of subtlety. More importantly, they are not yet able to doubt, nor admit their errors or their ignorance.
>
> In my everyday life, AIs fascinate me, I'm not afraid of those. I've written a thesis on sci-fi and fear of the machines, from Charlie Chaplin to Terminator. Since then, "AI" by Spielberg narrated the opposite scenario, where the AI is more sensitive than humans.

Thank you for your answers, it was very informative!

Bibliography

Abu-El-Haija, Sami et al. (2016). "Youtube-8m: A large-scale video classification benchmark". In: *arXiv preprint arXiv:1609.08675*. URL: https://arxiv.org/abs/1609.08675.

Afouras, Triantafyllos et al. (Sept. 2018). "Deep Audio-Visual Speech Recognition". In: *IEEE transactions on pattern analysis and machine intelligence*. DOI: 10.1109/TPAMI.2018.2889052.

Akaike, Hirotogu (1992). "Information Theory and an Extension of the Maximum Likelihood Principle". In: *Springer Series in Statistics*, pp. 610–624. URL: https://doi.org/10.1007/978-1-4612-0919-5_38.

Aner, Aya and John R. Kender (2002). "Video Summaries through Mosaic-Based Shot and Scene Clustering". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Copenhagen, Denmark, pp. –. URL: https://doi.org/10.1007/3-540-47979-1_26.

Antoniadis, Pavlos and Frédéric Bevilacqua (2016). "Processing of symbolic music notation via multimodal performance data: Brian Ferneyhough's Lemma-Icon-Epigram for solo piano, phase 1". In: *Proceedings of the International Conference on Technologies for Music Notation and Representation*. Cambridge, UK. URL: https://core.ac.uk/download/pdf/144831428.pdf.

Arandjelovic, Relja and Andrew Zisserman (2017). "Look, Listen and Learn". In: *Proceedings of IEEE ICASSP (International Conference on Computer Vision)*. Venice, Italy. DOI: 10.1109/ICCV.2017.73.

Arandjelović, Relja and Andrew Zisserman (2018). "Objects that Sound". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Munich, Germany. URL: https://doi.org/10.1007/978-3-030-01246-5_27.

Aucouturier, Jean-Julien and Francois Pachet (2002). "Finding Songs That Sound The Same". In: *Proceedings of the of IEEE Benelux Workshop on Model based Processing and Coding of Audio*. Leuven, Belgium. URL: https://www.researchgate.net/publication/2909604.

Aytar, Yusuf, Carl Vondrick, and Antonio Torralba (2016). "SoundNet: Learning Sound Representations from Unlabeled Video". In: *Advances in Neural Information Processing Systems*, pp. 892–900. DOI: 10.1145/1753326.1753620.

— (2017). "See, Hear, and Read: Deep Aligned Representations". In: *arXiv preprint arXiv:1706.00932*. URL: http://arxiv.org/abs/1706.00932.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*. URL: https://arxiv.org/abs/1409.0473.

Balntas, Vassileios, Shuda Li, and Victor Prisacariu (2018). "RelocNet: Continuous Metric Learning Relocalisation using Neural Nets". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Munich, Germany. URL: https://doi.org/10.1007/978-3-030-01264-9_46.

Baraldi, Lorenzo, Costantino Grana, and Rita Cucchiara (2017). "Recognizing and presenting the storytelling video structure with deep multimodal networks". In:

IEEE Transactions on Multimedia 19.5, pp. 955–968. DOI: 10.1109/TMM.2016.2644872.

Bauer, Christine and Markus Schedl (June 2019). "Global and country-specific mainstreaminess measures: Definitions, analysis, and usage for improving personalized music recommendation systems". In: *PLOS ONE* 14.6. DOI: 10.1371/journal.pone.0217389.

Bertin-Mahieux, Thierry et al. (2011). "The Million Song Dataset". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Miami, FL, USA. URL: https://ismir2011.ismir.net/papers/OS6-1.pdf.

Bittner, Rachel M et al. (2017). "Deep Salience Representations for F0 Estimation in Polyphonic Music". In: *Proceedings of ISMIR (International Conference for Music Information Retrieval)*. Suzhou, China. URL: https://archives.ismir.net/ismir2017/paper/000085.pdf.

Böck, Sebastian, Florian Krebs, and Gerhard Widmer (2016). "Joint Beat and Downbeat Tracking with Recurrent Neural Networks." In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. New York City, NY, USA. URL: https://archives.ismir.net/ismir2016/paper/000186.pdf.

Böck, Sebastian et al. (2016). "Madmom: A new Python audio and music signal processing library". In: *Proceedings of ACM Multimedia*. New York City, NY, USA, pp. 1174–1178. DOI: 10.1145/2964284.2973795.

Bogdanov, Dmitry et al. (2013). "ESSENTIA: an Open-Source Libraryfor Sound and Music Analysis". In: *Proceedings of ACM Multimedia*. New York, NY, USA: ACM Press. DOI: 10.1145/2502081.2502229.

Bozzon, Alessandro et al. (2008). "A music recommendation system based on semantic audio segments similarity". In: *Proceedings of Internet and Multimedia Systems and Applications*. Innsbruck, Austria.

Bromley, Jane et al. (1994). "Signature Verification using a "Siamese" Time Delay Neural Network". In: *Machine Perception and Artificial Intelligence*, pp. 25–44. URL: https://doi.org/10.1142/9789812797926_0003.

Brown, Judith C. and Miller S. Puckette (Nov. 1992). "An efficient algorithm for the calculation of a constant Q transform". In: *The Journal of the Acoustical Society of America* 92.5. DOI: 10.1121/1.404385.

Burns, Lori A. and Stan Hawkins (2019). *The Bloomsbury Handbook of Popular Music Video Analysis*. Bloomsbury Publishing USA.

Cemgil, Ali Taylan et al. (2000). "On tempo tracking: Tempogram representation and Kalman filtering". In: *Journal of New Music Research* 29.4, pp. 259–273. URL: https://www.mcg.uva.nl/papers/mmm-26.pdf.

Charbuillet, Christophe et al. (2011). "GMM supervector for content based music similarity". In: *Proceedings of DAFx (International Conference on Digital Audio Effects)*. Paris, France. URL: https://www.dafx.de/paper-archive/2011/Papers/98_e.pdf.

Chen, Yangyu et al. (2018). "Less Is More: Picking Informative Frames for Video Captioning". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Munich, Germany. URL: https://yugnaynehc.github.io/picknet.

Cheng, Zhiyong and Jialie Shen (Apr. 2016). "On Effective Location-Aware Music Recommendation". In: *ACM Transactions on Information Systems* 34.2, pp. 1–32. DOI: 10.1145/2846092.

Choi, Keunwoo, George Fazekas, and Mark Sandler (2016a). "Automatic tagging using deep convolutional neural networks". In: *Proceedings. of ISMIR (International Society for Music Information Retrieval Conference)*. Suzhou, China. URL: https://archives.ismir.net/ismir2016/paper/000009.pdf.

— (2016b). "Explaining Deep Convolutional Neural Networks on Music Classification". In: *arXiv preprint arXiv:1607.02444*. URL: https://arxiv.org/abs/1607.02444.

Chollet, François and others (2015). *Keras: Deep learning library for theano and tensorflow*. URL: https://keras.io/.

Choroś, Kazimierz (2018). "Video Genre Classification Based on Length Analysis of Temporally Aggregated Video Shots". In: *Computational Collective Intelligence. Lecture Notes in Computer Science*. Vol. 11056. Springer, Cham. URL: https://doi.org/10.1007/978-3-319-98446-9_48.

Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter (2016). "Fast and accurate deep network learning by exponential linear units (ELUs)". In: *Proceedings of ICLR (International Conference on Learning Representations)*. San Juan, Puerto Rico, USA. URL: https://arxiv.org/abs/1511.07289.

Clifford, Stephanie (2007). "Pandora's Long Strange Trip". In: *Inc.com*. URL: https://www.inc.com/magazine/20071001/pandoras-long-strange-trip.html.

Cohen-Hadria, Alice (2019). "Music Description Estimation with Deep Learning". PhD thesis. IRCAM.

Costa, Yandre M.G., Luiz S. Oliveira, and Carlos N. Silla (2017). "An evaluation of Convolutional Neural Networks for music classification using spectrograms". In: *Applied Soft Computing Journal* 52, pp. 28–38. DOI: 10.1016/j.asoc.2016.12.024.

Cramer, Jason et al. (2019). "Look, Listen, and Learn More: Design Choices for Deep Audio Embeddings". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Brighton, UK. DOI: 10.1109/icassp.2019.8682475.

Cunha, Renato L. F., Evandro Caldeira, and Luciana Fujii (2017). "Determining Song Similarity via Machine Learning Techniques and Tagging Information". In: *arXiv preprint arXiv:1704.03844*. URL: https://arxiv.org/abs/1704.03844.

Defferrard, Michaël et al. (2017). "FMA: A Dataset for Music Analysis". In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. Suzhou, China. URL: https://archives.ismir.net/ismir2017/paper/000075.pdf.

Defferrard, Michaël et al. (2018). "Learning to Recognize Musical Genre from Audio". In: *Companion of WWW (The Web Conference)*. New York City, NY, USA, pp. –. DOI: 10.1145/3184558.3192310.

Dieleman, Sander (2014). *Recommending music on Spotify with deep learning*. URL: http://benanne.github.io/2014/08/05/spotify-cnns.html.

Dieleman, Sander and Benjamin Schrauwen (2014). "End-to-end learning for music audio". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Florence, Italy. DOI: 10.1109/ICASSP.2014.6854950.

Doras, Guillaume and Geoffroy Peeters (2019). "Cover Detection Using Dominant Melody Embeddings". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Delft, The Netherlands. URL: https://archives.ismir.net/ismir2019/paper/000010.pdf.

Doras, Guillaume et al. (2020). "Combining musical features for cover detection". In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. Montréal, Canada. URL: https://archives.ismir.net/ismir2020/paper/000239.pdf.

Downie, J. Stephen (2005). *Music Information Retrieval Evaluation eXchange*. Tech. rep. URL: http://www.music-ir.org/mirex/wiki/MIREX_HOME.

Eck, Douglas et al. (2008). "Automatic Generation of Social Tags for Music Recommendation". In: *Proceedings of NIPS (Neural Information Processing Systems)*.

Vancouver, BC, Canada. URL: https://papers.nips.cc/paper/2007/file/c2aee86157b4a40b78132f1e71a9e6f1-Paper.pdf.

Ellis, Daniel P.W. (2007). *The "covers80" cover song data set*. URL: http://labrosa.ee.columbia.edu/projects/coversongs/covers80.

Ferraro, Andres et al. (2021). "How Low Can You Go? Reducing Frequency and Time Resolution in Current CNN Architectures for Music Auto-tagging". In: *Proceedings of EUSIPCO (European Signal Processing Conference)*. Amsterdam, The Netherlands. DOI: 10.23919/Eusipco47968.2020.9287769.

Foote, Jonathan (2000). "Automatic audio segmentation using a measure of audio novelty". In: *Proceedings of ICME (International Conference on Multimedia and Expo)*. New York City, NY, USA. DOI: 10.1109/ICME.2000.869637.

Fukushima, Kunihiko (1980). "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". In: *Biological Cybernetics* 36.4, pp. 193–202. DOI: 10.1007/BF00344251.

Gabbolini, Giovanni and Derek Bridge (2021). "An Interpretable Music Similarity Measure Based on Path Interestingness". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Online event. URL: https://archives.ismir.net/ismir2021/paper/000026.pdf.

Gabeur, Valentin et al. (2020). "Multi-modal Transformer for Video Retrieval". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Glasgow, UK, pp. 214–229. URL: https://doi.org/10.1007/978-3-030-58548-8_13.

Gemmeke, Jort F. et al. (2017). "Audio Set: An Ontology and Human-Labeled Dataset for Audio Events". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. New Orleans, LA, USA. DOI: 10.1109/icassp.2017.7952261.

George Awad et al. (2021). "Evaluating Multiple Video Understanding and Retrieval Tasks at TRECVID 2021". In: *Proceedings of TRECVID*. NIST, USA. URL: http://www-nlpir.nist.gov/projects/tvpubs/tv21.papers/tv21overview.pdf.

Gillet, Olivier, Slim Essid, and Gael Richard (2007). "On the correlation of automatic audio and visual segmentations of music videos". In: *IEEE Transactions on Circuits and Systems for Video Technology* 17.3, pp. 347–355. DOI: 10.1109/TCSVT.2007.890831.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. MIT Press. URL: http://www.deeplearningbook.org.

Goto, Masataka (2006). "A chorus section detection method for musical audio signals and its application to a music listening station." In: *IEEE Transactions on Audio, Speech, and Language Processing* 14.5, pp. 1783–1794. DOI: 10.1109/tsa.2005.863204.

Gowing, Marc et al. (2011). "Enhanced visualisation of dance performance from automatically synchronised multimodal recordings". In: *Proceedings of ACM MM (International conference on Multimedia)*. Scottsdale, AZ, USA, p. 667. DOI: 10.1145/2072298.2072414.

Grollmisch, Sascha et al. (2020). "Analyzing the Potential of Pre-Trained Embeddings for Audio Classification Tasks". In: *Proceedings of EUSIPCO (European Signal Processing Conference)*. Amsterdam, the Netherlands. DOI: 10.23919/Eusipco47968.2020.9287743.

Hadsell, Raia, Sumit Chopra, and Yann LeCun (2006). "Dimensionality Reduction by Learning an Invariant Mapping". In: *Proceedins of CVPR (Computer Society Conference on Computer Vision and Pattern Recognition)*. New York, NY, USA, pp. 1735–1742. DOI: 10.1109/CVPR.2006.100.

He, Kaiming et al. (2016). "Deep Residual Learning for Image Recognition". In: *Proceedings of CVPR (Conference on Computer Vision and Pattern Recognition)*. Las Vegas, NV, USA. DOI: 10.1109/CVPR.2016.90.

Hennequin, Romain, Jimena Royo-Letelier, and Manuel R. Moussallam (2018). "Audio Based Disambiguation of Music Genre Tags". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. URL: https://archives.ismir.net/ismir2018/paper/000163.pdf.

Hermans, Alexander, Lucas Beyer, and Bastian Leibe (2017). "In Defense of the Triplet Loss for Person Re-Identification". In: *arXiv preprint arXiv:1703.07737*. URL: http://arxiv.org/abs/1703.07737.

Hershey, Shawn et al. (2017). "CNN architectures for large-scale audio classification". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. New Orleans, LA, USA. DOI: 10.1109/ICASSP.2017.7952132.

Hoffer, Elad and Nir Ailon (2015). "Deep metric learning using triplet network". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 9370. Springer Verlag, pp. 84–92. URL: https://doi.org/10.1007/978-3-319-24261-3_7.

Hong, Sungeun, Woobin Im, and Hyun S. Yang (2018). "CBVMR: Content-based video-music retrieval using soft intra-modal structure constraint". In: *Proceedings of ACM ICMR (International Conference on Multimedia Retrieval)*. Yokohama, Japan. DOI: 10.1145/3206025.3206046.

Horak, Karel and Robert Sablatnig (2019). "Deep learning concepts and datasets for image recognition: Overview 2019". In: *Proceedings of ICDIP (International Conference on Digital Image Processing)*. Guangzhou, China. DOI: 10.1117/12.2539806.

Hsia, Chih-Chun et al. (Aug. 2018). "Representation Learning for Image-based Music Recommendation". In: *Proceedings of the Late-Breaking Results of ACM RecSys (Conference on Recommender Systems)*. URL: http://arxiv.org/abs/1808.09198.

Hsueh, Gigi (2020). "Carbon Footprint of Machine Learning Algorithms". PhD thesis. Bard College. URL: https://digitalcommons.bard.edu/senproj_s2020/296.

Hu, Di et al. (2020). "Curriculum audiovisual learning". In: *arXiv preprint arXiv:2001.09414*. URL: https://arxiv.org/abs/2001.09414.

Hu, Di et al. (2021). "Discriminative sounding objects localization via self-supervised audiovisual matching". In: *Advances in Neural Information Processing Systems* 33. URL: https://proceedings.neurips.cc/paper/2020/hash/7288251b27c8f0e73f4d7f483b06a785-Abstract.html.

Hua, Xian-Sheng, Lie Lu, and Hong-Jiang Zhang (2004). "Automatic music video generation based on temporal pattern analysis". In: *Proceedings of ACM Multimedia*. New York City, NY, USA. DOI: 10.1145/1027527.1027641.

Inskip, Charlie, Andy Macfarlane, and Pauline Rafferty (2008). "Music, Movies and Meaning: Communication in Film-makers' Search for Pre-existing Music, and the Implications for Music Information Retrieval". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Philadelphia, PA, USA. URL: https://archives.ismir.net/ismir2008/paper/000117.pdf.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *Proceedings of ICML (International Conference on Machine Learning)*. Lille, France. URL: http://proceedings.mlr.press/v37/ioffe15.html.

Jia, Bijue, Jiancheng Lv, and Dayiheng Liu (Dec. 2019). "Deep learning-based automatic downbeat tracking: a brief review". In: *Multimedia Systems* 25.6. DOI: 10.1007/s00530-019-00607-x.

Kaminskas, Marius and Francesco Ricci (2011). "Location-Adapted Music Recommendation Using Tags". In: *International conference on user modeling, adaptation, and personalization*. Springer, pp. 183–194.

Kim, Jaehun et al. (2020). "One Deep Music Representation to Rule Them All? : A comparative analysis of different representation learning strategies". In: *Neural Computing and Applications* 32.4, pp. 1067–1093. DOI: 10.1007/s00521-019-04076-1.

Kim, Taejun, Jongpil Lee, and Juhan Nam (2018). "Sample-Level CNN Architectures for Music Auto-Tagging Using Raw Waveforms". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. DOI: 10.1109/ICASSP.2018.8462046.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *Proceedings of ICLR (International Conference on Learning Representations)*. San Diego, CA, USA. URL: https://arxiv.org/abs/1412.6980.

Koelstra, Sander et al. (2012). "DEAP: A Database for Emotion Analysis Using Physiological Signals". In: *IEEE Transactions on Affective Computing, Special Issue on Naturalistic Affect Resources for System Building and Evaluation* 3.1. DOI: 10.1109/T-AFFC.2011.15.

Korzeniowski, Filip, Sergio Oramas, and Fabien Gouyon (2021). "Artist Similarity with Graph Neural Networks". In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. Online. URL: https://archives.ismir.net/ismir2021/paper/000043.pdf.

Krebs, Florian, Sebastian Böck, and Gerhard Widmer (2013). "Rhythmic Pattern Modeling for Beat and Downbeat Tracking in Musical Audio". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Curitiba, PR, Brazil. URL: https://archives.ismir.net/ismir2013/paper/000051.pdf.

Kroher, Nadine et al. (May 2016). "Corpus COFLA: a research corpus for the computational study of flamenco music." In: *Journal on Computing and Cultural Heritage (JOCCH)* 9.2. DOI: 10.1145/2875428.

Kumar, Anurag, Maksim Khadkevich, and Christian Fugen (2018). "Knowledge Transfer from Weakly Labeled Audio Using Convolutional Neural Network for Sound Events and Scenes". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Calgary, AB, Canada. DOI: 10.1109/ICASSP.2018.8462200.

Kuo, Fang-Fei, Man-Kwan Shan, and Suh-Yin Lee (2013). "Background Music Recommendation for Video Based on Multimodal Latent Semantic Analysis". In: *Proceedings of ICME (International Conference on Multimedia and Expo)*. San Jose, CA, USA. DOI: 10.1109/icme.2013.6607444.

Lacoste, Alexandre et al. (2019). "Quantifying the Carbon Emissions of Machine Learning". In: *Workshop on Tackling Climate Change with Machine Learning at NeurIPS*. Vancouver, Canada. URL: https://arxiv.org/abs/1910.09700.

Law, Edith et al. (2009). "Evaluation of Algorithms Using Games: The Case of Music Tagging". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Kobe, Japan. URL: https://ismir2009.ismir.net/proceedings/OS5-5.pdf.

LeCun, Yann and Yoshua Bengio (1995). "Convolutional networks for images, speech, and time series". In: *The handbook of brain theory and neural networks* 3361.10. URL: http://www.iro.umontreal.ca/~lisa/pointeurs/handbook-convo.pdf.

Lerch, Alexander. *Audio Content Analysis*. URL: https://www.audiocontentanalysis.org/data-sets/.

Li, Bochen and Aparna Kumar (2019). "Query by Video: Cross-Modal Music Retrieval". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Delft, The Netherlands. URL: https://archives.ismir.net/ismir2019/paper/000073.pdf.

Li, Bochen et al. (Feb. 2019a). "Creating a Multitrack Classical Music Performance Dataset for Multimodal Music Analysis: Challenges, Insights, and Applications". In: *IEEE Transactions on Multimedia* 21.2. DOI: 10.1109/TMM.2018.2856090.

Li, Bochen et al. (2019b). "Online Audio-Visual Source Association for Chamber Music Performances". In: *Transactions of the International Society for Music Information Retrieval* 2.1. DOI: 10.5334/tismir.25.

Liao, Chao, Patricia P. Wang, and Yimin Zhang (2009). "Mining Association Patterns between Music and Video Clips in Professional MTV". In: *Proceedings of MMM (International Conference on Multimedia Modeling)*. Sophia Antipolis, France. URL: https://doi.org/10.1007/978-3-540-92892-8_41.

Liem, Cynthia C. S., Martha Larson, and Alan Hanjalic (Mar. 2013). "When music makes a scene". In: *International Journal of Multimedia Information Retrieval* 2.1, pp. 15–30. DOI: 10.1007/s13735-012-0031-3.

Lin, Diana and Sampath Jayarathna (2018). "Automated Playlist Generation from Personal Music Libraries". In: *Proceedings of IRI (International Conference on Information Reuse and Integration)*. Salt Lake City, UT, USA. DOI: 10.1109/IRI.2018.00039.

Lin, Jen Chun, Wen Li Wei, and Hsin Min Wang (2015). "EMV-matchmaker: Emotional temporal course modeling and matching for automatic music video generation". In: *Proceedings of ACM Multimedia*. Brisbane, Australia, pp. 899–902. DOI: 10.1145/2733373.2806359.

Lin, Jen Chun et al. (2017). "Automatic music video generation based on simultaneous soundtrack recommendation and video editing". In: *Proceedings of MMM (International Conference on Multimedia Modeling)*. Reykjavik, Iceland. DOI: 10.1145/3123266.3123399.

Lottick, Kadan et al. (2019). "Energy Usage Reports: Environmental awareness as part of algorithmic accountability". In: *Workshop on Tackling Climate Change with Machine Learning at NeurIPS*. Vancouver, Canada. URL: https://arxiv.org/abs/1911.08354.

Lu, Rui et al. (2017). "Deep ranking: Triplet MatchNet for music metric learning". In: *Proceedings of IEEE ICASSP (International Conference on Computer Vision)*. New Orleans, LA, USA. DOI: 10.1109/ICASSP.2017.7952130.

Mallat, Stéphane (2020). *Modèles multi-échelles et réseaux de neurones convolutifs*. Course at the Collège de France. URL: https://www.college-de-france.fr/site/stephane-mallat/course-2019-2020.htm.

Manmatha, R. et al. (2017). "Sampling Matters in Deep Embedding Learning". In: *Proceedings of ICCV (International Conference on Computer Vision)*. Venice, Italy. DOI: 10.1109/ICCV.2017.309.

McCallum, Matthew C. (2019). "Unsupervised Learning of Deep Features for Music Segmentation". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Brighton, UK: IEEE. DOI: 10.1109/ICASSP.2019.8683407.

McFee, Brian, Luke Barrington, and Gert Lanckriet (2012). "Learning Content Similarity for Music Recommendation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 20.8, pp. 2207 –2218. DOI: 10.1109/TASL.2012.2199109.

McFee, Brian and Daniel P.W. Ellis (2014). "Learning to segment songs with ordinal linear discriminant analysis". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Florence, Italy, pp. 5197–5201. DOI: 10.1109/icassp.2014.6854594.

Mcfee, Brian and Gert Lanckriet (2010). "Metric Learning to Rank". In: *Proceedings of ICML (International Conference on Machine Learning)*. Haifa, Israel. URL: https://icml.cc/Conferences/2010/papers/504.pdf.

McFee, Brian, Justin Salamon, and Juan Pablo Bello (2018). "Adaptive pooling operators for weakly labeled sound event detection". In: *IEEE/ACM Transactions on Audio, Speech and Language Processing* 26.11, pp. 2180–2193. URL: https://par.nsf.gov/servlets/purl/10074710.

McFee, Brian et al. (2015). "librosa: Audio and music signal analysis in python". In: *Proceedings of Python in Science*. Austin, TX, USA. DOI: 10.25080/majora-7b98e3ed-003.

Mishchuk, Anastasiya et al. (2017). "Working hard to know your neighbor's margins: Local descriptor learning loss". In: *Proceedings of NIPS (Neural Information Processing Systems)*. Long Beach, CA, USA. URL: http://arxiv.org/abs/1705.10872.

Mitsufuji, Yuki et al. (2021). "Music Demixing Challenge at ISMIR". In: *arXiv preprint*. URL: https://arxiv.org/abs/2108.13559.

Mulhem, Philippe et al. (Apr. 2003). "Pivot vector space approach for audio-video mixing". In: *IEEE Multimedia* 10.2, pp. 28–40. DOI: 10.1109/MMUL.2003.1195159.

Müller, Meinard (2015). *Fundamentals of Music Processing*. Springer International Publishing. DOI: 10.1007/978-3-319-21945-5.

Müller, Meinard et al. (2019). "Cross-Modal Music Retrieval and Applications: An Overview of Key Methodologies". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Brighton, UK. DOI: 10.1109/MSP.2018.2868887.

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of ICML (International Conference on Machine Learning)*. Haifa, Israel. URL: https://icml.cc/Conferences/2010/papers/432.pdf.

Needleman, Saul B. and Christian D. Wunsch (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". In: *Journal of Molecular Biology* 48.3, pp. 443–453. DOI: 10.1016/b978-0-12-131200-8.50031-9.

Ngiam, Jiquan et al. (2011). "Multimodal Deep Learning". In: *Proceedings of ICML (International Conference on Machine Learning)*. Bellevue, WA, USA. URL: https://icml.cc/2011/papers/399_icmlpaper.pdf.

Nieto, Oriol and Juan Pablo Bello (2016). "Systematic Exploration Of Computational Music Structure Research." In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. New York City, NY, USA. URL: https://archives.ismir.net/ismir2016/paper/000043.pdf.

Nieto, Oriol et al. (2019). "The Harmonix Set: Beats, Downbeats, and Functional Segment Annotations of Western Popular Music". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Delft, The Netherlands. URL: https://archives.ismir.net/ismir2019/paper/000068.pdf.

Nieto, Oriol et al. (2020). "Audio-Based Music Structure Analysis: Current Trends, Open Challenges, and Applications". In: *Transactions of the International Society for Music Information Retrieval* 3.1. DOI: 10.5334/tismir.54.

Noulas, Athanasios, Gwenn Englebienne, and Ben J.A. Kröse (2012). "Multimodal Speaker diarization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. DOI: 10.1109/TPAMI.2011.47.

Oord, Aaron van den, Sander Dieleman, and Benjamin Schrauwen (2013). "Deep content-based music recommendation". In: *Proceedings of NIPS (Neural Information Processing Systems)*. Lake Tahoe, NV, USA. URL: http://papers.nips.cc/paper/5004-deep-content-based-.

Oquab, Maxime et al. (2014). "Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks". In: *Proceedings of IEEE CVPR (Conference on Computer Vision and Pattern Recognition)*. Columbus, OH, USA. DOI: 10.1109/cvpr.2014.222.

Owens, Andrew and Alexei A Efros (2018). "Audio-Visual Scene Analysis with Self-Supervised Multisensory Features". In: *Proceedings of IEEE ECCV (European Conference on Computer Vision)*. Salt Lake City, UT, USA. URL: https://doi.org/10.1007/978-3-030-01231-1_39.

Owens, Andrew et al. (2016). "Ambient sound provides supervision for visual learning". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Amsterdam, The Netherlands. URL: https://doi.org/10.1007/978-3-319-46448-0_48.

Parekh, Sanjeel (2019). "Learning representations for robust audio-visual scene analysis". PhD thesis. Université Paris-Saclay, Télécom ParisTech.

Parekh, Sanjeel et al. (2019). "Weakly Supervised Representation Learning for Audio-Visual Scene Analysis". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. DOI: 10.1109/taslp.2019.2957889.

Peeters, Geoffroy and Helene Papadopoulos (Aug. 2011). "Simultaneous Beat and Downbeat-Tracking Using a Probabilistic Framework: Theory and Large-Scale Evaluation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.6, pp. 1754–1769. DOI: 10.1109/TASL.2010.2098869.

Pilehvar, Mohammad Taher and Jose Camacho-Collados (2020). "Embeddings in natural language processing: Theory and advances in vector representations of meaning". In: *Synthesis Lectures on Human Language Technologies* 13.4. URL: http://josecamachocollados.com/book_embNLP_draft.pdf.

Pons, Jordi and Xavier Serra (2019). "musicnn: Pre-trained convolutional neural networks for music audio tagging". In: *Late Breaking Demo, ISMIR (International Conference on Music Information Retrieval)*. Delft, The Netherlands. URL: https://arxiv.org/abs/1909.06654.

Pons, Jordi et al. (2017). "Timbre analysis of music audio signals with convolutional neural networks". In: *Proceedings of EUSIPCO (European Signal Processing Conference)*. Kos, Greece. DOI: 10.23919/EUSIPCO.2017.8081710.

Pons, Jordi et al. (2018). "End-to-end learning for music audio tagging at scale". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Paris, France. URL: https://archives.ismir.net/ismir2018/paper/000191.pdf.

Prétet, Laure, Gaël Richard, and Geoffroy Peeters (2020). "Learning to Rank Music Tracks Using Triplet Loss". In: *Proceedings of IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Barcelona, Spain. DOI: 10.1109/icassp40776.2020.9053135.

— (2021a). "Cross-Modal Music-Video Recommendation: A Study of Design Choices". In: *Special Session on RLASMP, IJCNN (International Joint Conference on Neural Networks)*. Virtual Event. DOI: 10.1109/IJCNN52387.2021.9533662.

Prétet, Laure, Gaël Richard, and Geoffroy Peeters (2021b). “Is there a "language of music-video clips"? A qualitative and quantitative study”. In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Virtual Event. URL: https://archives.ismir.net/ismir2021/paper/000067.pdf.

Prétet, Laure et al. (2021). *Procédé d'entraînement d'un modèle statistique pour qu'il soit configuré pour être utilisé pour recommander, à partir d'un média d'un premier type, un média d'un deuxième type, et système associé*.

— (2022). “Video-to-Music Recommendation using Temporal Alignment of Segments”. In: *IEEE Transactions on Multimedia*. DOI: 10.1109/TMM.2022.3152598.

Priyadarshini, Sushree Bibhuprada B., Amiya Bhusan Bagjadab, and Brojo Kishore Mishra (Nov. 2020). “A Brief Overview of Natural Language Processing and Artificial Intelligence”. In: *Natural Language Processing in Artificial Intelligence*. Includes bibliographical references and index. DOI: 10.1201/9780367808495-8.

Raffel, Colin and Daniel P.W. Ellis (2015). “Feed-forward networks with attention can solve some long-term memory problems”. In: *arXiv preprint arXiv:1512.08756*. URL: https://arxiv.org/abs/1512.08756.

Ravanelli, Mirco and Yoshua Bengio (Dec. 2018). “Speaker Recognition from Raw Waveform with SincNet”. In: *Proceedings of SLT (Spoken Language Technology Workshop)*. Athens, Greece. DOI: 10.1109/SLT.2018.8639585.

Ruiduo Yang and M.S. Brown (2004). “Music database query with video by synesthesia observation”. In: *Proceedings of ICME (International Conference on Multimedia and Expo)*. Taipei, Taiwan: IEEE, pp. –. DOI: 10.1109/ICME.2004.1394186.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (Oct. 1986). “Learning representations by back-propagating errors”. In: *Nature* 323.6088. DOI: 10.1038/323533a0.

Sarasúa, Álvaro et al. (2017). “Datasets for the Analysis of Expressive Musical Gestures”. In: *Proceedings of MOCO (International Conference on Movement Computing)*. London, UK. DOI: 10.1145/3077981.3078032.

Sargent, Gabriel et al. (2016). “A scalable summary generation method based on cross-modal consensus clustering and OLAP cube modeling”. In: *Multimedia Tools and Applications* 75.15, pp. 9073–9094. DOI: 10.1007/s11042-015-2863-3.

Sasaki, Shoto et al. (2015). “Affective Music Recommendation System Based on the Mood of Input Video”. In: *Proceedings of MMM (International Conference on Multimedia Modeling)*. Sydney, Australia. DOI: 10.1109/culturecomputing.2013.42.

Schein, Andrew I. et al. (2002). “Methods and metrics for cold-start recommendations”. In: *Proceedings of ACM SIGIR (Conference on Research and Development in Information Retrieval)*. Tampere, Finland. DOI: 10.1145/564376.564421.

Schindler, Alexander (2019). “Multi-Modal Music Information Retrieval: Augmenting Audio-Analysis with Visual Computing for Improved Music Video Analysis”. PhD thesis. Technische Universität Wien. URL: https://arxiv.org/abs/2002.00251.

Schindler, Alexander and Andreas Rauber (2015). “An Audio-Visual Approach to Music Genre Classification through Affective Color Features”. In: *European conference on information retrieval*. Vienna, Austria. URL: https://doi.org/10.1007/978-3-319-16354-3_8.

— (2016). “Harnessing music-related visual stereotypes for music information retrieval”. In: *ACM Transactions on Intelligent Systems and Technology* 8.2, pp. 1–21. DOI: 10.1145/2926719.

— (2019). “On the Unsolved Problem of Shot Boundary Detection for Music Videos”. In: *Proceedings of MMM (International Conference on Multimedia Modeling)*. Vol. 11295

LNCS. Thessaloniki, Greece: Springer Verlag. URL: https://doi.org/10.1007/978-3-030-05710-7_43.

Schroff, Florian and James Philbin (2015). "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: *Proceedings of IEEE CVPR (Conference on Computer Vision and Pattern Recognition)*. Boston, MA, USA. DOI: 10.1109/cvpr.2015.7298682.

Serra, Joan et al. (2012). "Unsupervised Detection of Music Boundaries by Time Series Structure Features". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Toronto, ON, Canada. URL: https://ojs.aaai.org/index.php/AAAI/article/view/8328.

Shah, Rajiv Ratn, Yi Yu, and Roger Zimmermann (2014). "ADVISOR - Personalized video soundtrack recommendation by late fusion with heuristic rankings". In: *Proceedings of ACM Multimedia*. Orlando, FL, USA. DOI: 10.1145/2647868.2654919.

Shang, Lanyu et al. (Dec. 2021). "CCMR: A Classic-enriched Connotation-aware Music Retrieval System on Social Media with Visual Inputs". In: *Social Network Analysis and Mining* 11.1, p. 119. DOI: 10.1007/s13278-021-00821-4.

Shin, Ki-Ho and In-Kwon Lee (2017). "Music synchronization with video using emotion similarity". In: *Proceedings of IEEE BigComp (International Conference on Big Data and Smart Computing)*. Jeju Island, South Korea. DOI: 10.1109/bigcomp.2017.7881714.

Shuhei Tsuchida et al. (2019). "AIST Dance Video Database: Multi-genre, Multi-dancer, and Multi-camera Database for Dance Information Processing". In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. Delft, Netherlands.

Slaney, Malcolm, Kilian Weinberger, and William White (2008). "Learning a Metric for Music Similarity". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Philadelphia, PA, USA. URL: https://archives.ismir.net/ismir2008/paper/000148.pdf.

Smeaton, Alan F., Paul Over, and Wessel Kraaij (2006). "Evaluation campaigns and TRECVid". In: *Proceedings of ACM International Workshop on Multimedia Information Retrieval*. Santa Barbara, CA, USA: National Institute of Standards and Technology. DOI: 10.1145/1178677.1178722. URL: https://trecvid.nist.gov/.

Smith, Temple F. and Michael S. Waterman (1981). "Identification of common molecular subsequences". In: *Journal of molecular biology* 147.1, pp. 195–197. URL: http://www.gersteinlab.org/courses/452/09-spring/pdf/sw.pdf.

Souček, Tomáš, Jaroslav Moravec, and Jakub Lokoč (2019). "TransNet: A deep network for fast detection of common shot transitions". In: *arXiv preprint arXiv:1906.03363*. URL: https://arxiv.org/abs/1906.03363.

Srivastava, Nitish et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1, pp. 1929–1958. URL: https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf.

Stoller, Daniel (2020). "Deep Learning for Music Information Retrieval in Limited Data Scenarios". PhD thesis. Queen Mary University of London. URL: https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.820536.

Su, Han et al. (2013). "MediaEval 2013: Soundtrack Selection for Commercials Based on Content Correlation Modeling". In: *MediaEval Multimedia Benchmark Workshop*. Barcelona, Spain. URL: http://ceur-ws.org/Vol-1043/mediaeval2013_submission_98.pdf.

Surís, Didac et al. (2018). "Cross-modal Embeddings for Video and Audio Retrieval". In: *Proceedings of ECCV Workshops (European Conference on Computer Vision)*. Munich, Germany. URL: https://doi.org/10.1007/978-3-030-11018-5_62.

Szegedy, Christian et al. (2015). "Going deeper with convolutions". In: *Proceedings of CVPR (Conference on Computer Vision and Pattern Recognition)*. Boston, MA, USA. DOI: 10.1109/CVPR.2015.7298594.

Tian, Yapeng et al. (2018). "Audio-visual event localization in unconstrained videos". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Munich, Germany. URL: https://doi.org/10.1007/978-3-030-01216-8_16.

Tingle, Derek, Youngmoo E. Kim, and Douglas Turnbull (2010). "Exploring Automatic Music Annotation with Acoustically-Objective Tags". In: *Proceedings of MIR (International conference on Multimedia Information Retrieval)*. Philadelphia, PA, USA. DOI: 10.1145/1743384.1743400.

Urbano, Julián (2013). "Evaluation in Audio Music Similarity". PhD thesis. Universidad Carlos III de Madrid. URL: https://e-archivo.uc3m.es/bitstream/handle/10016/18198/tesis_julian_urbano_merino_2013.pdf?sequence=1.

Vaglio, Andrea et al. (2021). "The Words Remain the Same: Cover Detection with Lyrics Transcription". In: *Proceedings of ISMIR (International Conference for Music Information Retrieval)*. Online. URL: https://archives.ismir.net/ismir2021/paper/000089.pdf.

Varghese, Jina and K. N. Ramachandran Nair (2019). "A Novel Video Genre Classification Algorithm by Keyframe Relevance". In: *Information and Communication Technology for Intelligent Systems. Smart Innovation, Systems and Technologies*. Vol. 106. Singapore: Springer. URL: https://doi.org/10.1007/978-981-13-1742-2_68.

Vogl, Richard et al. (2017). "Drum Transcription via Joint Beat and Drum Modeling Using Convolutional Recurrent Neural Networks". In: *Proceedings of ISMIR (International Conference on Music Information Retrieval)*. Suzhou, China. URL: https://archives.ismir.net/ismir2017/paper/000123.pdf.

Vukotic, Vedran, Christian Raymond, and Guillaume Gravier (2018). "A Crossmodal Approach to Multimodal Fusion in Video Hyperlinking". In: *IEEE MultiMedia* 25.2, pp. 11–23. DOI: 10.1109/MMUL.2018.023121161.

Wang, Jiang et al. (2014). "Learning Fine-grained Image Similarity with Deep Ranking". In: *Proceedings of IEEE CVPR (Conference on Computer Vision and Pattern Recognition)*. Columbus, OH, USA. DOI: 10.1109/cvpr.2014.180.

Wang, Jianren, Zhaoyuan Fang, and Hang Zhao (2020). "AlignNet: A Unifying Approach to Audio-Visual Alignment". In: *Proceedings of WACV (Winter Conference on Applications of Computer Vision)*. Snowmass Village, CO, USA, pp. 3298–3306. DOI: 10.1109/WACV45572.2020.9093345.

Wang, Jinjun, Engsiong Chng, and Changsheng Xu (2006). "Fully and Semi-Automatic Music Sports Video Composition". In: *Proceedings of ICME (International Conference on Multimedia and Expo)*. Toronto, ON, Canada. DOI: 10.1109/ICME.2006.262926.

Wang, Jinjun et al. (2007). "Generation of Personalized Music Sports Video Using Multimodal Cues". In: *IEEE Transactions on Multimedia* 9.3. DOI: 10.1109/TMM.2006.888013.

Wang, Ju-Chiang et al. (2012). "The acousticvisual emotion guassians model for automatic generation of music video". In: *Proceedings of ACM Multimedia*. Nara, Japan. DOI: 10.1145/2393347.2396494.

Wang, Kaiye et al. (2016). "A Comprehensive Survey on Cross-modal Retrieval". In: *arXiv preprint arXiv:1607.06215*. URL: http://arxiv.org/abs/1607.06215.

Weinberger, Kilian Q. and Lawrence K. Saul (2009). "Distance metric learning for large margin nearest neighbor classification". In: *Journal of Machine Learning Research* 10.Feb, pp. 207–244. URL: https://www.jmlr.org/papers/volume10/weinberger09a/weinberger09a.pdf.

Weiss, Christof and Meinard Muller (2015). "Tonal complexity features for style classification of classical music". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. South Brisbane, Queensland, Australia, pp. 688–692. DOI: 10.1109/ICASSP.2015.7178057.

Wolff, Daniel and Tillman Weyde (2014). "Learning music similarity from relative user ratings". In: *Information Retrieval* 17.2, pp. 109–136. DOI: 10.1007/s10791-013-9229-0.

Won, Minz et al. (2021). "Emotion Embedding Spaces for Matching Music to Stories". In: *Proceedings of ISMIR (International Society for Music Information Retrieval Conference)*. Online. URL: https://archives.ismir.net/ismir2021/paper/000097.pdf.

Yamauchi, Masaki et al. (May 2006). "Chapter Generation for Digital Video Recorder Based on Perceptual Clustering". In: *IEEE Transactions on Consumer Electronics* 52.2, pp. 460–466. DOI: 10.1109/TCE.2006.1649665.

Yesiler, Furkan, Joan Serrà, and Emilia Gómez (2020). "Accurate and Scalable Version Identification Using Musically-Motivated Embeddings". In: *Proceedings of ICASSP (International Conference on Acoustics, Speech and Signal Processing)*. Barcelona, Spain. DOI: 10.1109/ICASSP40776.2020.9053793.

You, Junyong, Guizhong Liu, and Andrew Perkis (2010). "A semantic framework for video genre classification and event analysis". In: *Signal Processing: Image Communication* 25.4. DOI: 10.1016/j.image.2010.02.001.

Yu, Yi et al. (Feb. 2019). "Deep Cross-Modal Correlation Learning for Audio and Lyrics in Music Retrieval". In: *ACM Transactions on Multimedia Computing, Communications, and Applications* 15.1, pp. 1–16. DOI: 10.1145/3281746.

Yue, Joe et al. (2015). "Beyond Short Snippets: Deep Networks for Video Classification". In: *Proceedings of IEEE CVPR (Conference on Computer Vision and Pattern Recognition)*. Boston, MA, USA. DOI: 10.1109/cvpr.2015.7299101.

Zabih, Ramin, Justin Miller, and Kevin Mai (1995). "Feature-Based Algorithms for Detecting and Classifying Scene Breaks". In: URL: https://ecommons.cornell.edu/handle/1813/7187.

Zeng, Donghuo, Yi Yu, and Keizo Oyama (2018). "Audio-Visual Embedding for Cross-Modal Music Video Retrieval through Supervised Deep CCA". In: *Proceedings of IEEE ISM (International Symposium on Multimedia)*. Taichung, Taiwan. DOI: 10.1109/ism.2018.00-21.

Zhang, HongJiang, Atreyi Kankanhalli, and Stephen W Smoliar (1993). "Automatic partitioning of full-motion video". In: *Multimedia systems* 1.1, pp. 10–28. DOI: doi.org/10.1007/BF01210504.

Zhang, Xiaojing et al. (May 2017). "Cooling Energy Consumption Investigation of Data Center IT Room with Vertical Placed Server". In: *Energy Procedia* 105, pp. 2047–2052. DOI: 10.1016/j.egypro.2017.03.581.

Zhang, Yingying et al. (Dec. 2019). "Learning Incremental Triplet Margin for Person Re-identification". In: *Proceedings of AAAI (Innovative Applications of Artificial Intelligence Conference)*. Honolulu, Hawaii, USA. DOI: 10.1609/aaai.v33i01.33019243.

Zhao, Hang et al. (2018). "The Sound of Pixels". In: *Proceedings of ECCV (European Conference on Computer Vision)*. Munich, Germany. URL: http://sound-of-pixels.csail.mit.edu.

ECOLE DOCTORALE

Titre : Apprentissage métrique pour la recommandation vidéo-musique

Mots clés : analyse audio, analyse d'image, apprentissage métrique, recommandation

Résumé : À l'écran, la musique permet de communiquer, à travers des codes culturels établis, des émotions ou des éléments narratifs clés. Cette communication repose non seulement sur un choix judicieux de morceaux pour la bande son, mais aussi sur leur synchronisation avec les événements saillants de la vidéo. La tâche qui consiste à conseiller et réaliser cette association est au centre de l'industrie de la supervision musicale, et s'effectue traditionnellement à la main.

Dans cette thèse, l'on étudie l'automatisation des tâches liées à la supervision musicale. La quête de la musique idéale n'a généralement pas de solution unique, puisqu'en plus de l'analyse des contenus audiovisuels, il faut tenir compte de contraintes légales et budgétaires. Nous procédons donc par recommandation. Alors qu'une quantité toujours croissante de musique est produite chaque jour, il est raisonnable d'envisager une approche par apprentissage. Plus précisément, nous utilisons l'apprentissage métrique pour produire des représentations communes aux données musicales et visuelles.

Dans un premier temps, nous abordons un problème de similarité musicale, dont le but est d'élargir le champ de recherche dans les catalogues musicaux. Nous implémentons une imitation efficace d'une métrique de similarité par critères, applicable directement aux fichiers audio non annotés. Cette méthode repose sur des réseaux de neurones convolutionnels entraînés à reproduire des listes de recommandation. Puis, nous nous concentrons sur la recommandation directe de musique à partir d'une vidéo. Nous adaptons un système autodidacte simple et montrons comment en améliorer les performances, grâce à de meilleures représentations audio en entrée et un apprentissage de leur agrégation temporelle. Nous menons ensuite une étude quantitative et qualitative sur les clips musicaux, afin de mieux comprendre comment y sont articulés les événements audio et vidéo. Nos résultats démontrent avec quel soin la musique et l'image peuvent être synchronisés, mais aussi que le niveau de co-occurrence des différents événements dépend de plusieurs critères, tels que le genre musical ou vidéo. À partir de ce constat, nous présentons le premier système de recommandation dédié à la supervision musicale : le Seg-VM-Net, qui tient compte à la fois du contenu et de la structure pour apparier musiques et vidéos.

Title : Metric Learning for Video to Music Recommendation

Keywords : audio analysis, image analysis, metric learning, recommendation

Abstract : Music enhances moving images and allows to efficiently communicate emotion or narrative tension, thanks to cultural codes common to the filmmakers and viewers. A successful communication requires not only a choice of tracks matching the video's mood and content, but also a temporal synchronization of the audio and visual main events. This is the goal of the music supervision industry, which traditionally carries out the task manually.

In this dissertation, we study the automation of tasks related to music supervision. The music supervision problem generally doesn't have a unique solution, as it includes external constraints such as the client's identity or budget. It is thus relevant to proceed by recommendation. As the number of available musical videos is in constant augmentation, it makes sense to use data-driven tools. More precisely, we use the metric learning paradigm to learn the relevant projections of multimodal (video and music) data.

First, we address the music similarity problem, which is used to broaden the results of a music search. We implement an efficient content-based imitation of a tag-based similarity metric. To do so, we present a method to train a convolutional neural network from ranked lists. Then, we focus on direct, content-based music recommendation for video. We adapt a simple self-supervised system and we demonstrate a way to improve its performance, by using pre-trained audio features and learning their aggregation. We then carry a qualitative and quantitative analysis of official music videos to better understand the temporal organization of musical videos. Results show that official music videos are carefully edited in order to align audio and video events, and that the level of synchronization depends on the music and video genres. With this insight, we propose the first recommendation system designed specifically for music supervision: the Seg-VM-Net, which uses both content and structure to perform the matching of music and video.

Institut Polytechnique de Paris
91120 Palaiseau, France