Universitat Pompeu Fabra Barcelona

# Deep Audio Representation Learning for Music Using Weak Supervision

**Pablo Alonso Jiménez**

TESI DOCTORAL UPF / 2024

Thesis supervisors:

Dr. Dmitry Bogdanov
Dept. of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona, Spain

Dr. Xavier Serra Casals
Dept. of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona, Spain

Dissertation submitted to the Department of Information and Communication Technologies of Universitat Pompeu Fabra in partial fulfillment of the requirements for the degree of

DOCTOR PER LA UNIVERSITAT POMPEU FABRA




Music Technology Group (http://mtg.upf.edu), Dept. of Information and Communication Technologies (http://www.upf.edu/dtic), Universitat Pompeu Fabra (http://www.upf.edu), Barcelona, Spain.

The doctoral defense was held on ..... at the Universitat Pompeu Fabra and scored as .....

---

**Dmitry Bogdanov**
(Thesis Supervisor)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

---

**Xavier Serra**
(Thesis Supervisor)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

---

**Dr. Romain Hennequin**
(Thesis Committee Member)
Deezer, Paris, France

---

**Dr. Perfecto Herrera Boyer**
(Thesis Committee Member)
Universitat Pompeu Fabra (UPF), Barcelona, Spain

---

**Dr. Rachel Bittner**
(Thesis Committee Member)
Spotify, Paris, France

---

This thesis has been carried out at the Music Technology Group (MTG) of Universitat Pompeu Fabra in Barcelona, from September 2020 to July 2024. It has been supervised by Dr. Dmitry Bogdanov and Dr. Xavier Serra, and included a 6-month research visit at Utopia Music from July 2022 to December 2022 (remote), supervised by Dr. Thomas Lidy.

The work in Chapters 3, 4, and 6 was conducted in collaboration with Dmitry Bogdanov and Xavier Serra. Sections 3.3.1 are based on work done in collaboration with Dmitry Bogdanov, Jordi Pons, and Xavier Serra. The contributions from Chapter 5 were conducted in collaboration with Xavier Favory, Hadrien Foroughmand, Grigoris Bourdalas, Dmitry Bogdanov, Xavier Serra, and Thomas Lidy during a research visit at Utopia Music. The work in Chapter 7 was a collaboration with Leonardo Pepino, Roser Batlle-Roca, Pablo Zinemanas, Dmitry Bogdanov, Xavier Serra, and Martín Rocamora.

Essentia-TensorFlow (Section 8.1), and Essentia models (Section 8.1.2) were developed in collaboration with Dmitry Bogdanov. The development of Essentia.js was led by Albin Correya, with the participation of Luis Joglar-Ongay, Jorge Marcos-Fernández, Dmtry Bogdanov, and the author. The demos presented in Section 8.1.4 were developed by Jorge Marcos, Albin Correya, Xavier Lizarraga-Seijas, Dmitry Bogdanov, Alastair Porter, and the author.

*A Pepi, Tito, Julio y Luisi.*

# Acknowledgements

I arrived in Barcelona to start my Sound and Music Computing master's degree at the MTG in 2016 motivated by having completed Xavier Serra's Audio Signal Processing for Music Applications online course. Eight years later, after a journey full of learnings, experiences around the world, and meeting amazing people, I can only be grateful for that moment, nine years back, when I had this random impulse for googling `Coursera audio processing free`.

I will start by thanking my supervisors, Dmitry and Xavier, for their guidance and support throughout these years. Xavier, your teachings have gone beyond the academic. Dmitry, working with you has been an honor. Without you, I would not be the programmer, researcher, and tea drinker I am today.

I would also like to thank all the people who were fundamental in bringing me to the point of starting this thesis. Álvaro López, and David Rodri, for encouraging me to leave my hometown to specialize in audio processing. Professor Manuel Sobreira, for believing in me and helping me to secure my first research internship. David Pérez and Daniel Fernández, for being my first mentors in the industry. Professor Xulio Hermida, for being my first mentor in a research lab. To Merlijn Blaauw, Jordi Janer, Jordi Bonada, Oscar Mayor, and Alvaro Sarasúa, for their mentorship during my internship at VoctroLabs. And to Yuji Hisaminato and Ryonosuke Daido, for their mentorship during my research stay at Yamaha, Japan.

I want to express my gratitude to all my collaborators throughout this thesis, without whom it would not have been possible: Jordi Pons, Albin Correya, Luis Joglar, Jorge Marcos, Morgan Buisson, Xavier Favory, Hadrien Foroughmand, Grigoris Bourdalas, Xavier Lizarraga, Christos Plachouras, Leonardo Pepino, Roser Batlle, Pablo Zinemanas, and Martín Rocamora. Also, I am very grateful to all the people who have been around at the MTG during this time, both from the academic and personal sides. Following a room/year correlated-ish order: Minz Won, Seva Eremenko, Helena Cuesta, Georgi Dzhambazov, Frederic Font, Alastair Porter, Andrés Ferraro, Edu Fonseca, Rong Gong, Pritish Chandna, Olga Slizovskaia, Sergio Oramas, Marius Miron, António Ramires, Furkan Yesiler, Philip Tovstogan, Pablo Zinemanas, Juan Gómez, Lorenzo Porcaro, Alia Morsi, Tom Nuttal, Benno Weck, Miguel Pérez, Jyoti Narang, Behzad Haki, Hyon Kim, Enric Gusó, Guillem Cortès, Oğuz Araz, Penny Anastasopoulou, Pedro Ramoneda, Genís Plaja, and Lonce Wyse. Also thanks to Cristina Garrido and Sonia Espí for their support.

I would also like to acknowledge my friends from Asturies: Álvaro, Dani, Ceci,

Samu, Nuu, Luci, Xavi, Mikel, Menen, and Ger; from my years in Galicia: David, Didak, Fonsi, Julia, Togo, Belén, Anxo, Vero, and, Javi; and to the Flatfound extended family: Dani, Minz, Tessy, Lorenzo, Róisín, Joe, and Néstor for their support and love.

Next, I would like to thank all my family from Piedrasblancas, Málaga, Valladolid, Burgos, and Samartín. Specially, to my father for teaching me to question everything, to my mother for teaching me that *todo lo que ocurre conviene*, and to my brother, for being my best friend and my main valve of escape with "Llevólu'l Sumiciu".

Finally, to you, Laura. Gracies por ser casa. Quiérote.

# Abstract

Music audio tagging is the Music Information Retrieval task of assigning one or multiple labels to an audio signal. Music tagging systems are essential for developing applications involving cataloging, retrieval, or recommendation, so enhancing the accuracy, robustness, and efficiency of these models is beneficial for many real-world music applications. Current state-of-the-art music tagging systems rely on deep learning approaches, which offer high performance but also introduce challenges due to their large data requirements and tendency to overfit. In this thesis, we propose addressing music tagging from the perspective of *representation learning* to alleviate these limitations.

The goal of representation learning is to design pre-training objectives that make the learned representations suitable for several downstream tasks. When the representations are well-suited to the downstream task, it is often possible to achieve good performance using shallow models that require few resources to train and run. Additionally, using a single representation model to feed several shallow models is more efficient than having individual end-to-end models for each task, and enables addressing new related tasks with little additional effort.

Our work starts by investigating the capabilities of the representations learned by competitive music and audio tagging systems and evaluating their capabilities on out-of-distribution data, finding that pre-trained representations provide generalization benefits. To support the rest of this thesis, we create a large-scale dataset matched to Discogs'[1] open music metadata that we use to develop novel representation models. Then, we investigate the effectiveness of using editorial and consumption metadata (such as artist names and playlists) as a source of supervision, showing that this information favors downstream performance without the need for explicit annotations which are typically much harder to obtain. After this, we look into the transformer architecture, proposing design choices that optimize its performance for music representation learning. In our last contribution, we propose adapting existing audio interpretability strategies to operate with pre-trained representations, thus contributing to more insightful music classification models.

Finally, this work is carried out in the context of Essentia,[2] an open-source library and collection of models for audio and music analysis. The techniques and models developed in this thesis are openly available as part of Essentia and have already been used both by the research community and industry.

[1] http://discogs.com
[2] http://essentia.upf.edu

# Resum

La classificació d'àudio musical és una tasca del camp de la Recuperació de la Informació Musical que consisteix a assignar una o múltiples etiquetes a un senyal d'àudio. Els sistemes de classificació musical són essencials per al desenvolupament d'aplicacions que impliquen catalogació, recuperació o recomanació, de manera que millorar la precisió, la robustesa i l'eficiència d'aquests models és beneficiós per a moltes aplicacions musicals del món real. Els sistemes de classificació musical d'última generació es basen en enfocaments d'aprenentatge profund, que ofereixen un alt rendiment però també introdueixen desafiaments a causa dels seus grans requisits de dades i la tendència a sobreajustar-se. En aquesta tesi, proposem abordar la classificació musical des de la perspectiva de l'*aprenentatge de representacions* per minimitzar aquestes limitacions.

La meta de l'aprenentatge de representacions és dissenyar objectius de preentrenament que facin que les representacions apreses siguin adequades per a diverses tasques posteriors. Quan les representacions són adequades per a la tasca posterior, sovint és possible aconseguir un bon rendiment utilitzant models lleugers que requereixen pocs recursos per entrenar i inferir. A més, utilitzar un únic model de representació per alimentar diversos classificadors lleugers és més eficient que tenir models d'aprenentatge profund específics per a cada tasca, i permet abordar noves tasques relacionades amb poc esforç addicional.

El nostre treball comença investigant les capacitats de les representacions apreses per sistemes competitius de classificació musical i d'àudio i avaluant les seves capacitats en dades fora de distribució, trobant que les representacions preentrenades proporcionen beneficis per a la generalització. Per a donar suport a la resta d'aquesta tesi, creem un conjunt de dades a gran escala enllaçat a les metadades musicals obertes de Discogs[3] que utilitzem per desenvolupar nous models de representació. Després, investiguem l'eficàcia d'utilitzar metadades editorials i de consum (com noms d'artistes i llistes de reproducció) com a font de supervisió, demostrant que aquesta informació afavoreix el rendiment en tasques posteriors sense necessitat d'anotacions explícites, que normalment són molt més difícils d'obtenir. A continuació, examinem l'arquitectura del transformer, proposant dissenys que optimitzen el seu rendiment per a l'aprenentatge de representacions musicals. En la nostra última contribució, proposem adaptar estratègies d'interpretabilitat d'àudio existents per

[3] http://discogs.com

operar sobre representacions preentrenades, contribuint així a crear models de classificació musical més comprensibles.

Finalment, aquest treball es duu a terme en el context d'Essentia,[4] una biblioteca de codi obert i col · lecció de models per a l'anàlisi d'àudio i música. Les tècniques i models desenvolupats en aquesta tesi estan disponibles de forma oberta com a part d'Essentia i ja han estat utilitzats tant per la comunitat investigadora com per la indústria.

[4] http://essentia.upf.edu

# Resumen

La clasificación de audio musical es la tarea de Recuperación del campo de la Información Musical que consiste en asignar una o múltiples etiquetas a una señal de audio. Los sistemas de clasificación musical son esenciales para el desarrollo de aplicaciones que implican catalogación, recuperación o recomendación, por lo que mejorar la precisión, la robustez y la eficiencia de estos modelos es beneficioso para muchas aplicaciones musicales del mundo real. Los sistemas de clasificación musical de última generación se basan en enfoques de aprendizaje profundo, que ofrecen un alto rendimiento pero también introducen desafíos debido a sus grandes requisitos de datos y la tendencia a sobreajustarse. En esta tesis, proponemos abordar la clasificación musical desde la perspectiva del *aprendizaje de representaciones* para aliviar estas limitaciones.

La meta del aprendizaje de representaciones es diseñar objetivos de preentrenamiento que hagan que las representaciones aprendidas sean adecuadas para varias tareas posteriores. Cuando las representaciones son adecuadas para la tarea posterior, a menudo es posible lograr un buen rendimiento utilizando modelos ligeros que requieren pocos recursos para entrenar e inferir. Además, utilizar un único modelo de representación para alimentar varios clasificadores ligeros es más eficiente que tener modelos de aprendizaje profundo específicos para cada tarea, y permite abordar nuevas tareas relacionadas con poco esfuerzo adicional.

Nuestro trabajo comienza investigando las capacidades de las representaciones aprendidas por sistemas competitivos de clasificación musical y de audio y evaluando sus capacidades en datos fuera de distribución, encontrando que las representaciones preentrenadas proporcionan beneficios para la generalización. Pdeara apoyar el resto de esta tesis, creamos un conjunto de datos a gran escala enlazados a los metadatos musicales abiertos de Discogs'[5] que utilizamos para desarrollar nuevos modelos de representación. Luego, investigamos la eficacia de usar metadatos editoriales y de consumo (como nombres de artistas y listas de reproducción) como fuente de supervisión, demostrando que esta información favorece el rendimiento en tareas posteriores sin necesidad de anotaciones explícitas, que normalmente son mucho más difíciles de obtener. Después de esto, examinamos la arquitectura del transformer, proponiendo diseños que optimizan su rendimiento para el aprendizaje de representaciones musicales. En nuestra última contribución, proponemos adaptar estrategias de interpretabi-

[5] http://discogs.com

lidad de audio existentes para operar sobre representaciones preentrenadas, contribuyendo así a crear modelos de clasificación musical más comprensibles.

Finalmente, este trabajo se lleva a cabo en el contexto de Essentia,[6] una biblioteca de código abierto y colección de modelos para el análisis de audio y música. Las técnicas y modelos desarrollados en esta tesis están disponibles de forma abierta como parte de Essentia y ya han sido utilizados tanto por la comunidad investigadora como por la industria.

[6] http://essentia.upf.edu

# Contents

**Abstract** xi

**Contents** xvii

**List of figures** xxi

**List of tables** xxiii

Acronyms . . . xxvii

**1 Introduction** **1**
1.1 Motivation . . . 1
1.2 Scope of the thesis . . . 3
1.2.1 Representation learning . . . 4
1.2.2 MIR research based on weak supervision . . . 5
1.2.3 Essentia . . . 7
1.3 Challenges in music representation learning . . . 8
1.4 Research questions . . . 9
1.5 Dissertation outline . . . 11

**2 Scientific background** **15**
2.1 Introduction . . . 15
2.2 First neural networks for music classification . . . 15
2.3 Deep learning and music auto-tagging . . . 16
2.4 Representation learning for music classification . . . 18
2.5 Self-supervised approaches . . . 19
2.5.1 Contrastive learning . . . 21
2.5.2 Masked autoencoders . . . 22
2.6 Music representation learning with metadata . . . 23
2.6.1 Editorial metadata . . . 23
2.6.2 Consumption metadata . . . 24
2.7 General purpose audio representations . . . 24
2.8 Discogs in MIR research . . . 25
2.9 Transformers for music classification . . . 25
2.9.1 Transformers in audio classification tasks . . . 25
2.9.2 Music representation learning with transformers . . . 26
2.10 Interpretability . . . 27

**3 Representation learning based on labels** **29**

3.1 Introduction . . . 29
3.2 Method . . . 30
3.2.1 Transfer learning protocol . . . 30
3.2.2 5-fold cross-validation . . . 31
3.2.3 Cross-collection evaluation . . . 31
3.2.4 Fixed-splits evaluation . . . 32
3.2.5 Downstream tasks . . . 33
3.3 Experiments . . . 33
3.3.1 Preliminary evaluation on music genre recognition . . . 33
3.3.2 Evaluation of pre-trained models on music classification 37
3.3.3 DiscogsEffNet a model based on Discogs' music styles . 40
3.3.4 Evaluation on small downstream tasks . . . 41
3.3.5 Downstream dataset size analysis . . . 45
3.3.6 Combination of embeddings . . . 45
3.3.7 Cross-collection evaluation . . . 49
3.3.8 Additional benchmarks on public datasets . . . 52
3.3.9 Effect of the pre-training dataset size . . . 53
3.4 Conclusions . . . 54

**4 Representation learning based on editorial metadata 57**
4.1 Introduction . . . 57
4.2 Method . . . 59
4.3 Experiments . . . 60
4.3.1 Contrastive targets based on Discogs' metadata . . . 61
4.3.2 Pre-training models with editorial metadata . . . 62
4.3.3 Downstream datasets . . . 63
4.3.4 Transfer learning evaluation setup . . . 64
4.3.5 Stacks of embeddings . . . 64
4.3.6 Multitask model . . . 64
4.4 Results and discussion . . . 65
4.5 Conclusions . . . 68

**5 Representation learning based on consumption metadata 71**
5.1 Introduction . . . 71
5.2 Method . . . 72
5.3 Experiments . . . 73
5.3.1 Contrastive learning setup . . . 74
5.3.2 Architectures . . . 74
5.3.3 Pre-training models with consumption metadata . . . 75
5.3.4 Downstream datasets . . . 76
5.3.5 Music tagging . . . 76
5.3.6 Music similarity . . . 77
5.4 Results and discussion . . . 77

5.5 Improving the selection of positive pairs . . . 80
5.5.1 Modifications to the model . . . 81
5.5.2 Results and conclusions . . . 82
5.6 Conclusions . . . 83

**6 Representation learning with transformers and patchout 85**
6.1 Introduction . . . 85
6.2 Method . . . 86
6.3 Experiments and results . . . 87
6.3.1 Dataset and pre-processing . . . 87
6.3.2 Pre-training . . . 88
6.3.3 Evaluation . . . 89
6.3.4 Extracting embeddings from the transformer . . . 90
6.3.5 Impact of the initial weights . . . 90
6.3.6 Effect of the input segment length . . . 92
6.3.7 Performance in downstream tasks . . . 92
6.3.8 Faster feature extraction with inference patchout . . . 95
6.4 Conclusions . . . 96

**7 Sonified prototypical learning for interpretable music classification 97**
7.1 Introduction . . . 97
7.2 Method . . . 98
7.3 Experiments and results . . . 100
7.3.1 Datasets . . . 100
7.3.2 Implementation details . . . 101
7.3.3 Classification Results . . . 102
7.3.4 Effect of the decoder . . . 102
7.3.5 Sonifying the prototypes . . . 103
7.4 Conclusions . . . 104

**8 Applications and outreach 105**
8.1 Essentia . . . 105
8.1.1 Essentia-TensorFlow . . . 106
8.1.2 essentia models . . . 108
8.1.3 Essentia.js . . . 109
8.1.4 Essentia demos . . . 112
8.2 Examples of applications by third parties . . . 114
8.2.1 cosine.club . . . 114
8.2.2 SubmitHub . . . 115

**9 Conclusions and future perspectives 117**
9.1 Introduction . . . 117

9.2 Summary of contributions . . . . . 119
9.3 Open issues and future perspectives . . . . . 121
9.3.1 Self-supervised learning perspective . . . . . 121
9.3.2 Multitask learning . . . . . 121
9.3.3 Multi-modal learning . . . . . 122
9.3.4 Representation learning and time resolution . . . . . 122
9.3.5 Music understanding evaluation . . . . . 123
9.3.6 Interpretability in representation learning . . . . . 123

**Bibliography** **127**

**Appendix A: contributions by the author** **145**

# List of figures


1.1 Diagram of the representation learning pipeline. In the pre-training stage, a model is trained on a primary task using a large dataset. In the downstream stage, the model is fine-tuned or used as a feature extractor to train new models on downstream tasks. . . . . 5

1.2 Diagram of the stages in the proposed representation learning setup. Apart from the typical pre-training and downstream stages, we propose additional steps to evaluate the generalization capabilities of the models, their interpretability, and their deployment in real-world applications. The dashed lines mark the blocks for which we do not propose contributions. . . . . 10

3.1 Evolution of the normalized accuracies with respect to the number of samples per class expressed as the average of three trials. . . . . 46

3.2 Corrected rankings ($\bar{\#}$) obtained by the embeddings and their combinations. A score of 1 means that the model is not statistically different to the model with the highest accuracy according to a t-test with a p-value of 0.05. . . . . 48

3.3 Distribution of normalized accuracies obtained by classifiers trained on embeddings extracted from the EffNet model trained with different amounts of data. . . . . 53

4.1 Overview of the proposed pipelines for a single (left) and multiple tasks (right). $E$ is the encoder, $P$ the projection head, $B$ are the weights for the bilinear similarity, and $\mathcal{L}$ is the loss term. In the multitask setup, tasks go from *0* to *n*. . . . . 59

4.2 Training accuracies over the epochs for the different associations. . 63

5.1 Illustration of our pre-training pipeline. The features $x$ and $y$ from the associated pairs are input to the model $B(\cdot)$ and projector $H(\cdot)$. $B(\cdot)$ and $H(\cdot)$ are optimized using a contrastive loss. $B(\cdot)=B_x(\cdot)=B_y(\cdot)$ and $H(\cdot)=H_x(\cdot)=H_y(\cdot)$ in all the cases except for *Word2Vec representation*. . . . . 74

6.1 Illustration of our system at the training and downstream evaluation stages where $x$ is the input spectrogram, $k^0$ is the sequence of tokens after the patchout, $y$ is the target labels, and BCE is the binary cross-entropy loss. Trainable and frozen blocks are colored green and blue respectively. . . . . 86

6.2 mAP scores were obtained with our evaluation setup in the MTT dataset using embeddings extracted from different blocks and tokens from the transformer. We evaluate the *cls* (c), *dist* (d), and *avg* (a) tokens and stacks of their combinations extracted from the transformer blocks 5 to 12. . . . . 91
6.3 mAP scores against throughput for MAEST-30s under different amounts of frequency (F) and time (T) patchout. The radius is proportional to the parameter count and the inference is performed on the CPU. . . . . 95

7.1 Diagram of the proposed PECMAE model. The colored boxes indicate trainable modules. . . . . 99

8.1 this code snippet uses essentia-tensorflow algorithms to run a maest model from chapter 6. . . . . 108
8.2 essentia models site screenshot. . . . . 109
8.3 Inference example with musicnn-based models using *essentia.js-model* via es6 style imports. . . . . 111
8.4 cosine.club website screenshot. Given a query track, the interface returns the closest tracks in the database based on the cosine distance. . . . . 114

# List of tables


2.1 Comparison transformers from the literature in terms of initialization weights, number of GPUs used for training, training time, and mAP obtained in AudioSet. . . . . 26

3.1 Downstream tasks (ft.: full tracks, exc.: excerpts). . . . . 32
3.2 5-fold cross-validation and cross-collection evaluation results for the genre tasks. The 5-fold cross-validation results are the mean and standard deviation of the balanced accuracy across the folds. The best results are marked in bold. . . . . 36
3.3 Model embeddings. RF: Receptive field, Approach: fully-supervised (FS) or self-supervised (SS). . . . . 38
3.4 Class-weighted accuracies in the downstream tasks for the embeddings as an average of 5-fold cross-validation. The Top results are marked in bold. . . . . 39
3.5 Embedding extractors. RF: Receptive field in seconds, Approach: fully supervised (FS), self-supervised (SS), or conditioned supervision (CS). . . . . 42
3.6 Classification metrics per embedding extractor for all the downstream tasks. BL: baseline, #: rank, $\bar{\#}$: corrected rank, acc.: normalized accuracy. . . . . 44
3.7 Normalized accuracies (the best model in each task marked in bold) in the external validation. BL: Baseline, Num.: number of tracks used as external ground truth, Agr.: agreement rate. . . . . 51
3.8 Comparison of the SOTA models (up) with the embedding extractors (middle) and the best combinations for each task (down) in GTZAN and the moods and themes subset of the MTG-Jamendo dataset. . . . . 52

4.1 Statistics for the metadata association and their respective models. We show the number of pairs used, association diversity (number of different tracks, releases, artists, and record labels, respectively), training time (hours), and validation accuracy (%). . . . . 61
4.2 Considered downstream datasets. . . . . 64

4.3 ROC-AUC, PR-AUC, and accuracy metrics for the downstream datasets. The five horizontal groups represent SOTA models from the literature trained from scratch, SOTA feature extractors from the literature, baseline feature extractors trained by ourselves, the proposed feature extractors based on metadata associations, and the proposed feature extractors combining associations. (*) results were computed in a clean version of *MTAT* and are not directly comparable. . . . . 67

4.4 Top-5 combinations of the **Tr**ack, **Re**lease, **Ar**tist, **La**bel and **St**yle Tags features in *MTAT*. The results are sorted according to the ROC-AUC values. . . . . 68

5.1 Number of pairs per epoch and pair generation algorithms. *indicates that these are different pairs on each epoch. . . . . 76

5.2 Datasets used in the downstream evaluation. . . . . 77

5.3 Metrics in the music classification datasets expressed in macro ROC-AUC and Average Precision. For each architecture, we present the baselines on top and the proposed models below. Metrics statistically equivalent or higher than *Artist CO* according to a one-sided t-test ($p\text{-}value = 0.005$) are marked in light grey. The highest metric per dataset is marked in bold. . . . . 79

5.4 Music similarity accuracy and the average difference between anchor/negative and anchor/positive. . . . . 80

5.5 Macro ROC-AUC (ROC) and mean Average Precision (mAP) metrics in the music tagging datasets. The highest metric per dataset marked in bold. . . . . 82

5.6 Music similarity accuracy and average difference for *ICO* compared to our previous best model. . . . . 83

6.1 Automatic tagging datasets used in the downstream evaluation. The datasets are compared in terms of sample size, number of labels, audio duration (Full Tracks or excerpts of fixed duration), average labels per track, and the splits used in our evaluations. . 89

6.2 mAP scores obtained in the training and downstream tasks using different initializations. We considered Random Weights, and pretrained weights from DeiT and PaSST. . . . . 91

6.3 mAP scores obtained in the training and downstream tasks using different spectrogram segment lengths. $T$ represents the spectrogram segment length. . . . . 92

6.4 ROC-AUC and mAP scores obtained in the downstream tasks. Our baseline consists of an EffNet-B0 architecture trained in Discogs20. Additionally, we report the SOTA results distinguishing models with all parameters trained in the downstream tasks (fully trained) and models evaluated with shallow classifiers. For every task, we mark in bold the best score obtained by a MAEST model and highlight in grey models achieving better performance than the best open alternative. † Models not publicly available. . . . . 94

7.1 Considered datasets in terms of number of classes, samples, and total duration in hours. . . . . 101
7.2 Normalized classification accuracies. . . . . 103

8.1 Comparison of user preferences for the different music stlye labelling systems tested by SubmitHub. The sable shows the percentage of users that selected all, some, or no tags when using the different systems. . . . . 115

# Acronyms

**APNet** Audio Prototype Network

**CNN** Convolutional Neural Network

**CQT** Constant-Q transform

**CV** Computer Vision

**DNN** Deep Neural Network

**GPU** Graphics Processing Unit

**KNN** K-Nearest Neighbors

**LDA** Linear Discriminant Analysis

**MAE** Masked Autoencoder

**MER** Music Emotion Recognition

**MFCC** Mel–Frequency Cepstral Coefficient

**MGR** Music Genre Recognition

**MIR** Music Information Retrieval

**MLP** Multi-Layer Perceptron

**MPD** Million Playlist Dataset

**MSD** Million Song Dataset

**MTAT** MagnaTagATune

**MTG** Music Technology Group - Universitat Pompeu Fabra

**NLP** Natural Language Processing

**PLSA** Probabilistic Latent Semantic Analysis

**SOTA** State of the Art

**SVM** Support Vector Machine

CHAPTER 1

# Introduction

## 1.1 Motivation

Music is a form of artistic expression through sound developed by humans for tens of thousands of years.[7] Through music, cultures have expressed the whole spectrum of emotions, transmitted their myths and wisdom, and prepared for rituals and ceremonies. In the modern days, music is present in many aspects of our lives. For instance, religious festivities such as Christmas are accompanied by carols, urban subcultures such as the clubbing scene are associated with uncountable niche styles of electronic music,[8] and sports events are accompanied by anthems and fan songs. Additionally, music can be presented in combination with other cultural expressions such as cinema (soundtracks), dance (ballet), or theater (musicals), which further enriches the complexity of the messages it can express. These examples illustrate that music is able to convey an exceptional amount of information, and so it has been studied from many different angles including theoretical, musicological, anthropological, or psychological perspectives.

The Music Information Retrieval (MIR) community is interested in developing computational approaches to automatically analyze music signals to extract parts of this information. Some authors use the term *Music Understanding* to refer to the set of MIR tasks aiming to extract different types of semantic information from music signals. For example, Music Genre Recognition (MGR) or Music Emotion Recognition (MER) are two of the most active research areas in the field. Being able to automatically recognize the genre of a song, or the emotions it can evoke, has a wide range of potential applications such as enhancing music recommendation systems, helping in music discovery or collection organization, or advancing research in other fields such as computa-

[7] https://en.wikipedia.org/wiki/History_of_music
[8] https://en.wikipedia.org/wiki/Clubbing_(subculture)

tional musicology (Volk et al., 2011), neuroscience (Levitin & Tirovolas, 2009), or medicine (Hillecke et al., 2005).

In the last decade, approaches based on Deep Neural Networks (DNNs) have dominated the field of MIR achieving remarkable results in many of its tasks. In this sense, tasks that are typically modeled as classification problems such as MGR or MER are especially well-suited for DNNs (Matityaho & Furst, 1995; Feng et al., 2003). Other examples of music understanding tasks addressed as classification problems include instrument recognition (Kaminsky & Materka, 1995), artist identification (Berenzweig et al., 2002), or key detection (Laden & Keefe, 1989). Additionally, tasks that were traditionally addressed with signal processing techniques such as onset detection (Marolt et al., 2002), beat tracking (Scaringella & Zoia, 2004), and tempo (Schreiber & Meinard, 2018) or pitch estimation (Kim et al., 2018b) have also been successfully modeled as classification problems using DNNs. However, one of the main limitations of these methods is that they typically require a large amount of annotated data to train effectively.

In MIR, annotating data can be particularly challenging because of different factors. First, the temporal dimension of music is crucial in order to create annotations for tasks such as beat tracking, onset detection, or pitch estimation. This means that if the annotations are not precise enough, the models will not be able to learn the task correctly. However, generating annotations with high level of precision is time-consuming and difficult to achieve at scale. Second, certain tasks require a high level of musical expertise to be annotated. For instance, it is not possible to generate key or chord annotations without a certain level of musical training. Finally, while tasks such as onset detection or tempo estimation are relatively objective, others such as MGR or MER can be very subjective due to multiple factors. For instance, the boundaries between different music genres are not always clear, and the same track can be classified into different genres depending on the listener's background and musical knowledge. Also, the mood or emotion evoked by a piece of music can be very different depending on the cultural background of the listener or the context in which it is played. Because of these factors, obtaining large and balanced datasets for music understanding tasks is particularly challenging.

*Representation learning* is a machine learning paradigm that alleviates the need for large amounts of high-quality annotated data by splitting the learning process in two stages (Bengio et al., 2013). In the first stage, a model is trained on a primary task using a large amount data that can be easily obtained. Typically, the primary task is a self-supervised (not requiring human annotations), or relies on a source of inexpensive labels that are abundant and easy to obtain. After training, the model is used to ease solving the actual task of interest by fine-tuning its parameters or using it as a feature extractor to train a new model. This process is sometimes called *transfer learning*, and

it has been shown to be very effective in many domains including MIR. Importantly, the primary and secondary tasks need to be related up to certain extent to work. For instance, a model trained on image classification will not produce useful features for a music classification task (although it may be a better initialization than starting from scratch (Alonso-Jiménez et al., 2023b)).

With the representation learning paradigm in mind, this thesis explores the use of inexpensive sources of supervision to train music representation models to improve music understanding tasks. In particular, we focus on different sources of publicly available music metadata. First, we rely on Discogs,[9] a public-domain database of editorial metadata. Discogs' *release* entries contain information about the artist, album, record label, country, year of release, and genre and style tags among other details. Additionally, we are interested in exploring the use of consumption metadata, which encodes human judgments of music that is suitable for the same context. For instance, playlists, DJ setlists, radio programs, or listening histories are examples of consumption metadata that can be used to train music representation models.

Finally, we could consider the release of *Essentia models* (Alonso-Jiménez et al., 2020a)[10] as the starting point of this thesis. Essentia models had the goal of creating a repository of audio-based DNNs to replace the former Essentia classifiers based on Support Vector Machines (SVMs). This thesis can be seen as an iterative effort to improve Essentia models by searching for better techniques to create music representation models that are then used to power the Essentia models' classifiers.

In the following sections, we address in more detail the scope of this thesis by providing deeper insights into the representation learning paradigm, the potential of weak sources of supervision for music, and the relevance of this research in the context of Essentia. Then, we identify some of the current problems related to music classification that representation learning can alleviate, and the list of research questions to be addressed in this thesis. Finally, we present the outline of this dissertation.

## 1.2 Scope of the thesis

In this section we develop in more detail the key concepts that define the scope of this thesis.

[9] https://www.discogs.com/
[10] http://essentia.upf.edu/models.html

### 1.2.1 Representation learning

Representation learning is a machine learning paradigm that aims to create systems to extract information useful for several downstream tasks (Bengio et al., 2013). The typical representation learning pipeline is depicted in Figure 1.1. Given a model trained on a primary task, the most common approaches are fine-tuning the model (or parts of it) in a secondary task or using it as a feature extractor to train a new model. In the former case, the primary task is sometimes called "source", "parent", or "pre-training" task, the learned representations are the "deep" or "latent" features, "deep embeddings", or just "embeddings", and the secondary task is called "target" or "downstream" task. We will use those terms indistinctly in this dissertation.

This paradigm allows leveraging large amounts of data without expensive curated annotations to pre-train representation models useful to train downstream models that generalize well with small amounts of annotated data and computations. The strategies to create representation models include self-supervised approaches, tasks with inexpensive labels, or a combination of many weak tasks in a multitask approach. Within this paradigm, it is possible to address the music classification problems from a different perspective. Instead of creating large and balanced music classification datasets to alleviate the reported ambiguity factors, one could rely on embedding extractors to improve classification in small, clean datasets.

The success of representation learning in domains such as Natural Language Processing (NLP) or Computer Vision (CV) has motivated the MIR community to explore this paradigm. For instance, most State of the Art (SOTA) approaches in NLP rely on self-supervised language modeling (Radford et al., 2018). This task consist of predicting the next word in a sentence given the previous ones. This approach has demonstrated to be a flexible way to learn useful representations that can be fine-tuned to solve many downstream tasks (Radford et al., 2019; Qiu et al., 2020). Self-supervised representation learning has also reached popularity in the CV and audio communities. However, the dense nature of image and audio signals makes the design of self-supervised tasks more challenging. Masked Autoencoders (MAEs) are popular approaches both in image (He et al., 2022) and audio (Chong et al., 2022). They could be considered the most similar approach to NLP's language modeling, and they work by predicting the masked regions of the input signal from the latent representation of the unmasked part. Other popular directions exploit metric learning to create representations that are invariant to distortions on the input signal. For instance, SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) are popular CV approaches that rely on contrastive learning to discover useful representations. Both approaches have been successfully adapted to the audio domain (Wang et al., 2022).

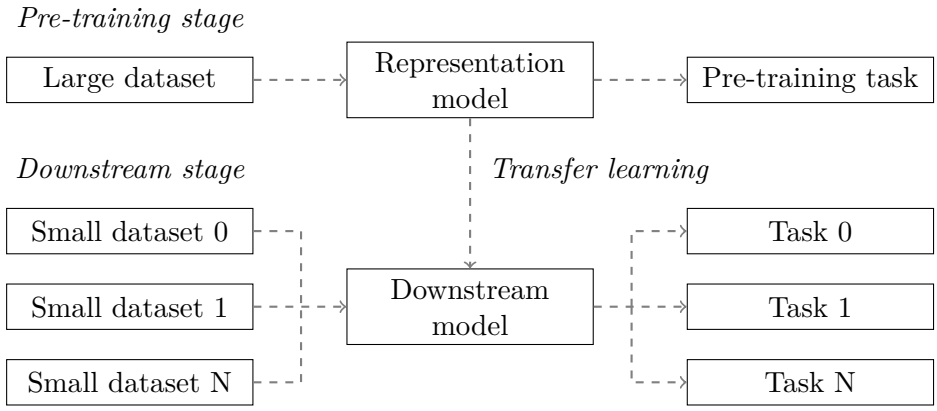


**Figure 1.1:** Diagram of the representation learning pipeline. In the pre-training stage, a model is trained on a primary task using a large dataset. In the downstream stage, the model is fine-tuned or used as a feature extractor to train new models on downstream tasks.

While the performance of self-supervised representation models have been improving drastically both in the CV and audio, it is still not clear that a single self-supervised pre-training method would be able to excel in all downstream tasks, and in particular tasks such as beat-tracking (Hung et al., 2022) or key estimation (Korzeniowski & Widmer, 2018) custom architectures and training objectives still perform better. Following this observation, we propose to explore different sources of inexpensive labels to train music representation models that can be used to achieve SOTA performance in certain music classification tasks (Alonso-Jiménez et al., 2023b).

### 1.2.2 MIR research based on weak supervision

Self-supervised representation learning approaches enable training models with large amounts of unannotated data and hold the SOTA in domains such as NLP. In practice, these approaches have a limited scope for domains where collecting unlabeled data at scale is difficult due to copyright limitations or simple shortage. In these cases, certain forms of weak supervision may compensate for the lack of training data.

In the last decades, many of the advances in music understanding have been achieved by relying on weak forms of supervision. To do this, researchers proposed ways of leveraging data available in the public domain instead of creating high-quality datasets from scratch.

Music auto-tagging is a popular MIR task formulated as a multi-label classification problem where the goal is to predict a set of tags describing the content of a music track. The main characteristic of music auto-tagging is that the

ground truth is typically obtained from open databases of music metadata maintained by online communities instead of curated taxonomies created by experts. Last.fm[11], MusicBrainz[12], Music4All[13], and Discogs are examples of databases providing music editorial metadata. Most of these platforms allow users to add free-form tags to the music tracks or albums, which is the main source of ground truth for music auto-tagging tasks. Bertin-Mahieux et al. (2011) created the Million Song Dataset (MSD), a dataset of close to one million 30-seconds previews linked to editorial metadata from Last.fm, becoming a reference dataset for music auto-tagging research. The MSD tags typically refer to high-level concepts such as genre, mood, instrumentation, or cultural origin. However, there are also many tags related to user preferences (e.g., "favorites", "seen live"), which are not useful for music understanding tasks. In general, the quality of the tags is not guaranteed, and the annotations are noisy and unbalanced. Van Den Oord et al. (2014) proposed using auto-tagging as pre-training method and then transfer the learned representations to MGR achieving competitive results. The authors proposed considering only the top 50 most frequent tags in the MSD to alleviate the imbalance and noise in the dataset, which became a common practice in the field. This work inspired a number of follow-up studies exploring the use of auto-tagging as a pre-training task for music classification tasks. See Chapter 2.3 for a more detailed review of the literature on music auto-tagging.

While auto-tagging can be considered a powerful approach for music representation learning (McCallum et al., 2022), it presents certain limitations. First, the ground truth tends to be noisy, which creates the need for additional data cleaning and preprocessing steps. Second, the ground trunth unbalance limits the exploitation of the full potential of the datasets. For example, only 20% of the tracks in the MSD are tagged with one of the top 50 tags, which is what is typically considered in the literature. Finally, since different auto-tagging datasets use different tag sets, it is not trivial to combine them to create larger datasets.

Accounting for this, researchers have considered using other types of music metadata that do not suffer from the shortcomings of auto-tagging datasets. In this thesis, we focus on two types of metadata that is commonly available in association with music: *editorial* and *consumption* metadata. Editorial metadata refers to the information used to catalog music (e.g., artist and album names, or country and year of release), and consumption metadata describes interactions of humans (or machines) with music (e.g., playlists, DJ setlists, radio programs, or listening histories). In general, music metadata is easily accessible in industrial contexts such as streaming services, or radio stations,

[11] https://www.last.fm/
[12] https://musicbrainz.org/
[13] https://music4all.upf.edu/

and can be combined by crossing data from different sources. Music metadata does not suffer from the same limitations as auto-tagging datasets. For instance, the name of the artist or a track is objective information, and while it is susceptible to errors (e.g., misspellings, alternative names), it is not expected to suffer the same level of noise as user-generated tags. Also, since this type of information is not attached to a particular taxonomy, it is easier to combine data from different sources to create larger datasets. Finally, as mentioned in Section 1.1, the information conveyed on music is sometimes very complex and hard to capture with language or tags. Instead, relying on the proposed types of associative metadata (e.g., two tracks that belong to the same artist, or that are played in the same playlist) could be a more direct way to capture the relations between music tracks that are relevant for music understanding tasks. As a drawback, this type of information is not suitable for standard classification approaches (i.e., the number of classes would be on the same order of magnitude as the number of tracks). To solve this, Kim et al. (2018a) targeted artists names summarized as Linear Discriminant Analysis (LDA) topics, and Park et al. (2018) used the triplet loss to attract tracks belonging to the same artist. Chapter 2.6 provides a detailed review of the literature on music representation learning approaches using metadata.

### 1.2.3 Essentia

Essentia[14] is an open-source library and collection of models for audio and music analysis released under the AGPLv3 license and well known for its capability to serve as a basis for large-scale industrial applications as well as a rapid prototyping framework (Bogdanov et al., 2013b). Some of its key features are:

- It is implemented in C++, with a great focus on efficiency, which makes it the fastest open-source library with the largest amount of features for audio analysis (Moffat et al., 2015).
- It supports a declarative approach to the implementation of signal processing pipelines with the "streaming mode" connecting algorithms for each computation step via ring buffers. This allows the user to streamline audio analysis processing input files or audio streams by chunks (in particular in real-time) and also limits memory usage, which can be crucial for many applications.
- It has a Python interface. Programming in an interpreted language while all the data flow is ultimately controlled by optimized C++ code provides a balance between functionality and flexibility.

[14] https://essentia.upf.edu

- It supports various platforms including Linux, Windows, MacOS, Android, iOS, and can be also cross-compiled to JavaScript.

The first generation of music classification models for Essentia consisted of a collection of SVM classifiers based on engineered features and trained on in-house music collections (datasets) available at Music Technology Group (MTG).[15] These classifiers are publicly available and have been used extensively for research Wack et al. (2009, 2010); Laurier (2011); Bogdanov et al. (2011, 2013a); Fricke et al. (2018) and in AcousticBrainz, an open database of music audio features Porter et al. (2015) with over 13.5 million analyzed tracks. Given its focus on efficiency, flexibility of use, modularity and easy extensibility, we consider Essentia an attractive infrastructure to build efficient and modular deep learning pipelines for to replace the former SVM classifiers.

In this sense, this thesis aims to improve the models available in Essentia by improving the quality of the music representation models used to power the classifiers. Alonso-Jiménez et al. (2020a) proposed the first steps in this direction, where we released a collection of audio-based deep learning models for music analysis. The rest of steps of this thesis are aimed to improve the quality of the classifiers in Essentia models by creating better music representation models.

## 1.3 Challenges in music representation learning

In recent years, DNNs have become the go-to approach for classification across multiple domains, including MIR. However, the characteristics of music and the complexity of the tasks at hand have not allowed considering most of the music classification tasks solved despite the vast amount of research on the topic and computation capabilities available nowadays.

We identify the following problems related to music classification:

- **Lack of training data.** Deep models require large datasets to generalize well. However, there are very limited large datasets for music classification. Most of the time there is a trade-off between label quality and the amount of data. Additionally, the annotation process of music is considered particularly difficult and time-consuming.
- **Subjectivity.** Many music classification tasks are relatively subjective, and factors such as the culture, age, or musical training of the listener influence the perceived genre, mood, or theme of music Gómez Cañón et al. (2020). Because of this, most music classification datasets suffer

[15] https://acousticbrainz.org/datasets/accuracy

from a strong bias induced by the fact that they do not have balanced annotations concerning those factors, which results in poor generalization.

- **Weak labeling.** The track could be considered the minimal unit of music suitable for annotation. However, this is very inconvenient for computational approaches relying on a fixed input length, so it is common to design systems operating on audio segments with a fixed duration. This is problematic because labels are not always consistent at the patch level. For example, a particular segment belonging to the intro may not represent the overall genre or instrumentation of the track. Furthermore, labels sometimes come from editorial metadata, which can only be available at the album level, resulting in even less label resolution.

- **Complex information.** If we compare music to speech or environmental recordings, it can be considered a rather crowded signal in terms of information. It may require much musical training for a human to distinguish the instrumentation of a track for certain styles. Furthermore, harmonic sources spread their energy across the spectral representations usually used to train these classifiers. Thus, this adds extra complexity to the process of pattern recognition in systems assuming the locality of patterns such as the Convolutional Neural Networks (CNNs).

Notice that some of these problems interact with each other and altogether contribute to the difficulty of music classification. As musical signals are very complex, models would need more data to learn the task correctly. However, it is also difficult for humans to create clean large datasets for the same reason. Furthermore, because of the high dataset bias induced by the subjectivity of the tasks, the metrics are probably over-optimistic, and it is not easy to assess the actual capabilities of the models.

Our intuition is that the solution should not rely on the classical end-to-end DNNs by collecting sufficient data for the task of interest. Instead, we propose relying on approaches that allow leveraging knowledge acquired on easy and data-abundant tasks to facilitate solving the complex classification tasks with small but well-curated datasets.

## 1.4 Research questions

So far, we have framed this thesis into the design and evaluation of models able to produce deep representations useful for music classification tasks. Our ultimate goal is to maximize the performance of music classification tasks by leveraging available sources of public data. For this, we will not only focus

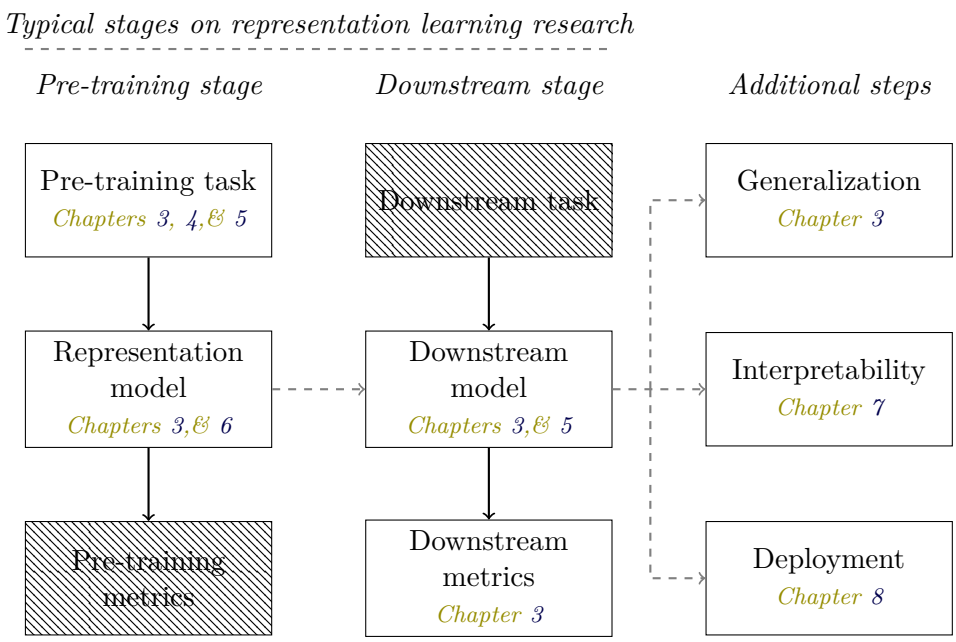


**Figure 1.2:** Diagram of the stages in the proposed representation learning setup. Apart from the typical pre-training and downstream stages, we propose additional steps to evaluate the generalization capabilities of the models, their interpretability, and their deployment in real-world applications. The dashed lines mark the blocks for which we do not propose contributions.

on learning from semantic tags or fully self-supervised approaches, but we will also explore editorial and consumption metadata as sources of supervision.

Figure 1.2 shows the standard stages of a representation learning pipeline and the chapters of this thesis contributing to any of their main components. In the pre-training stage, a *representation model* is trained on a *pre-training task*. In this dissertation we explore using semantic tags (Chapter 3), editorial metadata (Chapter 4), and consumption metadata (Chapter 5) as sources of supervision for the pre-training task. Additionally, we assess the impact of different design decisions on convolutional (Chapter 3) and transformer (Chapter 6) architectures in the learned representations. Typically, the performance of the model in the pre-training task is only used to monitor the training process, but it is not useful for practical purposes. In the downstream stage, the *representation model* is fine-tuned (Chapter 5) our used as a feature extractor (Chapters 3, 4, and 6) to train *downstream models* on one or multiple *downstream tasks*. In Chapter 3, we propose evaluation metrics that allow for a better comparison between representation models by considering statistical significance and normalizing the relative difficulty of downstream tasks. Additionally, we have identified a number of complementary steps that are not commonly covered in the representation learning literature, which are also part of the objectives of

this thesis. In Chapter 3, we pay attention to the generalization capabilities of the models based on deep representations in out-of-distribution scenarios. In Chapter 7, we explore the interpretability of the models, and in Chapter 8, we discuss the deployment of certain models developed on the scope of this thesis in real-world open source and commercial applications.

After this overview, we will now introduce the specific research questions that we aim to address in the following chapters:

- **Q1.** Can music classification models improve their performance by relying on deep audio representations? Does this benefit their generalization capabilities? How does this relate to the amount of available data in the downstream task?
- **Q2.** What is the best strategy to build a representation model for music classification tasks? Can we rely on inexpensive, abundant labels such as music editorial or consumption metadata? Can we rely on self-supervised techniques? Is there a benefit if we combine these objectives in a multitask setup?
- **Q3.** What type of deep architecture is the most convenient for the embedding model? Will transformers or other assumption-free models surpass the Convolutional Neural Network (CNN) models? Does the answer hold for online classification?
- **Q4.** Is it possible to operate representation learning models that do not behave as simple black boxes? Can we do something to enhance the interpretability of the resulting classification models?

## 1.5 Dissertation outline

This dissertation is organized in nine chapters. Chapters 3 to 9 present the main contributions of this thesis as reflected in Figure 1.2. These chapters address the research questions proposed in the previous section, and are primarily based on the following research papers: Alonso-Jiménez et al. (2020a,b, 2022, 2023a,b, 2024), and complemented with additional experiments, discussions, and contributions to other publications.

Chapter 2 provides a literate review covering the history of music classification methods based on neural networks. After this, it contains sections dedicated to the specific techniques and methods employed in the technical chapters of this thesis.

Chapter 3 starts by evaluating pre-trained audio tagging systems as feature extractors for music classification tasks. After showing promising results in

preliminary experiments, we compare a number of CNN architectures and pre-training strategies. According to the results, we propose a new model, DiscogsEffNet, trained on a 3.3 million collection of audio and music style tags coming from Discogs, which achieved state-of-the-art performance in several downstream tasks. We perform several experiments finding that the best-considered representation models allow for a better generalization in out-of-distribution scenarios and achieve competitive performance with a fraction of the data required by the end-to-end models.

In Chapter 4, we propose a novel approach to pre-train music embeddings using editorial metadata from Discogs. Our approach can be considered an extension of previous works using metadata notions to form the positive pairs (e.g., two tracks from the same artist form the anchor-positive pair) in conjunction with the triplet loss (Park et al., 2018). We replace the triplet loss with a SimCLR a contrastive learning method that already demonstrated good performance in fully supervised settings (Chen et al., 2020). Additionally, we experiment with album, artist, and record label notions, and propose a multitask setup. Our results show that our approach overcomes the standard SimCLR self-supervised baseline and our DiscogsEffNet model trained on music style tags associated with the same dataset. We conclude that the proposed approach combines the computational efficiency and competitive performance of supervised approaches, with the scalability and flexibility of self-supervised methods by preventing the need to work with fixed taxonomies or noisy labels.

Chapter 5 explores leveraging the techniques developed in Chapter 4 to pre-train music representation models using consumption metadata. The consumption metadata consists of playlists from the Million Playlist Dataset (MPD) (Chen et al., 2018), and the pre-trained models are fine-tuned instead of used as embedding extractors as in the previous chapters. Additionally, we develop new strategies to exploit the consumption metadata, treating the playlist information as an additional modality and achieving better performance than the baseline using editorial metadata (artist associations), which demonstrates the potential of this type of information. Our best model outperforms the considered baselines and achieved SOTA in the MagnaTagATune dataset (Law et al., 2009). Finally, we observe that consumption metadata provides better performance in the task of music similarity than the baseline methods based on classification or editorial metadata.

In Chapter 6, we explore the usage of transformers as representation models for music classification tasks. Since contrastive learning approaches require large amounts of Graphics Processing Unit (GPU) memory, and we are constrained to consumer grade hardware, we discarded experimenting with the methods presented in Chapters 4 and 5 in combination with transformers. Instead, we relied on the supervised approach based on music style tags presented in Chapter 3. We experiment with different pre-training configurations to find

the most suitable settings for our downstream classification tasks. We find that initializing the trained weights if the transformer is crucial, that transformers benefit from longer input sequences compared to CNNs, that the best representations to transfer are not in the last layer but in the intermediate ones (again opposing the CNN case), and that transformers are more flexible and support inference regimes that are more efficient than CNNs while keeping higher performance.

Chapter 7 explores interpretability in the context of music classification models based on deep representations. To do so, we take inspiration from the Audio Prototype Network (APNet) model by Zinemanas et al. (2021). APNet works by training a prototypical classifier in the latent space of an audio autoencoder, which allows for sonifying the class prototypes after training. We adapt this setup to perform the sonification through a diffusion decoder conditioned on arbitrary representations. Our method allows for a better trade-off between interpretability and model performance since this setup allows for using SOTA representations instead of learning them in the downstream task from scratch. Potentially this method allows for gaining further understanding about the representations used by the models.

Chapter 8 presents applications of the models developed in this thesis beyond the research context, including commercial and open-source applications. Chapter 9 concludes the dissertation and provides some directions for future work. Finally, Appendix 9.3.6 contains a list of the main contributions by the author in the scope of this thesis.

CHAPTER 2

# Scientific background

## 2.1 Introduction

Classification is arguably one of the fundamental pillars of Music Information Retrieval (MIR). The first models for tasks such as chord detection (Laden & Keefe, 1989) or music genre recognition (Matityaho & Furst, 1995) appeared even before its consolidation as a field, and some authors consider that classification is the most researched topic in MIR (Fu et al., 2010).

In this chapter, we focus on the evolution of neural-network-based approaches for music classification, which have been the most explored methods in the last decade. We start by reviewing the first shallow models that emerged in the late 80s and early 90s relying on symbolic representations. After this, we cover the first deep learning models that appeared in the mid-00s, and the subsequent wave of research that followed the irruption of the MSD (Bertin-Mahieux et al., 2011). Finally, we review the most recent approaches based on representation learning, including some of the techniques employed in this thesis, such as contrastive learning.

## 2.2 First neural networks for music classification

The first neural networks for music classification emerged in the late 80s following the development of algorithms that allowed efficient training, such as backpropagation (Rumelhart et al., 1985). Back then, Multi-Layer Perceptrons (MLPs) (or feedforward fully-connected neural network) were the most common architecture.

Due to the high dimensionality of audio signals or other representations such as spectrograms, researchers could not feed these signals directly to the MLPs, especially considering the computational resources available at the time. Instead, the first efforts focused on sparse symbolic representations.

The first known music classification neural network consisted of an MLP trained to distinguish among "major", "minor", or "diminished" chords from a 12-dimensional input vector representing the pitch classes of the chromatic scale (Laden & Keefe, 1989). Later on, models were able to operate on richer symbolic representations such as MIDI. Dannenberg et al. (1997) proposed different machine learning models, including MLPs, to classify MIDI files into nine different playing styles, including "frantic", "lyrical", or "pointillistic".

Matityaho & Furst (1995) trained an MLP to distinguish between "classical" and "pop" music using spectrograms as input for the first time. This was a critical breakthrough as it enabled porting computer vision approaches directly to the music domain, which is, in fact, still very popular nowadays.

The success of neural networks in these classification tasks encouraged researchers to use them for tasks traditionally treated with signal processing. For example, most researchers used to approach onset detection as a peak detection problem on a transformed version of the waveform. Marolt et al. (2002) approached this task with an MLP operating on a frame-wise manner deciding if the current timestamp is onset or not, converting it into an actual classification problem. Later on, tasks such as beat tracking (Böck & Schedl, 2011) or pitch detection (Kim et al., 2018b) were approached as frame-wise classification problems too.

## 2.3 Deep learning and music auto-tagging

With the irruption of GPU-accelerated systems in the mid-00s, researchers started training larger and deeper models that turned into dramatic performance improvements when sufficient training data was available. However, by that time, the MIR community lacked large-scale music datasets, and annotating music is a tedious task requiring significant amounts of time and domain knowledge.

At the same time, a number of music online communities started to develop, including platforms such as Last.fm[16] or Discogs. In order to facilitate browsing, discovering, or recommending music, some of these websites started to label their music following two different philosophies: taxonomies and folksonomies. While taxonomies are composed of a fixed hierarchy of labels that are unalterable by the users (e.g., Discogs), in folksonomies users have the freedom to create arbitrary tags (e.g., Last.fm). While taxonomies are generally well-structured and hierarchically organized, folksonomies may contain highly overlapping tags (e.g., "Hip Hop", "hiphoprap") and can be considered more noisy, as it is more difficult to enforce users to follow a common tag-

[16] last.fm

ging strategy. On the other hand, folksonomies have advantages. For example, they allow arbitrary label resolution according to the user's expertise or quickly adapt to trends without the intervention of the platform administrators.

For example, Discogs is a database where users can submit editorial metadata, including a fixed taxonomy of more than 570 music styles grouped in 8 broad musical genres. On the other hand, Last.fm is an online radio and music recommender where users can label music with an arbitrary number of free tags. With the growing demands of training data, researchers started to identify these platforms as potential sources of labels for music classification datasets. However, so far, most of the work has focused on folksonomies due to their larger size.

In this context, Eck et al. (2007) proposed matching a collection of 90,000 music tracks to Last.fm tags to build a dataset for music tag prediction giving birth to the task of music automatic tagging (or auto-tagging). Music tagging is a multi-label classification task using a vocabulary that can combine multiple music notions (e.g., genre, moods, eras). The motivation for this work was to use automatic tags to solve the problems of cold-start and explainability present in recommender systems based on collaborative filtering. They used an ADABoost (Freund et al., 1999) classifier and claimed that 70% of the tags corresponded to the track's genre, mood, or instrumentation.

The appearance of the MSD by Bertin-Mahieux et al. (2011) was an important milestone, as it provided a relation of Last.fm tags to approximately one million tracks accessible as audio features or 30-second previews. Dieleman & Schrauwen (2014a) used the MSD to train the first auto-tagging CNN operating directly on the waveform.

Choi et al. (2017a) experimented with recurrent layers on top of the convolutional feature extractor to summarize the temporal information finding benefits of using hybrid recurrent/convolutional architectures. Choi et al. (2018) were among the first researchers aiming to quantify noise on auto-tagging collections. Their methodology consisted of comparisons between feature vectors obtained from a data-driven approach and the ground truth.

According to a study comparing CNN architectures for auto-tagging Won et al. (2020b), simple $3 \times 3$ convolutional models based on spectrogram representations, firstly proposed by Choi et al. (2016), achieve the best performance in datasets of this order of magnitude. However, Pons et al. (2017) found that waveform-based architectures overcome spectrogram models when they use sufficient training data. In the proposed setup, this occurs when using approximately one million training samples.

A number of works propose neural architectures specifically designed for music applications. Pons et al. (2016) proposed a CNN architecture featuring vertical (frequency-wise) and horizontal (time-wise) convolutions aiming to capture

spectral and temporal patterns in music. Won et al. (2020a) proposed a CNN architecture with filters tuned to capture harmonic structures. Won et al. (2019) was one of the authors proposing the use of transformers for music auto-tagging, an architecture that has been widely adopted in the field since then.

## 2.4 Representation learning for music classification

One interesting property of deep convolutional architectures is their ability to learn hierarchical representations, from low-level patterns in the first layers to abstract features in the last ones. It is reasonable to think that features learned by models trained to optimize discriminative targets can be helpful for additional related discriminative tasks, and that the features produced in the different layers would reflect different levels of abstraction making them appropriate for different specific tasks. From the early 10s, classification based on semantic labels has probably been the most popular task to learn transferrable representations in MIR. Specifically, music auto-tagging is a multi-label task that covers a wide range of musical notions such as genre, instrumentation, moods, or era.

Hamel & Eck (2010) explored this idea by comparing Mel–Frequency Cepstral Coefficient (MFCC)s and embeddings from a four-layer Deep Belief Network (DBN) as input features for a SVM classifier. The experiment showed that the embeddings, particularly those from the final layers, performed better than the MFCC baseline. While this experiment showed the capabilities of deep models to operate as feature extractors, it cannot yet be considered transfer learning as defined in Section 2.1 because the authors trained both the DBN and the SVM in the same MGR dataset.

Hamel et al. (2013) conducted one of the first transfer learning experiments on MIR by using an auto-tagging dataset to learn a shallow embedding model used later to train a linear classifier for MGR. From this point, many works have relied on the task of music auto-tagging on datasets such as the MSD Bertin-Mahieux et al. (2011) because this was probably the largest source of annotated music audio back then. This experiment showed that the learned features were beneficial under small training data conditions. van den Oord et al. (2014) performed a similar experiment where they replaced the embedding model with a two-layer MLP trained in the MSD finding that the representations learned by solving an auto-tagging task improved the performance compared to traditional MFCCs for several classification tasks. Choi et al. (2017b) followed a similar approach replacing the MFCCs with a six-layer CNN. This experiment can be considered the first example of transfer learning featuring deep embeddings in MIR. Again, the authors found that the most useful features tend to

be in the deeper layers of the networks and that stacking the embeddings extracted from different layers could be beneficial for downstream performance. Lee et al. (2018) proposed a CNN architecture operating directly on the waveform with features achieving competitive performance in several classification tasks.

More recently, the research directions on the creation of embeddings for classification have diversified. Pons et al. (2019) compared various approaches intended for classification with few data, including highly regularized models, and prototypical learning, finding that transferring representations from pre-trained auto-tagging models resulted in the most effective technique. Kim et al. (2019a) measured the effect of perturbations in different feature spaces and found that deep embeddings tend to be more prone to alteration than MFCCs. Kim et al. (2020a) studied different strategies to simultaneously solve supervised and self-supervised tasks to train deep representation models. Lee et al. (2020b) compared classification to metric learning approaches for auto-tagging, finding that the classification embeddings performed better in the classification tasks. Additionally, the authors proposed a method to disentangle the embedding space in terms of the different musical notions present in the dataset that improved the performance in the classification tasks. Huang et al. (2020) combined classification and metric learning to leverage the co-listen information linked to the MSD.

Ultimately, the main bottleneck of most of these approaches is that they rely on labeled data that is relatively scarce or noisy (it is primarily auto-tagging datasets derived from folksonomies). One way of overcoming this limitation is to rely on models trained on general-purpose audio labels, such as VGGish (Hershey et al., 2017), whose features have been found useful for music classification for several authors (Alonso-Jiménez et al., 2020a; Koh & Dubnov, 2021). Additionally, researchers have shown interest in alternatives such as self-supervised methods or approaches requiring more accessible sources of supervision, such as music metadata.

## 2.5 Self-supervised approaches

Deep learning approaches have been shown to be particularly effective in learning representations from large amounts of data. However, obtaining human annotations can be time-consuming and expensive, especially for music tasks. For this reason, there is a growing interest in developing training paradigms that do not require human labels.

Some authors have explored leveraging models pre-trained on data from other domains. Palanisamy et al. (2020) fine-tuned CNNs pre-trained in image recognition tasks for audio classification, achieving competitive performance on

MGR. The study showed that fine-tuning the first layers of the CNNs does not significantly impact performance, but the final layers need drastic weight updates to perform well.

Additionally, self-supervised and unsupervised learning approaches try to learn representations without any external source of supervision, allowing them to expand to an arbitrary amount of unlabeled data. While unsupervised algorithms operate without the need for labels, self-supervised learning works as the conventional supervised approaches in which labels are automatically derived from the data. For example, a music similarity model could generate "similar"/"dissimilar" labels from sampling pairs of segments of the same track versus segments from different tracks.

The Look, Listen and Learn method relies on audiovisual correspondence by jointly training an audio and a video encoder connected to a classifier that decides if the input segments correspond to the same timestamp (Arandjelovic & Zisserman, 2017). Cramer et al. (2019) proposed OpenL3, an updated version of the same training approach where they analyzed the effect of the training data, the input representation parameters, and the network architecture on the performance of the model. Pons & Serra (2019b) experimented with randomly weighted neural networks as feature extractors for classification tasks finding a correlation between the randomly weighted and the fully-supervised version of the models, meaning that this approach helps identify architectures suitable for a given task. Finally, Castellon et al. (2021) extracted embeddings from Jukebox, a music generation model conditioned on musical genre, and lyrics (Dhariwal et al., 2020). While Jukebox's main purpose is music generation, the authors found that the embeddings extracted from the model were useful for diverse music classification tasks. Alonso-Jiménez et al. (2020b) compared the capabilities of different audio models trained on different tasks (auto-tagging, audio event recognition, source separation, tempo estimation, and audiovisual correspondence) as embeddings extractors for several music classification tasks including, music genre tagging, mood and theme recognition, and instrumentation detection. The study concluded that models trained on auto-tagging produced more suitable embeddings for the considered tasks.

Finally, it is also possible to combine supervised and self-supervised objectives in a multitask setup. Kim et al. (2019b) investigated this idea by exploiting the information available from the MSD-Last.fm dataset. The authors trained models to jointly solve different tasks including semantic music tagging, prediction of editorial metadata codified as Probabilistic Latent Semantic Analysis (PLSA) topics, and predicting whether two audio segments belong or not to the same track. Huang et al. (2020) combined classification and metric learning losses and used semantic tags, metadata, and co-listen information from the MSD to train a model that outperformed the models trained for individual tasks.

In the following subsections, we review in more detail two self-supervised approaches that have been applied to music representation learning in the context of this thesis: contrastive learning and masked token prediction.

### 2.5.1 Contrastive learning

Contrastive learning is a form of metric learning that aims to learn a representation space where similar samples are close to each other and dissimilar samples are far apart. In its typical form, the positive samples are pairs of augmented versions of the same input, and the negative samples are the rest of the samples in the batch.

The MIR community has lately adopted contrastive learning approaches. CLRM adapts SimCLR (Chen et al., 2020) by applying musically-motivated data augmentations in the waveform domain (e.g., reverb, saturation) to create the augmented views of the input signal (Spijkervet & Burgoyne, 2021). Then, the InfoNCE loss (Van den Oord et al., 2018) is used to minimize the distances between positive pairs while maximizing their distance to other samples. BYOL-A (Niizumi et al., 2021) relies on the BYOL (Grill et al., 2020) framework and adapts it to the audio domain by proposing specific augmentations and evaluating it on several audio tasks, including instrument classification. S3T (Zhao et al., 2022) combines the MoCo framework (He et al., 2020) with Swin Transformers (Liu et al., 2021) to learn music classification features. Wang et al. (2022) modify SimCLR by using a normalization-free SlowFast (Feichtenhofer et al., 2019) backbone and improve the performance in several audio tasks, including music auto-tagging. PEMR deals with the lack of temporal resolution of existing systems by learning to mask important or irrelevant parts of the mel-spectrogram to produce self-augmented positive/negative samples (Yao et al., 2022). COLA samples mel-spectrogram patches of the same clip as positives instead of generating augmented versions Saeed et al. (2021). It relies on a bilinear similarity layer to compute the distances between all samples in the batch and uses the cross-entropy loss to maximize the similarity between positive pairs while using the rest of the samples as negatives. Wang et al. (2022) compare several self-supervised contrastive learning methods for audio and music representation learning, finding that the best performance is achieved by the SimCLR approach. McCallum et al. (2022) evaluate the performance of supervised and contrastive learning approaches for music classification, finding that the models with supervised pre-training tend to perform better in the classification tasks. Meseguer-Brocal et al. (2024) compare the performance of self-supervised methods for music classification, finding that the models trained with contrastive learning tend to perform the best in the downstream evaluation.

Scientific evidence suggests that, in contrastive setups, it is beneficial to choose

positive pairs that share information relevant to the downstream task while being diverse with respect to irrelevant characteristics (Tian et al., 2020). However, most audio and music self-supervised contrastive methods rely on sample mixing (Wang et al., 2022), audio effects (Spijkervet & Burgoyne, 2021), or temporal crops (Saeed et al., 2021) to generate the augmented versions, which intuitively have a small potential to obtain samples that are distinct enough.

Alternatively, it is possible to use label information in combination with contrastive learning. One of the most popular approaches is to use contrastive learning to align representations coming from different modalities. Favory et al. (2020, 2021) use a contrastive objective to align embedded representations of tags and audio from Freesound (Fonseca et al., 2021). In a subsequent study, the system is enriched with playlists-track interactions as an additional modality to align (Ferraro et al., 2021). Manco et al. (2021) align text embeddings from music captions to the correspondent audio representations finding competitive performance in several downstream tasks. Elizalde et al. (2023) aligns pre-trained audio and text encoders using AudioSet (Gemmeke et al., 2017).

### 2.5.2 Masked autoencoders

Masked autoencoders follow a self-supervised learning approach that has been widely adopted in NLP and more recently has been applied to the music and audio domains. Given a partially masked input sequence, the model learns to predict the masked regions given the rest of the sequence.

Zhao & Guo (2021) used a bidirectional transformer architecture with two self-supervised objectives, predicting masked spectrogram timestamps and channels. Chong et al. (2022) proposed MaskSpec, an autoencoder transformer trained to reconstruct spectrogram chunks from the partially masked input.

Neural codecs have recently emerged as a way to efficiently compress audio (Zeghidour et al., 2021; Défossez et al., 2022; Kumar et al., 2023) and are sometimes used in combination with masked autoencoders. They usually comprise a convolutional autoencoder with a Residual Vector Quantization (RVQ) layer in the bottleneck. The neural codes at the output of the RVQ layer have a low bitrate because of the low time resolution (75 Hz /∼13 ms frames for EnCodec) and the quantization. The high reconstruction quality of these neural codes for generic audio, together with the low frame rate, make them suitable as features for music-related tasks. Yizhi et al. (2023) proposed MERT, a transformer trained to reconstruct the Constant-Q transform (CQT) and predict discrete neural codes from the masked part of the input. Pepino et al. (2023) developed EncodecMAE, which learns to reconstruct the masked segments from the non-quantized EnCodec features using a transformer architecture.

## 2.6 Music representation learning with metadata

The most common music representation learning approaches are rather fully self-supervised or rely on expensive semantic tags. We propose that music metadata can be seen as an intermediate between these two approaches.

On one hand, metadata is objective and expected to be very clean. On the other hand, it is not dependent on a specific taxonomy, and it is easy to combine collections with small preprocessing or data cleaning. Additionally, it is often available for music collections in the industry by default or can be easily obtained from online sources such as Discogs or MusicBrainz.

In Section 1.2.2, we introduce the concepts of editorial and consumption metadata and discuss their primary characteristics. Here, we review the literature on music representation learning using metadata as a source of supervision.

### 2.6.1 Editorial metadata

Using editorial metadata as a source of supervision was already considered in other domains, for example, in document classification (Joorabchi & Mahdi, 2011) or film recommendation (Leung et al., 2020). Likewise, music is naturally rich in metadata. Especially, this information tends to be extensively available in industrial scenarios. For example, physical formats typically contain detailed editorial information on their covers, digital audio files support containers such as ID3 for this purpose, and most streaming platforms offer album or artist-level browsability. Most music digital service providers routinely require such metadata from content uploaders, and therefore it is often available for music collections in the industry by default. Also, due to the objective nature of descriptors such as the artist or album name, it is expected to be less noisy than the semantic tags. Because of these reasons, such metadata allows the creation of potentially larger and less noisy datasets than tag annotations.

The use of editorial metadata has already been explored in the context of music representation learning in MIR. Park et al. (2017) trained a CNN targeting track artist names for the MSD and used the resulting embeddings for music classification tasks achieving good results. In subsequent work, (Park et al., 2018) used the artist's name to form positive pairs to apply the triplet loss. Kim et al. (2018a) proposed dealing with the high dimensionality of artist vectors by summarizing them into LDA topics to create targets suitable for a traditional classification setup. To further leverage existing metadata, Lee et al. (2019a) trained siamese networks on three different similarity concepts: correspondence to the same artist, album, and track, finding that learning the three notions at the same produced the best-performing features. Finally, Kim et al. (2020b) exploited album, release, and year information in combination

with semantic tags, and Huang et al. (2020) used semantic tags, editorial metadata, and co-listen information from the MSD outperforming the models trained for individual tasks. These works showed that editorial metadata may be used to complement other supervision sources.

### 2.6.2 Consumption metadata

Using consumption metadata as a source of similarity ground truth has already been explored in the recommender-systems literature, enabling tasks such as music playlist continuation (Chen et al., 2018). Also, while editorial relations are normally one- or few-to-many (e.g., album-songs), consumption is many-to-many (e.g., playlists-songs, listening histories-songs), resulting in a more dense co-occurrence space that may favor associating more heterogeneous music. Furthermore, the usage of consumption metadata for music representation learning has not been as extensively investigated yet (Huang et al., 2020; Ferraro et al., 2021) as the case of editorial metadata (Park et al., 2018; Kim et al., 2018a; Lee et al., 2019b; Kim et al., 2020b; Huang et al., 2020).

Ferraro et al. (2021) extended the work of Favory et al. (2020, 2021) by incorporating playlist information as an additional modality to align with the audio and semantic tag representations. Huang et al. (2020) combined classification and metric learning to leverage the co-listen information linked to the MSD. Note that, in none of these works, consumption metadata is used as the exclusive source of supervision of music representation learning models.

## 2.7 General purpose audio representations

Some authors have pursued general-purpose representation models to address simultaneously speech, audio event, and music tasks, which led to the proposal of challenges such as HEAR (Turian et al., 2022) and benchmarks such as HARES (Wang et al., 2022).

Typically, general-purpose audio models are trained following self-supervised objectives using data from multiple domains. Wang et al. (2022) used the SymCLR contrastive learning approach to find competitive performance in several audio tasks. Pepino et al. (2023) developed EncodecMAE, which learns to reconstruct the masked segments from the non-quantized EnCodec features using a transformer architecture and showed that the resulting features have a strong performance in a diverse set of audio-related tasks, including MGR and pitch detection among other speech and audio tasks. However, for now, there is no evidence that a single training paradigm can yield excellent performance in all the audio domains simultaneously.

Alternatively, audio representations can be optimized to a single domain leveraging specific data, which tends to produce better performance. In this sense, music-specific representation models are typically evaluated in music description in terms of genre, mood, era, rhythmic properties or arousal, and valence estimation, where the annotations are generally on the track level. Additionally, music representation models can be evaluated in more objective tasks such as tempo or key estimation, although, specific models using domain knowledge tend to be better suited for these tasks Schreiber & Meinard (2018).

## 2.8 Discogs in MIR research

The Discogs database has already been used for research. Bogdanov & Herrera (2012) propose using this collection in the context of music recommendation, MIR, and computational musicology (Bogdanov & Serra, 2017). The former study proposes a recommendation system based on the similarity between artist representations in the form of a tag cloud of the associated genre, style, record label, and release year and country metadata. The latter illustrates the potential uses of the database on various cultural analysis examples including the evolution of physical distribution formats, genre and style trends, and their co-occurrences. Similarly, some studies analyze music artist collaboration networks Burke et al. (2014); Budner & Grahl (2016); Andrade & Figueiredo (2016); Gienapp et al. (2021).

In the context of music genre classification, the AcousticBrainz Genre dataset contains mappings across different music genre taxonomies, including the one from Discogs Bogdanov et al. (2019a). Hennequin et al. (2018) use the genre and style labels from Discogs for genre tag disambiguation.

## 2.9 Transformers for music classification

Now, we review the literature on music representation learning using the transformer architecture. This section can be seen as a motivation for the experiments presented in Chapter 6.

### 2.9.1 Transformers in audio classification tasks

Transformers have become a popular choice for audio tasks due to their superior performance compared to their convolutional counterparts when sufficient data is available. AudioSet, with almost 2 million audio event excerpts, has become a popular benchmark led by transformer models.

| Model | Init. | GPUs | Time | mAP |
|---|---|---|---|---|
| ASTGong et al. (2021) | ViT | - | - | 45.9 |
| PaSSTKoutini et al. (2022) | DeiT | 2 RTX 2080ti | 24 h | 47.6 |
| MaskSpecChong et al. (2022) | FS | 64 Tesla V100 | 36 h | 47.3 |
| BeatsChen et al. (2022) | FS | 16 | - | 48.7 |

**Table 2.1:** Comparison transformers from the literature in terms of initialization weights, number of GPUs used for training, training time, and mAP obtained in AudioSet.

A popular approach consists of applying self-attention layers over small overlapping patches (e.g., 16 × 16) from the spectrogram using a classification objective. The sequence of spectrogram patches is linearly projected to a 1-D space where a trainable positional encoding signal is added. A trainable classification token is appended to the sequence of projections, and after several Transformer blocks, it is used to solve the classification task using a linear classifier. This idea was first introduced in the image domain by (Dosovitskiy et al., 2021) with the ViT and adapted to audio spectrograms in (Gong et al., 2021) with AST.

Koutini et al. (2022) extend this approach by applying patchout, a technique consisting of discarding random patches from the input spectrogram at training time. The resulting model is called the "Patchout faSt Spectrogram Transformer" (PaSST). Patchout can be seen as the application of masking as in the masked autoencoder methods introduced in Section 2.5.2 with the difference that the goal is to predict a label instead of reconstructing missing segments. This technique has two benefits. First, by discarding input patches, the training sequence length is significantly reduced, which can significantly increase the training speed, considering that transformers have a quadratic complexity with respect to the sequence length. Second, it acts as a regularization technique that improves the robustness of the transformer.

Table 2.1 compares PaSST with AST (Gong et al., 2021), and two self-supervised methods: MaskSpec (Chong et al., 2022), and Beats (Chen et al., 2022) in terms of the number of GPUs used for training, training duration, and mean Average Precision (mAP) on AudioSet. While self-supervised methods prevent the transformers from depending on initializing from weights of pretrained models, such systems are significantly more resource-demanding. Remarkably, PaSST achieves an excellent trade-off between mAP and the required resources.

### 2.9.2 Music representation learning with transformers

Some works have already combined music representation learning and pure-attention-based transformers. S3T combines MoCo's momentum-based self-

supervised contrastive learning with the Swin Transformer (Liu et al., 2021) architecture to learn music representations for classification (Zhao et al., 2022). Huang et al. (2022b) introduced MuLan, an audio representation model trained with cross-domain contrastive learning that aligns the latent representations of the associated audio and text pairs. The authors experiment both with a ResNet50 and an AST architecture, with the former obtaining better performance in downstream music tagging tasks. Yizhi et al. (2023) proposed MERT, a transformer trained to reconstruct the CQT and predict discrete neural codes from the masked part of the input.

## 2.10 Interpretability

In this last section, we review the literature on interpretability in the context of music representation learning. This section motivates the experiments presented in Chapter 7.

Interpretability strategies in the audio domain are rarely explored, especially in music-related tasks (Batlle-Roca et al., 2023). Among the existing methods for the classification of sounds and music (Dieleman & Schrauwen, 2014b; Mishra et al., 2017; Won et al., 2019; Loiseau et al., 2022), only a few are interpretable by design. In particular, APNet is an interpretable model for sound classification based on prototypes that are learned during training along with the latent space encoder and decoder (Zinemanas et al., 2021). The decoder is devised to reconstruct the prototypes back to the input representation (mel spectrogram) that can be sonified. The model shows compelling results illustrating that a system can be both interpretable and accurate.

However, this model presents scalability issues regarding the number of prototypes and classes, given that the latent space is of high dimension and the prototypes are learned in this space and stored in the model. Additionally, APNet's reconstruction process transfers information on which indices were kept in the pooling layers from the encoder into the unpooling layers of the decoder to improve the reconstruction quality. Assuming that the prototypes are similar to data instances, the pooling indices are extracted from the closest instance of the training set to reconstruct them, providing interpretability suitable for end users. However, the prototype reconstruction is strongly biased towards a training sample instead of what drives the classification decision, which would be more interesting from a developer's perspective.

CHAPTER 3

# Representation learning based on labels

## 3.1 Introduction

Representation learning has become a popular research direction that enables improving performance and generalization on tasks with small amounts of data that would otherwise fall out of the scope of modern deep learning techniques. Following this paradigm, researchers typically train large models targeting auxiliary objectives requiring weak or no supervision that are then used as feature extractors or fine-tuned for smaller downstream tasks.

The MIR community is also showing a growing interest in representation learning. Since many MIR tasks suffer a lack of large-scale datasets due to factors such as the annotation effort (in terms of time and expertise required for manual annotations), legal constraints on the distribution of music audio, or the high subjectivity of the tasks, transferring knowledge from a suitable representation model can be a viable strategy to improve the performance on downstream tasks. At the same time, working with low-dimensional embeddings has practical advantages for many use cases due to their compact size compared to the original signals. Also, storing audio datasets as embeddings can facilitate data reusability, speed up the transfer to downstream tasks, and facilitate cross-discipline research for communities that are not familiar with the specifics of audio.

In this context, we start our research by evaluating the performance of existing representation models on standard music classification tasks. Since previous studies such as Bogdanov et al. (2016) suggest that some MIR classification tasks suffer from severe dataset bias, we are particularly interested in understanding if relying on pre-trained representation models can improve the generalization capabilities of the resulting classifiers. To this end, we follow

the methodology proposed in that work and complement the evaluation of the downstream classifiers within the downstream datasets with a cross-collection evaluation using an unseen collection annotated with the same taxonomy. Once we have observed the generalization benefits obtained with pre-trained representation models, we work upon the premise that we can create better representation by scaling up the training data. We address this by training a model on an in-house collection of 3.3 million music tracks and comparing it to SOTA models trained under different paradigms. After this, we design several experiments to further understand the performance of the considered representation models. In our experiments, we investigate the effect of limiting the amount of data in the downstream tasks, the benefits of combining different embedding types, the generalization capabilities of the downstream classifiers using a novel dataset of cross-collection annotations, and the performance on two large-scale publicly benchmarked datasets.

The experiments of this chapter are based on Alonso-Jiménez et al. (2020a), presented virtually on ICASSP 2020, and Alonso-Jiménez et al. (2020b), presented virtually on ISMIR 2020. Finally, to promote reproducibility, we provide C++ and Python implementations for most of the considered models within the Essentia Library.[17]

## 3.2 Method

To compare the representation learning models, we train shallow classifiers on several music classification tasks. Similar methodologies are widely used in the MIR field, for example, Choi et al. (2017b); Castellon et al. (2021).

### 3.2.1 Transfer learning protocol

Our transfer learning protocol consists of training shallow classifiers on top of the representations extracted from intermediate layers of the representation model of interest. For this, we typically use MLPs with a single hidden layer.

The datasets for the considered downstream tasks are composed of full-track recordings of variable length, fixed-length excerpts, or a mixture of both. The embedding extractors transform the one-dimensional audio streams into a two-dimensional representation with shape $T \times D$, where $T$ is the number of timestamps depending on the hop time of the analysis and $D$ is the number of dimensions of the embedding. Each timestamp $t \in [0,T]$ summarizes a time window given by the receptive field of the model. Since we propose experiments that involve training thousands of MLPs, we pre-extract the em-

[17] https://essentia.upf.edu/models.html

beddings and save them as 16-bit floats (half precision) as this did not seem to affect the performance in preliminary experiments. As some experiments rely on combinations of embeddings by different models, they need to be temporally aligned (i.e., given embeddings $x$ and $y$, $T_x = T_y$). To achieve this, we defined a target rate of 1Hz (i.e., one embedding per second) and set the hop time of each extractor to produce embeddings at this rate when possible. Since the receptive field is not the same embedding extractor, the MLPs learn to make predictions over time windows of different lengths depending on the architecture of the extractor.

To fit variable-length input tracks into regular-shaped batches, we pick a single $t$ with a random offset per track on every epoch. In validation and testing times, we average the $T$ predictions per track to derive track-level metrics.

### 3.2.2 5-fold cross-validation

We evaluate the downstream tasks in a 5-fold cross-validation setting. On each fold, 20% of the data is reserved for the testing set. The remaining data is split into 85% for training and 15% for validation. Every split is stratified to keep a constant label distribution in the training, validation, and testing sets.

Following the transfer learning protocol from Section 3.2.1, we train a shallow classifier for each of the folds. After this, we compute the following metrics:

- **Average normalized accuracy**. The average of the class-weighted accuracies obtained for each fold.
- **Standard deviation**. The standard deviation of the class-weighted accuracies obtained for the different folds. This is a dispersion metric used to evaluate the statistical significance of the results.

### 3.2.3 Cross-collection evaluation

Previous studies have shown that evaluating music classifiers on a single dataset can lead to overfitting and biased results Bogdanov et al. (2016). Accounting for this, we complement the aforementioned 5-fold cross-validation with an evaluation using external datasets. In this case, we use 80% of the downstream data for training and 20% for validation. We do not reserve data for testing in this case.

We use the resulting model to predict the labels of an unseen collection annotated with the same taxonomy as the downstream task. We consider two approaches to obtain the evaluation data. First, by relying on auto-tagging datasets that contain the same taxonomy as the downstream task, and second

| | Dataset | Classes | Size |
|---|---|---|---|
| genre | dortmund | alternative, blues, electronic, folkcountry, funksoulrnb, jazz, pop, raphiphop, rock | 1820 exc. |
| | gtzan | blues, classic, country, disco, hip hop, jazz, metal, pop, reggae, rock | 1000 exc. |
| | rosamerica | classic, dance, hip hop, jazz, pop, rhythm and blues, rock, speech | 400 ft. |
| mood | acoustic | acoustic, non acoustic | 321 ft. |
| | aggressive | aggressive, non aggressive | 280 ft./exc. |
| | electronic | electronic, non electronic | 332 ft./exc. |
| | happy | happy, non happy | 302 exc. |
| | party | party, non party | 349 exc. |
| | relaxed | relaxed, non relaxed | 446 ft./exc. |
| | sad | sad, non sad | 230 ft./exc. |
| high-level | danceability | danceable, non danceable | 306 ft. |
| | voice/instrum. | voice, instrumental | 1000 exc. |
| | gender | male, female | 3311 ft. |
| | tonal/atonal | atonal, tonal | 345 exc. |

**Table 3.1:** Downstream tasks (ft.: full tracks, exc.: excerpts).

by manually collecting annotations for the downstream tasks of interest. In this case, we only compute class-normalized metrics, since there is no way to obtain dispersion metrics from a single evaluation.

### 3.2.4 Fixed-splits evaluation

Fixed-split evaluation is a common practice with publicly available datasets. It consists of training and testing the models using a pre-defined split previously used in the literature.

This is the less informative of all the considered evaluation approaches. However, it is useful to compare the performance of the models with previous works eliminating the variability introduced by random sampling. We will consider this evaluation for a specific dataset with publicly available splits.

### 3.2.5 Downstream tasks

There are many annotated in-house music collections (a large number of which were created by researchers at Music Technology Group) that are used extensively in Essentia and a number of related large-scale projects such as AcousticBrainz (Porter et al., 2015). Table 3.1 describes these collections in terms of classes and number of training examples. Even though their scale is not comparable with many recent datasets, they are interesting to work with because they represent a typical use-case of a small amount of data available for a particular application. In addition, our intention is to understand the impact of different representation models on the downstream performance and not to challenge the state of the art on any particular task.

We select 14 single-label classification tasks from public and in-house datasets, aiming to cover some of the most popular topics in music classification. These tasks broadly fall into three categories: *genre*, *mood*, and other *high-level* musical concepts, aiming to cover some of the most popular topics in music classification. As our cross-collection evaluation experiments involve generating new ground truth, we limited the selection of tasks to problems that could be annotated without particular music expertise. This leaves regression tasks (e.g., arousal-valence estimation) or multi-label ones (e.g., auto-tagging) out of the scope of this evaluation. Also, we want to represent the variability of dataset sizes present in typical transfer learning scenarios, from a few hundred to thousands of tracks.

## 3.3 Experiments

### 3.3.1 Preliminary evaluation on music genre recognition

In our first experiment, we are interested in assessing if relying on features learned by audio tagging models can improve the generalization capabilities of classifiers trained on small datasets. We use the proposed cross-collection evaluation methodology as a proxy of generalizability, for which we need a separate collection annotated with the same taxonomies as our downstream tasks. To this end, we focus on the task of music genre recognition, for which we have three downstream datasets: *genre-dortmund*, *genre-gtzan*, and *genre-rosamerica*. As external data sources, we use two datasets, both containing tag annotations including genres:

- **MSD-test** is the test set of 28,000 tracks from the MSD dataset with Lastfm tags. Note that MSD has been also used for the pre-trained MusiCNN and VGG-Pons models, but they were trained on the train split and there is no overlap.

- **MTG-Jamendo-test** is the *split-0* test set of 11,000 tracks from the MTG-Jamendo dataset for music tagging (Bogdanov et al., 2019b). [18]

Following the cross-collection methodology introduced in Section 3.2.3, we took advantage of the taxonomy used by the Lastgenre plugin for Beets[19] to generate ground-truth genre labels from the tags in the MSD-test and MTG-Jamendo-test. We only considered tracks with one or more tags matching an element in the taxonomy. Those tags were mapped to their parent in the hierarchy (e.g., "progressive rock" to "rock") unless there was a direct match to one of the classes of our classifiers. The resulting genre annotations are multi-label, and to evaluate each group of classifiers (corresponding to one of our downstream tasks) we use the subset of tracks that have a ground-truth label matching one of the classes. That is, we only give them music by genres they can theoretically predict. A prediction is considered correct if it matches one of the labels of the track.

Regarding the pre-trained models, we consider the following CNN architectures to create our classifiers:

- **VGGish** is a CNN based on a deep stack of 3×3 convolutional layers (Hershey et al., 2017). The architecture follows configuration "E" from the original VGG architecture paper by Simonyan & Zisserman (2014a), with the difference that the output layer is modified to match the number of output labels. VGGish was trained using 72M YouTube videos not specific to music annotated by 3087 labels derived from their titles and descriptions. The same pool of videos was later used to develop the AudioSet dataset (Gemmeke et al., 2017). This model has 62 million parameters.

- **VGG-Pons** is an additional instance of the VGG architecture, introduced by Choi et al. (2016) and following the implementation of Pons & Serra (2019a). Similar to MusiCNN, it was trained on the MSD train set annotated by the top 50 Last.fm tags. This model has 605,000 trainable parameters.

- **MusiCNN** is a CNN model featuring vertical and horizontal convolutional filters aiming to capture timbral and temporal patterns, respectively. MusiCNN was trained using 200,000 tracks from the train set of the publicly available MSD annotated by the 50 Last.fm tags most frequent in the dataset.[20] The model contains 6 layers and 787,000 parameters (Pons & Serra, 2019a).

[18] https://mtg.github.io/mtg-jamendo-dataset
[19] http://beets.io
[20] http://millionsongdataset.com/lastfm

Our goal is to assess if the pre-trained models can improve the generalization capabilities of the resulting classifiers compared to the same architectures trained from scratch. When training from scratch, all the layers of the model are randomly initialized and trained on the downstream dataset. Our transfer learning setup follows the approach described in Section 3.2.1 by extracting representations from the penultimate layer of the pretrained model and training a small neural network on top of it. We consider two variations for the classification neural network:

- *A)* One fully-connected layer with $n$ output units.
- *B)* Two fully connected layers of 100 and $n$ units.

In both cases, $n$ is the number of classes in the downstream dataset. Variant $A$ performed better for MusiCNN and VGG-Pons, while variant $B$ gave better results for VGGish. We used the variant providing the best performance for each model in the rest of the experiment. For the models trained from scratch, we consider MusiCNN and VGG-Pons. In this case, their parameters are randomly initialized and all the layers are trained.

All our CNNs were trained on mel-spectrograms. For the models trained from scratch, we used the implementation in Essentia with 96 bands. In the case of transfer learning, we used 96 bands for MusiCNN and VGG-Pons, and 64 bands for VGGish. The models are trained using a batch size of 32 samples. Each sample is a randomly selected segment of 3 seconds from a different track of the training set. We employ the Adam optimizer for 600 epochs on the models trained from scratch and 150 epochs in the transfer learning ones since these models require fewer iterations to converge. All the models are initialized with a learning rate of 1e−3. If the loss obtained on the validation set has not decreased for the last 75 epochs, the learning rate is reduced by half.

The baseline for our experiments comprises the SVM classifiers available in Essentia.[21] They rely on a combination of low-, mid-, and high-level music audio features describing timbre, rhythm and tonality (Porter et al., 2015). The best parameters for the SVMs are found in a grid search in the 5-fold cross-validation, and the final SVM models that we evaluate are trained on the entire data.[22]

[21] We used the latest Essentia 2.1-beta5 version.
[22] https://essentia.upf.edu/documentation/FAQ.html

| Dataset | Baseline | From scratch | | Transfer learning | | |
|---|---|---|---|---|---|---|
| | SVM | MusiCNN | VGG-Pons | MusiCNN | VGG-Pons | VGGish |
| 5-fold cross-validation | | | | | | |
| dortmund | 0.42±.01 | 0.40±.03 | 0.43±.02 | 0.51±.02 | 0.47±.02 | **0.52±.02** |
| gtzan | 0.74±.01 | 0.83±.02 | 0.82±.01 | 0.81±.03 | 0.83±.01 | **0.86±.02** |
| rosamerica | 0.86±.02 | 0.93±.03 | 0.88±.02 | 0.92±.02 | 0.92±.02 | **0.94±.02** |
| Cross-collection evaluation on MSD-test | | | | | | |
| dortmund | 0.36 | 0.35 | 0.35 | 0.39 | 0.37 | **0.42** |
| gtzan | 0.28 | 0.34 | 0.49 | 0.51 | 0.50 | **0.54** |
| rosamerica | 0.44 | 0.44 | 0.46 | 0.52 | 0.50 | **0.54** |
| Cross-collection evaluation on MTG-Jamendo-test | | | | | | |
| dortmund | 0.18 | 0.32 | 0.35 | 0.37 | 0.35 | **0.41** |
| gtzan | 0.11 | 0.35 | 0.37 | 0.40 | 0.37 | **0.44** |
| rosamerica | 0.37 | 0.43 | 0.44 | 0.46 | 0.44 | **0.48** |

**Table 3.2:** 5-fold cross-validation and cross-collection evaluation results for the genre tasks. The 5-fold cross-validation results are the mean and standard deviation of the balanced accuracy across the folds. The best results are marked in bold.

Table 3.2 contains the balanced accuracies obtained by each architecture and training strategy in both the 5-fold cross-validation and cross-collection evaluation on the MSD-test and MTG-Jamendo-test. These accuracies are computed by averaging the individual accuracy values obtained for each class. For the 5-fold cross-validation results, we additionally indicate the standard deviation of the balanced accuracies across folds. Our results show that the models trained from scratch are not superior to the SVM baseline. However, the transfer learning models outperform the SVM baseline for all the datasets and models, demonstrating the potential of features learned by audio tagging models. The Cross-collection evaluation results show that the SVM suffer a significant drop in performance when evaluated in the cross-validation setting, especially on the MTG-Jamendo-test dataset. Deep learning models, on the other hand, show less performance degradation, with the VGGish model showing the best results. Interestingly, the AudioSet model is not specifically trained for music content, but it is still capable of getting the best results, potentially due to its training data size and complexity. We hypothesize that having been trained on a larger dataset, despite not being music-specific, allows the model to generalize better to unseen data.

### 3.3.2 Evaluation of pre-trained models on music classification

We continue our research by extending our evaluation to additional pre-trained models. We considered all the downstream tasks presented in Section 3.2.5 and limit our evaluation to the 5-fold cross-validation setup.

Apart from the models presented in the previous experiment, we consider several CNNs trained for MIR tasks other than tagging with the goal of understanding if they can learn useful representations for music classification:

- **Tempo-CNN** is a family of models for tempo estimation introduced by Schreiber & Müller (2019). For the embeddings, we selected the DeepSquare model with $k = 16$ and used the logits of the last layer.

- **OpenL3** is a multi-modal self-supervised model trained to predict the correspondence between audio and video segments of YouTube videos trained by Cramer et al. (2019). For this work, we used the version trained on musical data on 128-bin mel-spectrograms, and the architecture with the embedding layer of 512 units. We chose this parametrization because it is the closest to other CNN models with a fixed number of mel-bends (MusiCNN: 96, VGGish: 64) and embedding dimensions (MusiCNN: 200, VGGish: 128). Additional informal testing revealed that the denser versions (256 mel bands, 6144 embedding dimensions) did not affect the performance significantly. We consider this model

| Model | RF (s) | Dims. | Params. | Data size | Appr. |
|---|---|---|---|---|---|
| MusiCNN | 3 | 200 | 787K | 220K | FS |
| MusiCNN-T200 | 3 | 200 | 787K | 350K | FS |
| VGG-Pons | 3 | 256 | 605K | 220K | FS |
| VGG-Pons-T200 | 3 | 256 | 605K | 350K | FS |
| VGGish | 1 | 128 | 62M | 70M | FS |
| Tempo-CNN | 12 | 256 | 1.2M | 11K | FS |
| OpenL3 | 1 | 512 | 4.7M | 296K | SS |
| Spleeter | 12 | 1280 | 49M | - | FS |

**Table 3.3:** Model embeddings. RF: Receptive field, Approach: fully-supervised (FS) or self-supervised (SS).

as a representative of unsupervised approaches with extensive usage for transfer learning (Grollmisch et al., 2021; Weck et al., 2021).

- **Spleeter** is a collection of source separation models using a separate U-Net architecture for each stem (Hennequin et al., 2020). We selected the 5-stem model and concatenated the bottleneck layers of the stems to create our embedding. We applied 4×4 max-pooling to reduce dimensionality.

Additionally, we trained two variants of the music auto-tagging models, VGG-Pons-T200 and MusiCNN-T200, using the 200 most frequent tags of MSD-Last.fm instead of just 50. Using more tags allowed for increasing the training size from 220K to 350K tracks.

We use an MLP with a single hidden layer of 100 units on top of the proposed embeddings. The model has a ReLU activation after the hidden layer and a Softmax or Sigmoid activation for multi-class and multi-label target tasks, respectively. We use the Adam optimizer with a learning rate of $1 \times 10^{-3}$ and weight decay of $1 \times 10^{-5}$. The model is optimized for a maximum of 30 epochs, but we only save the weights on the epochs achieving a historical minimum on the validation loss. This is commonly achieved between the first 10 to 20 iterations, depending on the task. Table 3.3 shows the models used in this experiment in terms of the receptive field, embedding layer dimension, number of parameters of the network, and amount of training data.

In preprocessing, we removed the problematic tracks in *gtzan* according to Sturm (2013). For each task, we trained a classifier on top of the different embeddings extracted by the models according to the transfer learning protocol described in Section 3.2.1. The architecture consists of an MLP with a single hidden layer with 100 neurons and ReLU activations. The output layer uses a Softmax activation to generate the output class probabilities.

| | SVM | MusiCNN | MusiCNN-T200 | VGG-Pons | VGG-Pons-T200 | VGGish | OpenL3 | Spleeter | Tempo-CNN |
|---|---|---|---|---|---|---|---|---|---|
| dortmund | 0.42 | **0.61** | 0.46 | 0.54 | 0.26 | 0.50 | 0.38 | 0.35 | 0.16 |
| gtzan | 0.77 | **0.86** | 0.79 | 0.83 | 0.46 | 0.84 | 0.58 | 0.57 | 0.26 |
| rosamerica | 0.86 | **0.94** | 0.90 | 0.93 | 0.66 | 0.93 | 0.84 | 0.70 | 0.46 |
| voice/instrum. | 0.93 | **0.98** | 0.93 | 0.97 | 0.78 | **0.98** | 0.89 | 0.76 | 0.58 |
| tonal/atonal | 0.98 | 0.87 | 0.91 | 0.92 | 0.78 | **0.93** | 0.89 | 0.89 | 0.70 |
| gender | **0.88** | 0.87 | 0.79 | 0.84 | 0.70 | 0.83 | 0.55 | 0.55 | 0.51 |
| danceability | 0.90 | **0.98** | 0.94 | 0.94 | 0.71 | 0.94 | 0.90 | 0.90 | 0.66 |
| acoustic | 0.93 | **0.96** | 0.93 | 0.93 | 0.83 | 0.93 | 0.89 | 0.89 | 0.75 |
| aggressive | 0.97 | 0.97 | 0.97 | **0.99** | 0.82 | **0.99** | 0.91 | 0.93 | 0.69 |
| electronic | 0.83 | 0.93 | 0.88 | 0.88 | 0.74 | **0.94** | 0.77 | 0.77 | 0.64 |
| happy | 0.81 | 0.86 | 0.77 | **0.89** | 0.69 | 0.86 | 0.76 | 0.70 | 0.68 |
| party | 0.88 | **0.92** | **0.92** | 0.64 | 0.84 | 0.90 | 0.77 | 0.87 | 0.73 |
| relaxed | 0.89 | 0.89 | 0.86 | **0.91** | 0.79 | 0.90 | 0.81 | 0.80 | 0.72 |
| sad | 0.88 | 0.87 | 0.88 | 0.86 | 0.83 | **0.89** | 0.85 | 0.83 | 0.84 |

**Table 3.4:** Class-weighted accuracies in the downstream tasks for the embeddings as an average of 5-fold cross-validation. The Top results are marked in bold.

Table 3.4 reports the results in both evaluations. The embeddings produced by the MusiCNN, MusiCNN-T200, VGG-Pons, and VGGish models achieved the best performance in different tasks. OpenL3, the only self-supervised model considered, obtained competitive results in *tonal/atonal* and *danceability*, but performed poorly in the rest of the tasks. The source separation model *Spleeter* achieved relatively competitive results in the *tonal/atonal*, *danceability*, *mood-aggresive*, and *mood-party* tasks, revealing that the embeddings produced by this model can be useful for some music classification tasks. The Tempo-CNN model, trained for tempo estimation, did not achieve competitive results in any of the tasks but *mood-sad* which correlates with the intuition that tempo may be highly correlated with specific musical moods. In general, the embeddings produced by the models trained for music auto-tagging tasks achieved the best results, suggesting that the semantic labels continue to be the most valuable pre-training tasks for music classification. The 200-tag models, MusiCNN-T200 and VGG-Pons-T200, did not outperform their 50-tag counterparts, suggesting that the additional tags and tracks did not provide useful information.

### 3.3.3 DiscogsEffNet a model based on Discogs' music styles

From the results of the previous experiments, we conclude that the features transferred from audio tagging models are beneficial for the creation of downstream classifiers for music understanding tasks. We find that using in-distribution data is important, and models trained on music-tagging tasks (e.g., MusiCNN) can perform on pair or better than models trained on a more general set of tags even when using much more parameters and data (e.g., VGGish). Following this, we are interested in developing a music representation model that can be used for a wide range of high-level music description tasks by exploiting large amounts of music data.

We collect metadata from the Discogs dump[23] as it was already hypothesized that it constitutes a detailed source of music metadata Bogdanov & Serra (2017); Hennequin et al. (2018). From the dump, we could map about 4 million tracks to our in-house audio collection. We consider the top 400 music style labels and keep only tracks annotated with one to five music styles. This results in a final dataset of 3.3 million tracks. We reserve 100,000 tracks each for testing and validation ensuring that each split has at least ten releases (e.g., album, EP, compilation...) of each style. To prevent artist leakage, we do not allow music from the same artist to appear in more than one split. Given the large size of this dataset, we store the audio as 16-bit mel-spectrograms with 96 mel bands. Because of this, we focus on common CNN architectures that have been successfully applied to two-dimensional representations before. We perform a grid search over architectures and hyper-parameters before training the final model:

- Architectures: EfficietNet B0 (Tan & Le, 2019), EfficientNet v2 S (Tan & Le, 2021), and ResNet50 (He et al., 2016)
- Learning rates: $1 \times 10^{-2}$, $1 \times 10^{-3}$, $1 \times 10^{-5}$
- Patch size durations: 2 seconds, 3 seconds

On Every epoch, the model receives a randomly selected patch from each track, which is equivalent to 76 days of music. We use the Adam optimizer and a scheduler halves the learning rate if the validation loss has not decreased in the last 10 epochs. We rely on 6 Titan 2080 Ti GPUs with a batch size of 256 each, resulting in an aggregated batch size of 1536 mel-spectrogram patches. The validation loss reached a plateau after completing 80 epochs in approximately 5 days.

The results reveal that the ResNet50 trained for 3 seconds achieves the lowest training loss. However, all the architectures and patch durations reach very

[23] http://data.discogs.com/

similar validation metrics for the learning rate of $1 \times 10^{-3}$. We hypothesize that the models cannot improve the validation metrics because of the noise in the Discogs labels, however, we are interested in keeping the experiments simple as the goal is to produce good music embeddings and not to fine-tune the style recognition task. Because of this, we select the much smaller EfficientNet (Tan & Le, 2019) on its B0 configuration trained on 2-second patches. For the rest of this chapter, we refer to this model as DiscogsEffNet or **EffNet (E)**.

### 3.3.4 Evaluation on small downstream tasks

In this experiment, we compare our EffNet with several pre-trained feature extractors. To maximize the relevance of the selection, we only consider models that have already shown competitive performance in music classification. These models must have an implementation and weights openly accessible to allow reproducibility without re-training. All of them share that they were designed to operate as embedding extractors, feature convolutional architectures, and use mel-spectrograms as input representation. We consider **MusiCNN (M)**, **VGGish (V)**, and **OpenL3 (O)** since they are the best-performing models from the previous experiments. Additionally, we include **Jukebox (J)** and **YAMNet (Y)**:

- **Jukebox (J)** is a music synthesis model conditioned on genre, artist name, and, optionally, lyrics (Dhariwal et al., 2020). Castellon et al. (2021) showed that the internal representations of this model are powerful embeddings for music classification. As the authors propose, we extract the average of the 36th layer of the model as embeddings.
- **YAMNet (Y)** is a MobileNet V1 architecture trained to predict 521. It was trained on the Youtube-AudioSet corpus, a curated version of the dataset used for VGGish. We extract representations from its penultimate layer containing 1024 dimensions. While VGGish and YAMNet could be expected to behave very similarly, this is the only case in which we test models trained in the same task. This also allows extracting conclusions about the impact of the architecture.

Table 3.5 compares the representation extractors in terms of the receptive field, number of embedding dimensions, number of parameters, training dataset size, and training approach.

In this experiment, we follow the 5-fold cross-validation methodology described in Section 3.2.2 for all the downstream tasks from Section 3.2.5 and the aforementioned representation extractors. As a baseline, we include a MusiCNN architecture trained from scratch on the downstream tasks. We chose this architecture because it has been found to perform well in small datasets (Won

| Model | RF (s) | Dimensions | Parameters | Data size | Approach |
|---|---|---|---|---|---|
| EffNet | 2 | 200 | 4.1M | 2.2M | FS |
| MusiCNN | 3 | 200 | 787K | 220K | FS |
| OpenL3 | 1 | 512 | 4.7M | 296K | SS |
| VGGish | 1 | 128 | 62M | 70M | FS |
| YAMNet | 1 | 1024 | 3.7M | 2M | FS |
| Jukebox | 24 | 4800 | 5B | 1.2M | CS |

**Table 3.5:** Embedding extractors. RF: Receptive field in seconds, Approach: fully supervised (FS), self-supervised (SS), or conditioned supervision (CS).

et al., 2020b). We follow the transfer learning protocol described in Section 3.2.1 with the parameters described in Section 3.3.2 for all models but Jukebox. In this case, we only extract a single timestamp (i.e., $1 \times D$) because it has a large receptive field of 24 seconds which makes it unstable for direct time-wise comparison with its CNN counterparts (with receptive fields ranging from one to three seconds). We consider that (from a human perspective) 24 seconds are generally enough to categorize audio for the proposed taxonomies, so it should be enough for the model to produce a competitive representation. For datasets with excerpts of less than 24 seconds (e.g., genre-dortmund) we repeat the audio to reach the required input size.

Since the considered datasets feature different degrees of difficulty and dispersion on the metrics, the accuracies cannot be directly compared (i.e., an improvement by 10% is a different achievement on different datasets). To account for this, we express the results as rankings # (i.e., scores 1 to $N$ where $N$ is the number of considered models and the model with the best accuracy gets a score of 1). In addition, we take advantage of the standard deviations collected on the 5-fold cross-validation to compute a ranking correction $\bar{\#}$. We assign the same ranking to the models that are not statistically different, using a t-test with a $p\text{-}value > 0.05$ criterion. This means that embedding extractors with equivalent performance will share the same *corrected rank*. In other words, the corrected rank of an embedding model is the rank of the best model that is not statistically different from it.

Table 3.6 reports the metrics obtained by the considered models. At least one of the embedding models surpassed or matched the baseline in all but the *happy* task. However, we do not see a clear advantage of the transfer learning paradigm in the 5-fold cross-validation setup compared with the baseline for many of the tasks. Despite this, the metrics outline some tendencies that are aligned with our expectations: provided that EffNet was trained in a massive corpus of music styles, it gets the best performance for all the genre tasks. Also, Jukebox excels in *gender*, *tonal/atonal*, and *voice/instrumental*, which are notions required to perform music and singing voice synthesis. On the

other hand, we find it surprising that EffNet is the best model for many mood-related tasks. While datasets such as MSD and AudioSet contain explicit mood labels (e.g., "angry music", " happy music"), EffNet was exclusively trained on music styles. We hypothesize that the relevance of the EffNet features could be due to the larger training dataset and the presence of genre bias in some mood tasks.[24] Our ranking analysis suggests that Jukebox is not significantly different from EffNet in the genre tasks, that EffNet, VGGish, and YAMNet are not different for the moods tasks, and that EffNet is not different from Jukebox for *gender*, *tonal/atonal*, and *voice/instrumental*. Overall, EffNet is not statistically different from the best-performing model for every task. Our interpretation is that, while the accuracies suggest which could be the best model for a given task, the corrected ranking tells in which cases more evaluation data is needed to confirm it.

[24] We observed that some mood classes are heavily genre-biased (e.g., most of the "aggressive" items in the *mood_aggressive* dataset are metal) so that they could be proxied as combinations of Discogs styles.

| | Baseline | EffNet | MusiCNN | OpenL3 | VGGish | YAMNet | Jukebox |
|---|---|---|---|---|---|---|---|
| | acc. | # $\bar{\#}$ acc. | # $\bar{\#}$ acc. | # $\bar{\#}$ acc. | # $\bar{\#}$ acc. | # $\bar{\#}$ acc. | # $\bar{\#}$ acc. |
| dortmund | 0.57 | **1 1 0.64** | 3 2 0.60 | 6 6 0.43 | 4 4 0.51 | 5 4 0.50 | 2 1 0.61 |
| rosamerica | **0.95** | **1 1 0.95** | 4 1 0.94 | 6 6 0.85 | 5 2 0.93 | 3 1 0.94 | 2 1 0.94 |
| gtzan | 0.89 | **1 1 0.92** | 3 3 0.86 | 6 6 0.74 | 4 4 0.84 | 5 4 0.83 | 2 1 0.91 |
| acoustic | 0.94 | 2 1 0.95 | 5 1 0.93 | 4 1 0.93 | 3 1 0.94 | **1 1 0.96** | 6 2 0.91 |
| aggressive | 0.98 | 3 1 0.98 | 5 1 0.97 | 6 1 0.96 | **1 1 0.99** | 2 1 0.98 | 4 1 0.97 |
| electronic | 0.91 | 3 1 0.93 | **2 1 0.93** | 6 2 0.86 | 1 1 0.93 | 4 1 0.91 | 5 1 0.89 |
| happy | **0.88** | 1 1 0.87 | 4 1 0.84 | 6 5 0.79 | 2 1 0.86 | 3 1 0.85 | 5 1 0.83 |
| party | 0.91 | **1 1 0.93** | 4 1 0.90 | 6 1 0.89 | 3 1 0.91 | 2 1 0.91 | 5 1 0.89 |
| relaxed | 0.90 | **1 1 0.91** | 4 1 0.89 | 6 2 0.87 | 3 1 0.89 | 2 1 0.90 | 5 3 0.87 |
| sad | **0.90** | **1 1 0.90** | 6 2 0.85 | 5 1 0.88 | 4 1 0.89 | 3 1 0.89 | 2 1 0.89 |
| danceability | 0.96 | **1 1 0.97** | 2 1 0.95 | 6 5 0.88 | 4 2 0.94 | 3 1 0.94 | 5 1 0.94 |
| gender | 0.86 | 2 1 0.86 | 3 2 0.85 | 6 5 0.79 | 4 2 0.84 | 5 5 0.80 | **1 1 0.88** |
| tonal/atonal | 0.94 | 3 1 0.96 | 4 1 0.95 | 6 4 0.91 | 5 1 0.93 | 2 1 0.96 | **1 1 0.97** |
| voice/inst. | **0.98** | 2 1 0.98 | 3 1 0.98 | 6 2 0.96 | 5 1 0.97 | 4 1 0.97 | **1 1 0.98** |

**Table 3.6:** Classification metrics per embedding extractor for all the downstream tasks. BL: baseline, #: rank, $\bar{\#}$: corrected rank, acc.: normalized accuracy.

### 3.3.5 Downstream dataset size analysis

With this experiment, we want to evaluate the performance of the embeddings under different amounts of training data. We considered $n \in \{1, 5, 10, 30\}$ samples per class and compared the results to those obtained using the entire downstream datasets. In the 5-fold cross-validation, we randomly choose $n$ samples per class from the training split, but we still use the whole testing split to compute the metrics. We observe that the selection of the training samples produced a large variability in the resulting accuracies, especially for the small values of $n$. To account for this, we repeat the whole process for three trials with a different random seed number to select the tracks. We compare five models in 14 target tasks using 5-fold cross-validation for three trials. Overall, this resulted in training more than 1,000 MLPs.

Figure 3.1 shows the normalized accuracies growing with respect to the amount of training data. We find that, for most of the tasks, an MLP based on convenient embeddings achieves most of the performance within the first 30 examples per class. *gender* is the only task that shows a clear improvement by using the whole dataset, probably because it is relatively difficult, and it has a larger availability of data. Additionally, when the embeddings are well aligned with the downstream task, we observe that the model can learn from fewer instances (e.g., EffNet in the genre tasks). On the other hand, models that are not very well aligned will benefit from seeing more data (e.g., OpenL3 in *tonal_atonal*). In contrast to the observations from the previous experiment, we find a huge performance gap compared to the baseline trained from scratch, especially on very small training sizes. Finally, we find two tasks: *mood_aggresive* and *mood_party*, which show very small improvement with the increase of data, which indicates that the classes are easily separable in all the considered embeddings spaces. The following experiments will help to identify if this is due to the easiness of the task or an oversimplification in the considered dataset.

### 3.3.6 Combination of embeddings

Embedding extractors are practical because they can generate low-dimensional representations, condensing information for new downstream tasks. Following this, extractors trained on different upstream tasks may distill different types of knowledge that can be complementary. A combination of multiple embeddings might therefore be beneficial for the downstream tasks.

To investigate this hypothesis, we stack embeddings to train our downstream models. Given a pair of embeddings generated by different extractors $x$ and $y$ corresponding to the same recording and sampled at an equal rate, a training

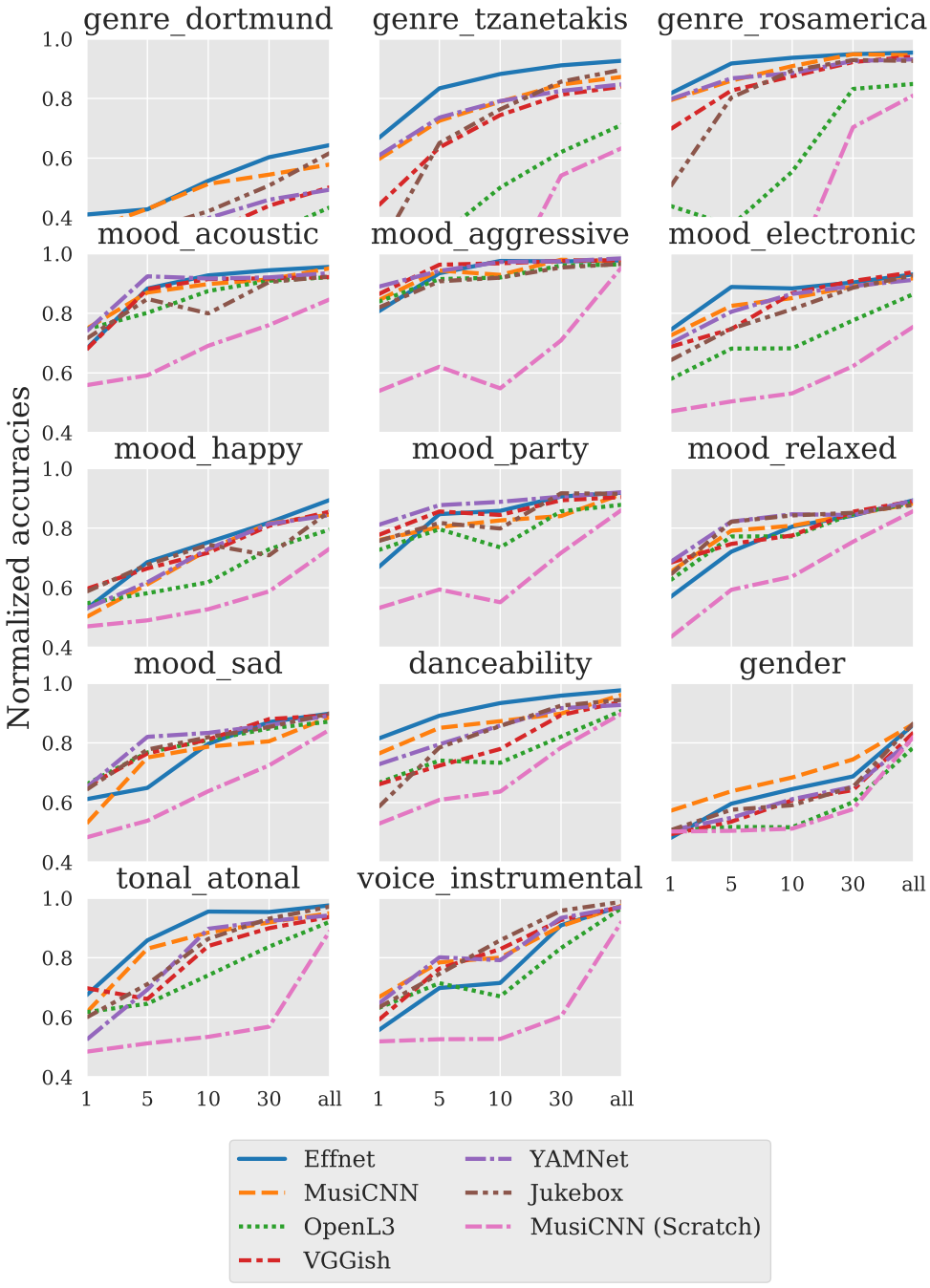


**Figure 3.1:** Evolution of the normalized accuracies with respect to the number of samples per class expressed as the average of three trials.

sample is formed by choosing a random time index $t$ and stacking $x_t$ and $y_t$[25] forming a one-dimensional representation of length $D_x + D_y$. We investigate combinations of two, three, four, five, and all the types of embeddings by training models for all the downstream tasks following the transfer learning protocol described in Section 3.2.1 and express the results as the distribution of normalized rankings per embedding type. We consider a total of 63 combinations, resulting in 4,410 MLPs. One of our principal concerns about the stacking method is that embeddings with different $D$ will have different opportunities to contribute to the downstream model. To mitigate this effect, we experiment with applying PCA to reduce the dimensionality of all the embedding types to a common value.

Figure 3.2 shows the metrics for all the combinations of embeddings in terms of corrected rankings ($\bar{\#}$). The rows are grouped by genre, mood, and other high-level tasks. The first six columns correspond to the embedding models alone (same as in the previous section), followed by all the combinations. In general, the results suggest that the combinations are not able to provide significantly better results, as in most cases, there is at least one single model with $\bar{\#}1$. However, we need to combine embeddings to achieve $\bar{\#}1$ for every task. Among those, two models combine three types of embeddings only: EffNet, MusiCNN, and VGGish (EMV), and EffNet, MusiCNN, and Jukebox (EMJ). It is interesting that models such as MusiCNN and VGGish are not among the most powerful when considered alone, yet they contribute to one of the best combinations. This suggests that providing complementary information is key to producing successful combinations. On the other hand, we notice that VGGish and YAMNet both obtained an average $\bar{\#}14$, but their combination (VY) was only able to ascend two positions ($\bar{\#}12$), suggesting that the additional dimensions are not beneficial if they contain redundant knowledge. We observe that combining features do not guarantee a performance improvement (e.g., EffNet plus OpenL3 performs worse than EffNet alone). Finally, we notice combining all the embeddings is just marginally better than EffNet or Jukebox, the best single models.

[25] for Jukebox we always take $t = 0$ since we do not have timestamps.

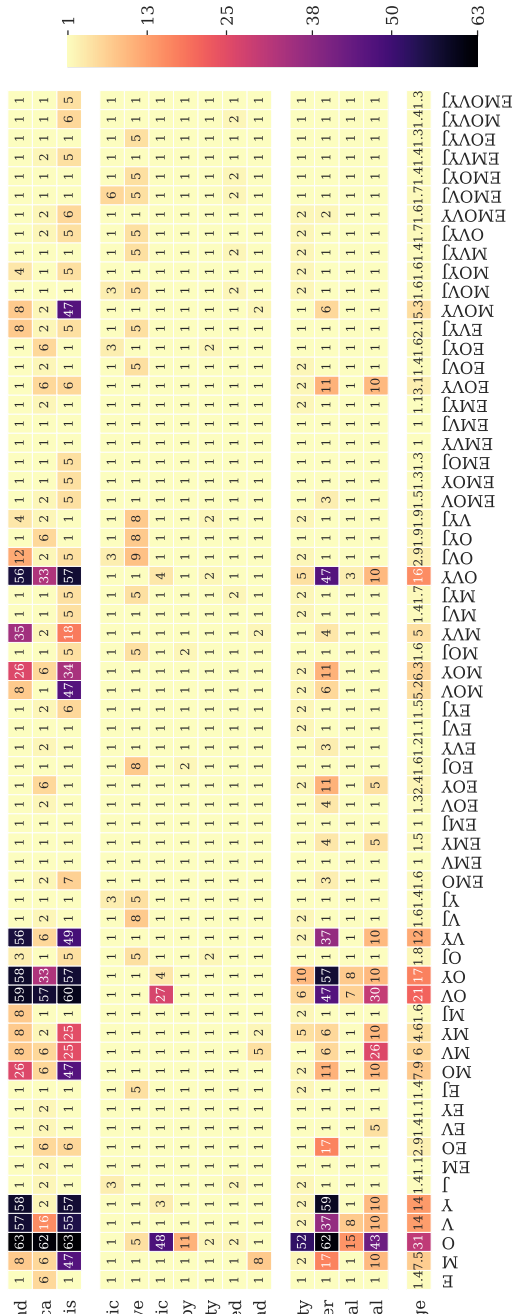


**Figure 3.2:** Corrected rankings ($\bar{\#}$) obtained by the embeddings and their combinations. A score of 1 means that the model is not statistically different to the model with the highest accuracy according to a t-test with a p-value of 0.05.

### 3.3.7 Cross-collection evaluation

Existing research suggests that music classification datasets are heavily influenced by the way they were collected and annotated (Bogdanov et al., 2016). To assess how the different embedding types affect the generalization capabilities of the models, we propose to further evaluate our downstream models in external data following the cross-collection evaluation methodology described in Section 3.2.3.

We use the *split-0* test subset of the MTG-Jamendo Dataset (Bogdanov et al., 2019b) containing 11,356 tracks as the source for our external validation. Since the original labels of this dataset do not have a one-to-one correspondence with the taxonomies of the downstream tasks, we developed an annotation tool to manually create a new set of labels. This tool requires the user to choose a class from the taxonomy of each task. For *genre* and *gender* tasks, an additional option "unmatched" was provided for tracks not fitting in any available class. To compensate for the subjectiveness of most of the downstream task classes, we decide to collect annotations by three different annotators for each track. We evaluate our downstream models only on tracks with a perfect inter-annotator agreement, not considering tracks labeled as "unmatched". As a result, the number of tracks used for this validation varies from task to task. We gathered and published ground-truth annotations from a total of 42 annotators for all mood and miscellaneous tasks and a subset of 1,000 tracks for the genre tasks.[26]

Table 3.7 shows the annotation agreement rates, number of annotations, the accuracies of the considered embeddings, and the best combination of embeddings per task. The agreement rates express the percentage of tracks in which the three annotators agreed on the label, and the number of annotations represents this subset after removing the "unmatched" tracks. These results estimate the generalization capabilities of the trained downstream models.

One of our first observations is that tasks with high agreement rates (e.g., *voice/instrumental*, *gender*) tend to correlate with less performance drop with respect to the 5-fold cross-validation results from Table 3.6. Due to the additional difficulties of the tasks, we could not collect many genre annotations, and the results for these tasks are less consistent. Opposing what we found in the 5-fold cross-validation setup, in the external validation, the models based on transfer learning are superior to the baseline by a large margin. This finding supports the idea that transfer learning enhances generalization when only small datasets are available. The advantage of EffNet for the genre tasks is only clearly visible in *rosamerica*, but it is close to other models in the other two tasks. EffNet is the most frequent embedding in the best combination,

[26] https://github.com/MTG/mtg-jamendo-dataset

suggesting that it has the most valuable information. Apart from this, the best combinations do not seem to follow a stable pattern. On average, we find that VGGish provides the best performance in external data for a small margin. We hypothesize that this is because it is the most compact representation, so it is less prone to overfitting. On the contrary, Jukebox (with 37 times more dimensions) is not performing very well despite its excellent results in the 5-fold cross-validation. On average, the best combination generalizes just slightly better than VGGish.

| Task | | | BL | Single models | | | | | | Combs. | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Agr. | Num. | M | E | M | O | V | Y | J | Acc. | Comb. |
| dortmund | 47% | 380 | 0.43 | 0.55 | 0.55 | 0.45 | 0.55 | **0.55** | 0.55 | 0.66 | EMO |
| rosamerica | 53% | 373 | 0.56 | **0.88** | 0.80 | 0.69 | 0.83 | 0.81 | 0.73 | 0.90 | EY |
| gtzan | 44% | 270 | 0.23 | 0.68 | **0.71** | 0.44 | 0.70 | 0.70 | 0.70 | 0.78 | EVY |
| acoustic | 60% | 6429 | 0.78 | **0.90** | 0.81 | 0.88 | 0.88 | 0.85 | 0.83 | 0.93 | EOV |
| aggressive | 71% | 7686 | 0.87 | 0.86 | **0.87** | 0.72 | 0.87 | 0.83 | 0.85 | 0.90 | MOVYJ |
| electronic | 66% | 7143 | 0.73 | 0.91 | 0.87 | 0.85 | 0.91 | **0.92** | 0.85 | 0.95 | EOY |
| happy | 53% | 5702 | 0.61 | 0.65 | 0.68 | 0.62 | **0.68** | 0.64 | 0.62 | 0.73 | EMOJ |
| party | 69% | 7476 | 0.81 | **0.93** | 0.91 | 0.87 | 0.92 | 0.91 | 0.85 | 1.00 | MOVJ |
| relaxed | 57% | 6163 | 0.77 | 0.80 | **0.83** | 0.75 | 0.83 | 0.78 | 0.78 | 0.86 | MVY |
| sad | 53% | 5678 | 0.69 | 0.72 | 0.71 | 0.65 | 0.72 | 0.69 | **0.76** | 0.90 | EMOJ |
| danceability | 44% | 4476 | 0.83 | 0.74 | 0.78 | **0.83** | 0.82 | 0.76 | 0.79 | 0.89 | EOJ |
| gender | 87% | 2070 | 0.78 | 0.81 | 0.85 | 0.71 | 0.81 | 0.76 | **0.87** | 0.89 | EMO |
| tonal/atonal | 75% | 7533 | 0.51 | 0.58 | 0.66 | 0.59 | **0.70** | 0.59 | 0.60 | 0.70 | V |
| voice/inst. | 87% | 8756 | 0.83 | 0.89 | 0.83 | 0.92 | **0.92** | 0.91 | 0.91 | 0.95 | MOYJ |
| average | - | - | 0.67 | 0.78 | 0.78 | 0.71 | 0.80 | 0.76 | 0.76 | 0.80 | EMOV |

**Table 3.7:** Normalized accuracies (the best model in each task marked in bold) in the external validation. BL: Baseline, Num.: number of tracks used as external ground truth, Agr.: agreement rate.

| Model | GTZAN | MTG-Jamendo | |
|---|---|---|---|
| | Acc. | PR-AUC | Mediaeval |
| SampleCNN (Lee et al., 2018) | 0.821 | - | - |
| lileonardo (Bour, 2021) | - | 0.1509 | 1 |
| EffNet | **0.848** | **0.1352** | 7 |
| MusiCNN | 0.819 | 0.1172 | 11 |
| OpenL3 | 0.588 | 0.1164 | 11 |
| VGGish | 0.786 | 0.1351 | 7 |
| YAMNet | 0.828 | **0.1352** | 7 |
| Jukebox | 0.800 | 0.0880 | 16 |
| EOVY | 0.898 | 0.13932 | 6 |
| EMOY | 0.795 | 0.14414 | 4 |

**Table 3.8:** Comparison of the SOTA models (up) with the embedding extractors (middle) and the best combinations for each task (down) in GTZAN and the moods and themes subset of the MTG-Jamendo dataset.

### 3.3.8 Additional benchmarks on public datasets

Since most of the considered downstream datasets are in-house, we select two additional datasets with well-documented results in the literature to evaluate the performance of the models using standardized fixed splits with results extensively reported in the literature. Our goal is to link our classifiers to the SOTA to contextualize their performance.

- **GTZAN**. We already include GTZAN in our experiments in previous sections, discarding the problematic files as suggested by Sturm (2013). For compatibility with other studies, we additionally re-ran our evaluation using the data splits proposed by Kereliuk et al. (2015).

- **The MTG-Jamendo Mood and Theme split**. is a multi-label classification task based on a subset of the MTG-Jamendo Dataset with 18,486 full-length audio tracks and a predefined data split for training, validation, and testing (corresponding to the dataset *split-0* fold). For this experiment, we reduced the learning rate to $5 \times 10^{-4}$, which slightly enhanced the performance. Additionally, since this is a multi-label classification task, we use the PR-AUC metric to evaluate the models.

Table 3.8 contains the metrics obtained by the SOTA models, the proposed embeddings, and the best combinations of embeddings for each task. For the MediaEval task, we also show the position the models would have achieved

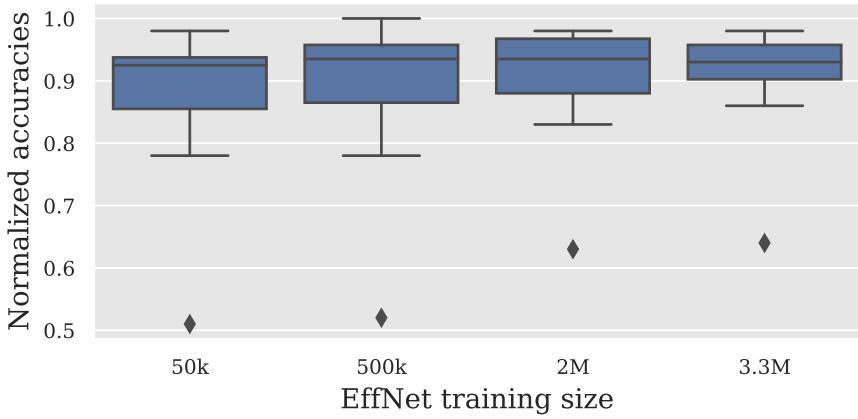


**Figure 3.3:** Distribution of normalized accuracies obtained by classifiers trained on embeddings extracted from the EffNet model trained with different amounts of data.

in the classification ranking of the 2021 challenge.[27] From this comparison, we conclude that our best downstream models are on par with the SOTA. Besides, specific combinations of embeddings outperform them. We also find that some combinations do not improve the individual models' performance, or even decrease it. All our downstream models and embedding combinations outperform the VGGish baseline proposed by the challenge organizers. Using specific embedding combinations, we achieve good results that rank just below the top-performing submissions. While the latter potentially uses highly optimized methods for this task, our downstream models are generic and demonstrate how embeddings enable successful systems.

### 3.3.9 Effect of the pre-training dataset size

One of the axes of research for this work hypothesizes that using more annotated data improves music embedding models. Figure 3.3 shows the variation in performance in the downstream tasks for the amount of training data in EffNet. For every model, the plot shows the mean, quartiles, and the 95% confidence interval of the normalized accuracies for all the considered downstream tasks. The version trained on 3.3 million tracks is the one used in the previous experiments. The smaller versions of EffNet were created in preliminary experiments during the development of the dataset.

These results show how increasing the amount of training data consistently improves the overall downstream metrics. While the improvements in the last

[27] https://multimediaeval.github.io/2020-Emotion-and-Theme-Recognition-in-Music-Task/results

iterations (2 million version 3.3 million) are not significative, we could not find a clear saturation point at the experimented scales. This suggests that the models could benefit from even more data.

## 3.4 Conclusions

In this chapter, we have explored the usage of audio models trained on labels to extract representations for music classification tasks. We started by evaluating existing pre-trained models, which led us to the conclusion that using large collections of annotated music would be the best approach to developing novel models. After this, we created a new model, EffNet, using a large dataset of music annotated by the public domain metadata available in the Discogs database. Note that while the Discogs' dump contains other types of valuable information such as artists or record label relations, release year, or country, we decided to simply target the music styles, leaving the exploration of other metadata for the next chapters.

We performed a systematic of our model in the context of other SOTA pre-trained architectures used as feature extractors. First, we conducted a 5-fold cross-validation revealing that EffNet is not statistically different from the best SOTA model in every task. However, we found that the smaller datasets are not appropriate to measure the differences between splits due to the additional dispersion among folds. The dataset size experiment shows that, as expected, the models based on embeddings achieve good performance with significantly fewer examples than the baseline model trained from scratch. Additionally, we observe that less data is required to achieve good performance when the source and downstream tasks are more closely aligned (e.g., music style classification and genre recognition). In the experiment combining different embeddings types, we do not find significant improvements on top of original embeddings for particular tasks, but we find specific combinations that show good performance in more tasks. In the evaluations on external data, we find that all the embedding-based systems improve the generalization on top of the baseline trained from scratch. This opposes the results observed in 5-fold cross-validation and aligns with other works recommending transfer learning approaches for contexts of few data. Still, there is a performance gap (huge for some tasks) demonstrating the need for more work in musical classification generalization. Interestingly, VGGish is the model with the best metrics in the external validation, while it was not among the best embeddings in the 5-fold cross-validation. We hypothesize that this is because it is the embedding with fewer dimensions, and the MLPs have fewer chances to overfit the downstream datasets. Then, we find competitive results in two benchmarked datasets, GTZAN, and the MediaEval Mood and Theme Recognition Challenge, show-

ing that our models are on par or close with the SOTA. Our interpretation is that results from datasets with fixed splits, such as in the benchmarked tasks we considered, are prone to highlighting minimal improvements that, when seen from the perspective of the statistically corrected metrics or the external validation, turn out to be not very significant. Thus, we want to highlight the importance of such studies to understand the true meaning of the metrics. Finally, we observe that the downstream performance is correlated with the amount of data used to train the embedding models, suggesting that more data is beneficial to developing more robust representations.

CHAPTER 4

# Representation learning based on editorial metadata

## 4.1 Introduction

From the results of Chapter 3, we conclude that supervised approaches are, for now, more compute-efficient than their unsupervised counterparts, and we will continue to explore them. However, obtaining large amounts of labeled music audio data is expensive and time-consuming. In Section 1.2.2, we identified that one interesting characteristic of music is that it tends to be naturally distributed together with metadata such as the artist, album, or genre names. We refer to this information as associative metadata since it allows for establishing potential similarity relationships between different tracks. For example, two tracks are more likely to be similar if they were composed by the same artist.

Following Section 1.2.2, we divide associative metadata into two types: editorial and consumption. While editorial metadata contains information about the music itself, such as the artist, album, or genre, consumption metadata is related to how the music is consumed, for example, the number of plays, likes, or the playlists in which a track was included. In this chapter, we focus on training music representation models based on editorial metadata available from Discogs. The study of consumption metadata is addressed in Chapter 5.

One of the most appealing characteristics of editorial metadata is its ubiquitous nature, which makes it much more scalable than relying on consistent taxonomies requiring professional annotators. Basic editorial metadata, such as the artist or album name, is usually embedded in the audio files themselves in the form of standardized containers such as ID3[28] or Vorbis comment.[29]

[28] https://en.wikipedia.org/wiki/ID3
[29] https://xiph.org/vorbis/doc/v-comment.html

Additionally, online databases such as Discogs,[30] MusicBrainz,[31] or Last.fm[32] provide more detailed information such as the release year, country, record label, or genres/styles of each release. The metadata of most of these platforms are publicly available under permissive licenses, which motivates leveraging this information as a source of supervision to train music representation models. Notably, metadata can contain inconsistencies, and the process of matching it to audio may introduce noise compromising the quality of the resulting dataset.

Park et al. (2018) proposed one of the first representation learning models based on editorial metadata by using artist names as a training target, which resulted in features close in performance to those obtained from systems trained on crowdsourced tags of the MSD. Since the artist information is too sparse to be learned efficiently in a classification setup, they approached the task using metric learning, by creating triplets with anchor and positive samples of tracks belonging to the same artist and a negative sample from a track belonging to a different one. In a posterior study, Lee et al. (2019b) extended this approach to learning track-, album-, and artist-level associations, finding that a combination of all provided the best representation models.

In this chapter, we propose creating representation learning models focused on music tagging using editorial metadata as an inexpensive source of supervision by extending the work by Park et al. (2018) and Lee et al. (2019b) with three key contributions. First, we rely on the much larger Discogs20 dataset containing 3.3 million tracks, which is an order of magnitude larger than the datasets used in previous works. Second, we adopt a contrastive learning approach already performant in a self-supervised setup instead of siamese networks. Specifically, we choose COLA Saeed et al. (2021) for its simplicity, operating directly on spectral representations, and requiring no data augmentation, which makes it efficient and suitable for large-data regimes. COLA works by maximizing the bilinear similarity between a pair of mel-spectrogram patches (anchor and positive) cropped from the same audio clip while minimizing it for the rest of the patches in the batch. We propose to modify this self-supervised approach by constructing the anchor/positive pairs according to the different types of metadata considered (i.e., same track, release, artist, and record label). Additionally, we explore whether different metadata associations generate complementary information by combining the embeddings produced by their respective models and creating multitask systems jointly optimized to learn them. Finally, we develop a strategy to disentangle the metadata notions to isolate their effect on the learned representations, an aspect that was never considered before.

[30] https://www.discogs.com/
[31] https://musicbrainz.org/
[32] https://www.last.fm/

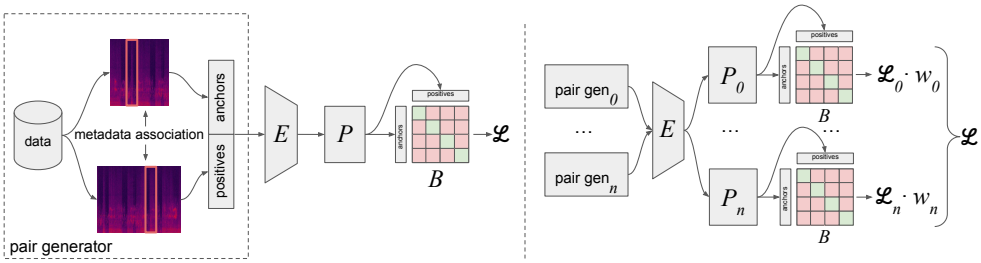


**Figure 4.1:** Overview of the proposed pipelines for a single (left) and multiple tasks (right). $E$ is the encoder, $P$ the projection head, $B$ are the weights for the bilinear similarity, and $\mathcal{L}$ is the loss term. In the multitask setup, tasks go from *0* to *n*.

The experiments of this chapter are based on Alonso-Jiménez et al. (2022).

## 4.2 Method

We are interested in assessing the representation power of features derived from associations of commonly available editorial metadata. This goal is motivated by previous works that already showed the usability of the track, artist, and album similarities as supervision targets Lee et al. (2019b). As these works already studied the influence of the dataset size and showed a positive correlation with the downstream metrics, we will only perform full-size experiments and focus on unexplored aspects.

In particular, we are interested in imposing tight constraints on the pair generation process to emphasize the differences between the different associations. To formalize our problem, we consider a collection of music where each song has a track ID ($t$) and appears in one or more releases ($R$). Each release is produced by one or more artists ($A$) for one or more record labels ($L$). We define one self-supervised and three metadata-based associations:

- *track association*, the anchor and positive samples come from the same track (self-supervised),
$$t_a = t_p$$
- *release association*, $t_a$ and $t_p$ have at least a release in common,
$$|R(t_a) \cap R(t_p)| > 0$$
- *artist association*, $t_a$ and $t_p$ have at least an artist in common, but they do not appear in the same release,
$$|A(t_a) \cap A(t_p)| > 0, \text{ and } |R(t_a) \cap R(t_p)| = 0$$

- *label association*, $t_a$ and $t_p$ have at least a record label in common, but they do not share any artist,

$$|L(t_a) \cap L(t_p)| > 0, \text{ and } |A(t_a) \cap A(t_p)| = 0$$

Note that this approach considerably limits the number of pairs that can be matched and presumably leads to suboptimal representations. However, our goal is to limit the number of overlapping pairs to emphasize the differences between associations. For instance, the release associations would be broadly a subset of the artist ones without the proposed constraints.

For each association, we generate a dataset of anchor/positive pairs. We initialize a pool with all the songs in the music collection. For each song, we pick a random pair that complies with the association's condition and remove both from the pool. While this approach does not exploit every possible association, it gives each track one association attempt, which provides us with sufficient training data for the scope of this work. We also considered a balanced version of the algorithm (e.g., the same number of tracks per artist) that we discarded as the performance dropped due to the reduced dataset size without providing additional insights.

Following our contrastive paradigm, we do not need to create explicit anchor/negative pairs. For a given anchor, the rest of the positive samples in the batch are considered negatives (see Figure 4.1). While this naive approach implies that two pairs of tracks associated with the same artist will be respectively used as negative examples, there is a small probability of having repeated artists in a batch, and we ignored this issue.[33]

## 4.3 Experiments

We pre-trained different contrastive systems following the proposed similarity notions. To evaluate the quality of the learned representations, we trained shallow classifiers on different multi-class and multi-label classification tasks. Additionally, we conducted experiments to understand the complementarity of such representations.

We follow a straightforward implementation of COLA, but instead of relying on the same audio clip (purely self-supervised), we form the anchor/positive pairs according to relationships from the metadata. The complete pipeline is represented in Figure 4.1 (left). This is, to the best of our knowledge, the first usage of this framework in the context of MIR.

[33]For example, considering a dataset of one million tracks, a batch size of 200 pairs, and an average of 6 tracks per release, the probability of having two pairs from the same release in the batch is $6e-4$, which we consider an affordable false-negative rate.

| Association | Pairs | Diversity | Time | Acc. |
|---|---|---|---|---|
| track | 3.3M | 3.3M | 63h | 88.7 |
| release | 846K | 2.0M | 21h | 35.7 |
| artist | 1.2M | 257K | 33h | 41.1 |
| label | 1.1M | 142K | 48h | 24.7 |

**Table 4.1:** Statistics for the metadata association and their respective models. We show the number of pairs used, association diversity (number of different tracks, releases, artists, and record labels, respectively), training time (hours), and validation accuracy (%).

### 4.3.1 Contrastive targets based on Discogs' metadata

The Discogs database provides publicly available dumps of their *release* data that we used to create our training targets. According to Discogs, a release is "a broad term for any audio product that is made for general public consumption"; albums, singles, or compilations are examples of releases. Each release entry contains lists of the artists and record labels involved, the year and country of release, and a list of music genres and styles according to the Discogs' taxonomy. We matched our in-house audio collection to releases and master releases (groupings of different versions of a release) in the Discogs metadata dump resulting in 4 million tracks, with 58.2% of the tracks belonging to more than one release. A track may be linked to many releases for multiple reasons, including remasters, reeditions, or compilations containing it. For each track, we generated lists of artists and record labels by pooling the metadata on the releases linked to it. We observed that different versions of a release could contain very different information, so for us, this was a simple way of maximizing the amount of information available per track.

We selected the subset with the top 400 most popular music styles resulting in 3.3 million tracks to train a baseline model with a multi-label classification objective (Style tags model). We reserved subsets of 50,000 tracks without artist overlap and a minimum frequency of 50 releases per music style tag for testing and validation. These sets were used as data pools to generate training, validation, and testing sets for the considered metadata associations following the methodology presented in Section 4.2. Note that we did not apply any data cleaning or deduplication, meaning that there may be room for improving the proposed representation models. Table 4.1 contains the resulting number of pairs and association diversities for each association, as well as the training time, and the accuracy obtained by their respective models.[34]

[34]The high number of releases is partially due to albums with multiple reeditions. On average, each track in our dataset is linked to 5 releases.

### 4.3.2 Pre-training models with editorial metadata

We used an EfficientNet v1 on its B0 configuration as a backbone to learn the embedding representations Tan & Le (2019). Our models operate on 2-second patches of mel-spectrograms with 96 bands extracted with the same parametrization as for MusiCNN (Pons & Serra, 2019a) using the Essentia library.[35]

Due to the massive dataset size, we stored the features as half-precision (16-bit) floats and split the data into three machines with two GeForce RTX 2080 Ti GPUs each to parallelize the training. For every epoch, we fed the model with a random 2-second crop of the mel-spectrogram from each track in the training set. We also used a random patch per track for validation, but in this case, we used the same offset on every epoch to get more stable metrics. We relied on the Adam optimizer with an initial learning rate of $1 \times 10^{-3}$ and a scheduler that reduced the learning rate by a factor of 10 if the validation loss had not decreased in 10 epochs. We trained the models for 100 epochs and considered two versions. The first one had the weights from the epoch where the lowest validation loss was achieved. The second model was obtained with stochastic weight averaging (Izmailov et al., 2018). For the last 25 epochs, we imposed a learning rate of $1 \times 10^{-3}$ and kept a moving average of the weights. As the latter version only reported minimum improvements in specific tasks, we decided to present the results of the former only.

As a baseline, we trained a model targeting the top 400 Discogs music styles with the multi-label soft-margin loss by connecting a fully connected layer to the flattened output of the last convolutional block with 1280 units (embedding layer). As discussed on the Discogs website,[36] music styles usually go beyond purely stylistic descriptions and encode cultural, temporal, or geographical information, so we hypothesize that the learned representations are valuable beyond the task of genre recognition.

For the contrastive objectives, we used a fully connected projection head with 512 dimensions on top of the same embedding layer, followed by a normalization layer and a *tanh* activation. We used the bilinear similarity ($a^T Bp$) and the cross-entropy loss as in the original implementation. While the authors of COLA showed that larger batch sizes improved the performance, we could only afford a batch size of 200 pairs due to the memory size of our GPUs. We parallelized the training so that each optimizer step aggregated six batches computed by different GPUs. This setup was close to the optimal number of pairs per optimizer step found in the original publication but used a larger ratio of positive samples.

[35] https://essentia.upf.edu/reference/std_TensorflowInputMusiCNN.html
[36] https://blog.discogs.com/en/genres-and-styles

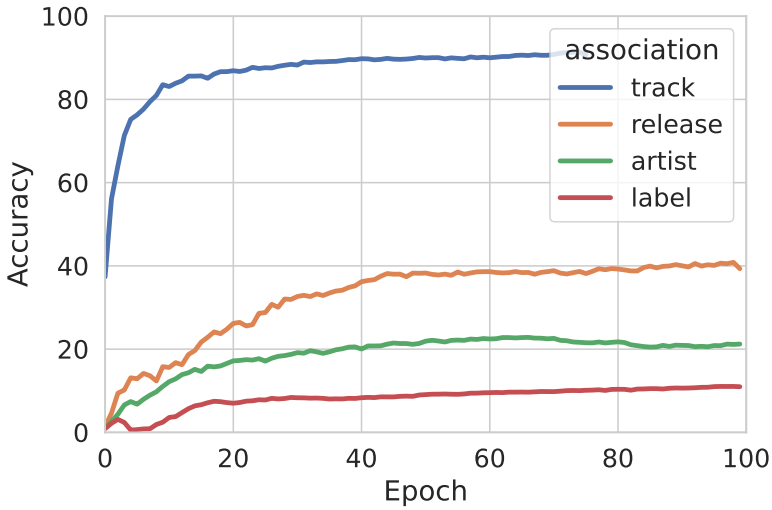


**Figure 4.2:** Training accuracies over the epochs for the different associations.

To get additional insights into the models, we computed the top-1 accuracies by taking the *arg max* of the similarity matrices. Table 4.1 shows the training times and validation accuracies obtained by the model of each metadata association, and Figure 4.2 shows the evolution of the training accuracies over the epochs. The accuracies show that the associations have different difficulty levels, which aligns with our expectations.

All the pre-trained models are publicly available for feature extraction within the Essentia library.[37]

### 4.3.3 Downstream datasets

We evaluated our models as frozen feature extractors following a transfer learning setup. We consider three well-known music auto-tagging and classification datasets: FMA-small (Defferrard et al., 2016) (*FMA*), MagnaTagATune (Law et al., 2009) (*MTAT*), and the MTG-Jamendo Dataset (Bogdanov et al., 2019b). The latter contains different subsets for *Genre*, *Instrument*, and mood/theme (*Mood*) tags, as well as the top-50 tags in the dataset (*Top50*) that we treated independently. For *FMA* and *MTAT*, we used the splits proposed by the authors (Defferrard et al., 2016) and the 12:1:3 partition (Van Den Oord et al., 2014) respectively. In the MTG-Jamendo tasks, we used the sets defined by its *split-0* similarly to previous works (Won et al., 2020b; Manco et al., 2021). Table 4.2 shows the size of the considered datasets.

[37] https://essentia.upf.edu/models.html

| Dataset | Size | Classes | Type |
|---|---|---|---|
| Genre | 55,215 | 87 | Multi-label |
| Instrument | 25,135 | 40 | Multi-label |
| Mood | 18,486 | 56 | Multi-label |
| Top50 | 54,380 | 50 | Multi-label |
| MTAT | 25,860 | 50 | Multi-label |
| FMA | 8,000 | 8 | Multi-class |

**Table 4.2:** Considered downstream datasets.

### 4.3.4 Transfer learning evaluation setup

As for the pre-training stage, we trained the downstream models with 2-second patches randomly cropped on each epoch. For validation, we averaged over the activations from the half-overlapped patches of the entire tracks. We passed the patches through the frozen backbone and used the flattened output of the last convolution layer as the input to a multi-layer perceptron with a single hidden layer of 512 units with a *sigmoid* or *softmax* activation for the multi-label or multi-class tasks. We used the cross-entropy loss and the Adam optimizer with a starting learning rate of $1 \times 10^{-3}$ and added a weight decay of $1 \times 10^{-5}$. A scheduler divided the learning rate by half if the loss had not decreased in five epochs. We trained the models for 30 epochs and used the weights from the epoch achieving the lowest validation loss to evaluate the test set.

### 4.3.5 Stacks of embeddings

Apart from evaluating the embeddings obtained from every association individually, we were interested in understanding their complementarity. We wanted to understand if the individually learned representations could be combined to boost the performance, and if so, which were the best combinations. To investigate this, we ran stacks of embeddings through the presented evaluation protocol. First, we evaluated the stack of the Track, Release, Artist, and Label features for all the datasets. Additionally, we performed a systematic evaluation considering all the possible combinations of the four contrastive models plus the model trained on tags on *MTAT* considering two, three, four, and five embeddings.

### 4.3.6 Multitask model

We also considered training a multitask model to learn the metadata associations jointly. The architecture of the proposed system is depicted in Figure 4.1 (right). For each association, we have a separate pair generator and projection

head. To perform an optimization step, we ran a batch of pairs from each association through the shared encoder and its specific projection head and computed a weighted sum of the losses. The loss weights were empirically selected prioritizing associations with better single-model performances (*track*: 0.1, *release*: 0.15, *artist*: 0.6, and *label*: 0.15). Additionally, we initialized the encoder with the weights of the model based on artist associations on its 20th epoch to speed up the training. While we experimented with multitask models based on two association types, we did not find additional insights and decided to omit those results. Due to the additional model size, we had to reduce the batch size to 50 pairs per association.

## 4.4 Results and discussion

Table 4.3 reports the metrics obtained by our models and selected works from the literature. In descending order, the five groups in the table contain SOTA models trained from scratch without additional data, SOTA embedding models, baseline embedding models trained by us, our proposed embedding models trained on metadata associations, and models combining metadata associations. In the first group, we include MusiCNN (Pons & Serra, 2019a), Harmonic CNN (Won et al., 2020b), and the winning submission of the 2021 Emotion and Theme Recognition challenge (team Lileonardo) (Tovstogan et al., 2021)[38] as the best models from the literature in *MTAT*, *Top50*, and *Mood*, respectively. Similarly, MuLaP (Manco et al., 2021) reports the best performance as a frozen embedding extractor in *Genre*, *Mood*, *Top50*, and *FMA*, and the same applies to CALM Castellon et al. (2021) in *MTAT*. Note that comparisons against these reported metrics may be unreliable due to differences in training and evaluation settings.

We computed three baselines based on audio embeddings from an EfficientNet architecture with random weights, an EfficientNet architecture trained on music style tags as described in Section 4.3.2, and the VGGish model with its pre-trained weights (Hershey et al., 2017).

Concerning our contrastive models, we observe that the model based on track associations (Track model) achieves competitive performance in some tasks, especially in *Top50*. Nevertheless, the models using metadata associations show better or equivalent performance despite seeing fewer pairs of tracks in the pre-training stage. In particular, we find that the model based on artist associations (Artist model) is the best-performing with a few exceptions, which aligns with previous studies in metadata-based music representation learning (Lee et al., 2019b).

[38] https://multimediaeval.github.io/2021-Emotion-and-Theme-Recognition-in-Music-Task/

*Instrument* and *MTAT* are the tasks where our models are further from the SOTA. For *MTAT*, we attribute this to the fact that CALM has 1,000 times more parameters and that the authors report the best metrics from a grid search over shallow classifiers and hyperparameters. Also, we observed that models operating in the full *MTAT* previews tend to report higher PR-AUC performances (Castellon et al., 2021; Zhao et al., 2022) than models operating in short chunks (Pons & Serra, 2019a; Spijkervet & Burgoyne, 2021).

| | Genre | | Instrument | | Mood | | Top50 | | MTAT | | FMA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | ROC | PR | ROC | PR | ROC | PR | ROC | PR | ROC | PR | Acc. |
| Lileonardo | - | - | - | - | **77.5** | **15.1** | - | - | - | - | - |
| Harmonic CNN | - | - | - | - | - | - | 83.2 | 29.8 | *91.3 | *45.9 | - |
| MusiCNN | - | - | - | - | - | - | - | - | 90.7 | 38.4 | - |
| MuLaP | 85.9 | - | 76.8 | - | 76.1 | - | 82.8 | - | *89.3 | *40.2 | **61.1** |
| CALM | - | - | - | - | - | - | - | - | **91.5** | **41.4** | - |
| Random weights | 50.7 | 3.1 | 49.9 | 6.4 | 50.4 | 3.4 | 48.3 | 6.5 | 50.0 | 5.3 | 12.5 |
| Style tags | 87.7 | 19.9 | 77.6 | 19.8 | 75.6 | 13.6 | 83.1 | 29.7 | 90.2 | 37.4 | 59.1 |
| VGGish | 86.3 | 17.2 | **77.8** | **20.2** | 76.3 | 14.1 | 83.2 | 28.2 | 90.2 | 37.2 | 53.0 |
| Track associations | 86.3 | 18.0 | 69.9 | 16.7 | 74.0 | 12.8 | 82.9 | 29.4 | 89.7 | 36.4 | 58.9 |
| Release associations | 86.9 | 18.9 | 71.9 | 17.2 | 72.8 | 11.7 | 83.2 | 29.8 | 90.3 | 37.1 | 60.9 |
| Artist associations | **87.7** | **20.3** | 69.7 | 16.9 | 76.3 | 14.3 | **83.6** | **30.6** | 90.7 | 38.0 | 59.1 |
| Label associations | 87.0 | 19.4 | 75.0 | 18.2 | 74.8 | 12.8 | 83.1 | 29.9 | 88.7 | 34.2 | 59.5 |
| Stack | 86.9 | 19.4 | 74.7 | 18.8 | 74.3 | 13.0 | 83.4 | 30.0 | 90.8 | 38.6 | 59.8 |
| multitask | 87.2 | 19.9 | 70.5 | 17.2 | 76.1 | 14.4 | 83.5 | 30.3 | 90.8 | 37.8 | 60.0 |

**Table 4.3:** ROC-AUC, PR-AUC, and accuracy metrics for the downstream datasets. The five horizontal groups represent SOTA models from the literature trained from scratch, SOTA feature extractors from the literature, baseline feature extractors trained by ourselves, the proposed feature extractors based on metadata associations, and the proposed feature extractors combining associations. (*) results were computed in a clean version of *MTAT* and are not directly comparable.

| Tr | Re | Ar | La | St | ROC-AUC | PR-AUC |
|---|---|---|---|---|---|---|
| ✓ | | ✓ | ✓ | ✓ | 90.93 | 38.75 |
| ✓ | ✓ | ✓ | | ✓ | 90.92 | 38.69 |
| | ✓ | ✓ | ✓ | ✓ | 90.92 | 38.51 |
| | ✓ | ✓ | | ✓ | 90.89 | 38.57 |
| | | ✓ | ✓ | ✓ | 90.87 | 38.57 |

**Table 4.4:** Top-5 combinations of the **Tr**ack, **Re**lease, **Ar**tist, **La**bel and **St**yle Tags features in *MTAT*. The results are sorted according to the ROC-AUC values.

The Stack is obtained by concatenating the embeddings from the models based on single associations as input for the MLPs. Except for *MTAT*, we do not get improvements over the best single model in any dataset despite the additional input dimensionality (5,120). Table 4.4 shows the top-5 results of models trained on combinations of embeddings for *MTAT*. The representations from the Artist and Style tags models are the only ones present in all the top 5 combinations, suggesting that these are the representations with the most valuable information. This observation aligns with the results from Table 4.3.

Similarly, our multitask model is not superior to the best single model in any dataset, but generally, it is close in performance to the Artist model. We observe that multitask only overcomes Artist in the datasets where other associations perform better (i.e., *Instrument* and *FMA*), which shows its capability to pick the best information from different association types.

## 4.5 Conclusions

In this chapter, we studied the usage of editorial metadata as a source of supervision to create contrastive representation learning models intended for music tagging. We demonstrated that it is possible to rely on this very inexpensive source of supervision to learn competitive representations. In Chapter 3.3.3, we propose DiscogsEffNet, a model featuring the same architecture trained to predict 400 music style labels from Discogs. Our model trained on artist associations achieves better performance in 5 out of the 6 datasets considered, showing the potential of metadata-based representations for music classification tasks.

Our approach extends previous works on metadata-based representation learning by scaling up the models in terms of training dataset size, considering additional editorial metadata notions, and experimenting with a new contrastive learning setup. We could validate that some of the observations from previous studies still hold for models trained on 10 times more data. Namely, we observed that some metadata-based representations are superior to their

tag-based counterparts and that artist associations provide the best representations. Additionally, we found that the features learned from different metadata notions can be combined or jointly learned, showing slight performance improvements for particular tasks. To the best of our knowledge, this was the first study using Discogs' metadata to train music representation models. We did this by generating pairs of tracks according to metadata notions with simple matching rules, which leaves room for experimentation with the dataset.

CHAPTER 5

# Representation learning based on consumption metadata

## 5.1 Introduction

In Chapter 4, we explored the use of editorial metadata to pre-train models for music tagging tasks, showing that the use of artist co-occurrences as positive pairs can lead to better representations than those obtained using style labels as classification targets. In this chapter, we extend this approach by exploring the use of consumption metadata for the same purpose.

As discussed in Section 1.2.2, we identify two main categories of music metadata: editorial and consumption. While the former typically refers to information specific about the music pieces (e.g., artist and album names, or country and year of release), the latter describes interactions of humans (or machines) with them (e.g., playlists, DJ setlists, radio programs, or listening histories). Using consumption metadata as a source of similarity ground truth has already been explored in the recommender-systems literature, enabling tasks such as music playlist continuation (Chen et al., 2018). However, the usage of consumption metadata for music representation learning has not been as extensively investigated yet (Huang et al., 2020; Ferraro et al., 2021) as the case of editorial metadata (Park et al., 2018; Kim et al., 2018a; Lee et al., 2019b; Kim et al., 2020b; Huang et al., 2020; Alonso-Jiménez et al., 2020b), thus leaving room for further exploration.

Consumption metadata contains information on human judgments of music that fit in the same context and thus may be more suitable for learning representations that encode perceived similarity. Also, while editorial relations are normally one- or few-to-many (e.g., album-songs), consumption is many-to-many (e.g., playlists-songs), resulting in a potentially more dense co-occurrence space that may favor associating more heterogeneous music.

In general, music metadata of diverse types is easily accessible at industrial contexts such as streaming services, or radio stations, and can be combined by crossing data from different sources. Specifically, in this chapter, we focus on music consumption metadata in the form of music playlists. However, our method could be easily extended to other types of many-to-many consumption data, such as radio programs, DJ setlists, or listening histories.

In this chapter, we propose strategies to pre-train models using music playlist data. We start by comparing the performance of three models based on playlist data and four baselines using two different architectures in one similarity and five classification tasks. Then, we find that the proposed pre-training strategies using playlist information lead to superior performance compared to previous approaches based on editorial metadata in several music classification tasks. Additionally, we show that some models trained with playlists achieve better similarity metrics than those based on self-supervision or editorial metadata. Finally, we propose a method that leverages training-time similarity estimations to optimize the process of selecting positive pairs from the playlists showing benefits in the tagging and similarity tasks.

The main contributions of this chapter were developed during a 6-month internship in Utopia Music, [39] and the experimental results are based on Alonso-Jiménez et al. (2023a). Additionally, the algorithm proposed in Section 5.5 has been registered on the patent "Training methodology for contrastive learning based on associative metadata."

## 5.2 Method

We investigate methods to obtain targets from music playlist datasets to pre-train models using contrastive learning. Instead of using augmentation techniques to obtain the anchor and positive samples, as done in standard self-supervised contrastive approaches, we propose using pairs originating from different tracks by exploiting playlist information. This idea is motivated by the assumption that songs co-existing in a playlist tend to share a certain degree of similarity.

The number of possible pairs of elements that co-occur in a playlist of size $n$ corresponds to the number of combinations without repetition $\binom{n}{2}$. This produces many pairs when considering millions of playlists, making the exhaustive usage of the pairs difficult to scale with this contrastive learning approach. Because of this, we propose algorithms that rely on heuristics to create audio pairs that utilize a wide range of tracks while preventing track repetitions, as well as an embedding learning-based technique to create the target pairs.

[39] http://utopiamusic.com

Considering a dataset of *playlists* $P = \{p_0, ..., p_N\}$, and *tracks* $S = \{s_0, ..., s_M\}$, we propose the following strategies:

- *Co-Occurrence.* This approach randomly generates pairs by producing combinations using the available tracks in each playlist $p_i$ and with each track appearing in only one pair. We iterate randomly through $P$ generating $\lfloor \frac{|p_i|}{2} \rfloor$ pairs per playlist and discarding the associated tracks from the set of available tracks. This algorithm is executed at the beginning of each training epoch.

- *Top Co-Occurrence.* This algorithm counts the number of co-occurrences of the tracks in all the playlists. For each anchor track, we randomly select its associated pair among its top 10 most co-occurring tracks while ensuring that every track appears only in one pair. To do so, at each epoch, we initialize a set of available tracks $A = S$. We randomly iterate through $A$ and for a given track $s_j$ we select one of the top co-occurring tracks $s_k$ and discard $s_j$ and $s_k$ from $A$.

- *Word2Vec representation.* This is a multi-modal approach in which, for a given track, we align the projection of its audio representation $z'_x$ to the projection of its *Word2Vec* embedding $z'_y$ (Mikolov et al., 2013). We train a *Word2Vec* model by considering playlists as sentences and track IDs as words. We rely on the Continuous Bag of Words approach with a context window that includes the entire playlist [40] and a learning rate of 0.02 for 20 epochs.[41] In this case $B_y(\cdot)$ is the frozen pre-trained *Word2Vec* model, $z_y$ is a *Word2Vec* embedding, and $H_y(\cdot)$ is a different projector from $H_x(\cdot)$ featuring the same hyper-parameters and dimensions.

## 5.3 Experiments

Following the proposed methodology, we pre-train music representation models using contrastive learning with positive pairs sampled by (i) randomly sampling tracks co-occurring in playlists, (ii) constraining the positive pairs to the top co-occurrences across playlists, and (iii) using alternative track representations obtained using a Word2Vec (Mikolov et al., 2013) model trained on the playlist sequences as associated pair. We consider the ResNet50 (He et al., 2016) and VGGish (Hershey et al., 2017) architectures and use the playlist data from the MPD (Chen et al., 2018). After pre-training, the models are fine-tuned and

[40] We also tested a W2V sensitive to the track positions in the playlists by using smaller window sizes. However, this degraded the performance.

[41] We use the Gensim implementation https://radimrehurek.com/gensim/models/word2vec.html

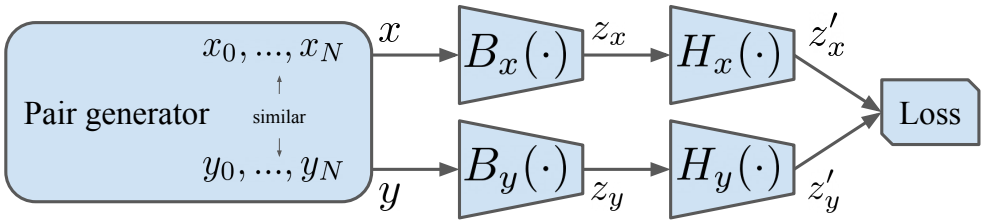


**Figure 5.1:** Illustration of our pre-training pipeline. The features $x$ and $y$ from the associated pairs are input to the model $B(\cdot)$ and projector $H(\cdot)$. $B(\cdot)$ and $H(\cdot)$ are optimized using a contrastive loss. $B(\cdot)=B_x(\cdot)=B_y(\cdot)$ and $H(\cdot)=H_x(\cdot)=H_y(\cdot)$ in all the cases except for *Word2Vec representation*.

evaluated in the downstream music tagging or directly evaluated in a music similarity task.

### 5.3.1 Contrastive learning setup

Our architecture consists of a convolutional backbone $B(\cdot)$ and a projector $H(\cdot)$ that map a mel-spectrogram input $x \in \mathbb{R}^{T \times F}$ with $T$ timestamps and $F$ frequencies into latent representations $z \in \mathbb{R}^D$, and $z' \in \mathbb{R}^{D'}$ respectively. The model is trained to bring $z'_x$ close to $z'_y$ while pulling it apart from samples in the same batch following SimCLR (Chen et al., 2020). After pre-training, $H(\cdot)$ is discarded, and $B(\cdot)$ is used in the downstream tasks. Our setup is depicted in Figure 5.1.

### 5.3.2 Architectures

We repeated all experiments with two standard backbone architectures. Additionally, we use an auxiliary projection head $H(\cdot)$ to map the latent representations to a joint embedding space during the contrastive learning phase.

- **VGGish** (Hershey et al., 2017). This is a variant of the VGG (Simonyan & Zisserman, 2014b) architecture popular in the audio domain. It has 128 output dimensions. We consider the original model weights obtained from a classification task in a proprietary dataset as a baseline.[42] When pre-training the architecture with our data, we use our 3-second 96-band mel-spectrogram patches and modify the kernel of the first pooling layer from $2 \times 2$ to $4 \times 4$ to keep the number of dimensions after the convolutional layers close to the one in the original model.
- **ResNet50** (He et al., 2016). We use the standard ResNet50 model considering its good performance in audio and music applications (Wang

[42] https://github.com/tensorflow/models/tree/master/research/audioset/vggish

et al., 2022). We reduce the output of the last dense layer with global max- and mean-pooling and concatenate the resulting vectors, leading to an output embedding of 4,096 dimensions.

- **Projector**. The same projector is used for the two backbone architectures and every contrastive learning setup. It consists of an MLP model with a single hidden layer of 128 dimensions and ReLu activation and 128 output dimensions.

### 5.3.3 Pre-training models with consumption metadata

We pre-train several models following the proposed pair generation strategies. *SimCLR* is a baseline where $x$ and $y$ are alternative views of the same audio patch mixed with random patches from the batch scaled with a gain factor sampled from a $\beta(5,2)$ distribution similar to previous work (Wang et al., 2022). *Artist CO* is another baseline that applies the *Co-Occurrence* strategy to the artist names (i.e., considering the set of tracks by each artist as a playlist) without preventing track repetitions.

*Playlist CO*, *Playlist TCO*, and *Playlist W2V* use the *Co-Occurrence*, *Top Co-Occurrence*, and *Word2Vec representation* strategies respectively to generate the $x/y$ pairs using the playlist information. Table 5.1 shows the number of pairs per epoch and model. In *SimCLR* and *Playlist W2V*, the number of pairs corresponds to the number of tracks in the dataset since these methods do not associate different tracks. In *Playlist CO* and *Playlist TCO*, we constrain to a single track occurrence per epoch, which results in fewer pairs per epoch. We train the models for a fixed number of 50 epochs. This makes the pair generation algorithms execute the same number of times, which leads to a different number of batch optimization steps for each model.

We pre-train all our models using the Million Playlist Dataset (*MPD*) (Chen et al., 2018) matched to our in-house music collection, resulting in 1,779,072 tracks and 999,219 playlists. [43] $H(\cdot)$ has a single hidden layer with 128 units and a ReLu activation and $D' = 128$. We use the NT-Xent loss (Chen et al., 2020) with a fixed $\tau$ value of 0.1 using a batch size of 384 pairs, and the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate is increased linearly from 0 to $1 \times 10^{-4}$ for the first 5,000 steps and then decreased following a cosine decay until the models complete 50 epochs similar to Wang et al. (2022). We train the models on 96-band, 256-timestamp (∼3 seconds) mel-spectrogram patches randomly selected at each iteration from the 30-second excerpts available for each track.

[43] Our dataset has an average number of tracks per artist and playlist of 7.2 and 66.3, respectively. On average, a track appears on 29.3 playlists and belongs to 1.28 artists.

| Model | Pairs per epoch | Pair generation algorithm |
|---|---|---|
| SimCLR | 1,779,072 | - |
| Artist CO | 1,014,528* | *Co-Occurrence* |
| Playlist CO | 826,368* | *Co-Occurrence* |
| Playlist TCO | 731,520* | *Top Co-Occurrence* |
| Playlist W2V | 1,779,072 | *Word2Vec representation* |

**Table 5.1:** Number of pairs per epoch and pair generation algorithms. *indicates that these are different pairs on each epoch.

### 5.3.4 Downstream datasets

Table 5.2 contains the datasets used in our experiments, including the number of samples, and their purpose.

We use the Genre, Instrument, and Mood subsets of the MTG-Jamendo Dataset (Bogdanov et al., 2019b) and the MagnaTagATune (MTAT) dataset (Law et al., 2009) considering its top-50 tags and using the standard 12:1:3 assignment of the train/val/test partition proposed by Van Den Oord et al. (2014). Additionally, we consider an in-house genre dataset containing 2-minute excerpts and 72 classes, referred to as Genre Internal. With this dataset, our goal is to assess if our pre-training approaches are still beneficial when a bigger and arguably more curated collection is available.

For the music similarity evaluation, we use the dim-sim dataset consisting of a collection of music similarity triplets produced by human raters (Lee et al., 2020a). Each triplet was annotated by 5 to 12 people, and the official clean version of the dataset contains 879 triplets with a high inter-annotator agreement (superior to 0.9).

### 5.3.5 Music tagging

Our first evaluation consists of solving multi-label music classification tasks by fine-tuning the pre-trained models. We keep the pre-trained $B(\cdot)$ and replace $H(\cdot)$ with an MLP with the same hidden layer configuration and output dimensions matching the number of classes followed by a Sigmoid activation. We optimize $B(\cdot)$ and the new $H(\cdot)$ using Adam ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) and cross-entropy loss with an L2 regularization term of $1 \times 10^{-5}$ for a maximum of 50 epochs. We use a cyclical triangular scheduler that varies the learning rate from $1 \times 10^{-5}$ to $1 \times 10^{-4}$ (Smith, 2017). The weights are selected from the epoch with the highest Average Precision on the validation set. We apply early stopping after ten epochs without any improvement on this metric. In training, we use the same random patch selection approach as in pre-training. During inference, we average the activations from non-overlapping patches.

| Dataset | Size | Purpose |
|---|---|---|
| MagnaTagATune | 25,860 | tagging (50 classes) |
| MTG-Jamendo Genere | 55,215 | tagging (87 classes) |
| MTG-Jamendo Instrument | 25,135 | tagging (40 classes) |
| MTG-Jamendo Mood&Theme | 18,485 | tagging (56 classes) |
| Genre Internal | 87,542 | tagging (73 classes) |
| dim-sim | 879 | similarity (triplets) |

**Table 5.2:** Datasets used in the downstream evaluation.

We use 30 seconds of audio from the center of the track in validation, and the full duration available in testing.

### 5.3.6 Music similarity

We extract representations $z$ for the clean subset of dim-sim using the pre-trained models without fine-tuning. Following the common evaluation approach used by (Lee et al., 2020a), we measure the cosine distance between anchor/positive, and anchor/negative, and consider the triplet prediction correct if the latter is larger. We report the prediction accuracy and the average difference between anchor/negative and anchor/positive distances.

## 5.4 Results and discussion

Table 5.3 shows the macro ROC-AUC and Average Precision[44] metrics for all the datasets and models as the average ± the standard deviation of three runs. On each trial, a different random seed is used for the initialization of the MLP classifier, the selection of offsets for the mel-spectrogram patches, and the order in which tracks are presented to the model. Our baselines consist of fine-tuning the original VGGish (Hershey et al., 2017) (*VGGish FT*), randomly initialized models (*From Scratch*), *SimCLR*, and *Artist CO*. The proposed models based on metadata are *Artist ICO*, *Playlist CO*, *Playlist TCO*, and *PlaylistW2V*.

We note that the contrastive approaches based on artist and playlist metadata always achieve better performance than the *VGGish FT*, *From Scratch*, and *SimCLR* baselines, which aligns with previous works indicating the benefits of metadata-based supervision (Park et al., 2018; Kim et al., 2018a; Lee et al., 2019b; Alonso-Jiménez et al., 2022). The models based on playlist information achieve equivalent or superior performance to those based on *Artist CO* on

[44] Average Precision is also referred to as the area under the precision-recall curve (PR-AUC) in the literature.

most datasets and metrics, and *Playlist W2V* with the ResNet50 architecture achieves the best performance in at least one metric for the Genre, Instrument, Mood, and Genre Internal datasets.

| Dataset | Genre | | Instrument | | Mood | | MTAT | | Genre Internal | |
|---|---|---|---|---|---|---|---|---|---|---|
| | AP | ROC | AP | ROC | AP | ROC | AP | ROC | AP | ROC |
| | | | | | VGGish | | | | | |
| *VGGish FT* | 15.8±0.3 | 84.9±0.5 | 18.1±0.6 | 74.2±1.2 | 12.1±0.9 | 72.7±0.8 | 44.4±0.6 | 90.6±0.1 | - | - |
| *From Scratch* | 13.4±0.2 | 82.6±0.3 | 15.5±0.4 | 72.1±0.5 | 9.3±0.2 | 70.6±0.4 | 40.2±0.7 | 88.9±0.2 | 54.3±0.3 | 96.7±0.0 |
| *SimCLR* | 15.2±0.3 | 83.6±0.4 | 16.4±0.3 | 72.1±0.5 | 10.7±0.2 | 70.1±0.2 | 41.1±0.6 | 88.9±0.2 | 61.7±0.1 | 97.6±0.0 |
| *Artist CO* | 17.3±0.1 | 85.6±0.1 | 20.4±0.4 | 76.7±0.1 | 13.9±0.2 | 74.5±0.5 | 46.2±0.1 | 91.1±0.1 | 68.8±0.2 | 98.3±0.1 |
| *Playlist CO* | 17.0±0.1 | 85.4±0.2 | 20.2±0.4 | 76.1±0.3 | 13.3±0.8 | 73.8±0.8 | 45.9±0.2 | 90.9±0.0 | 67.7±0.7 | 98.2±0.1 |
| *Playlist TCO* | 17.5±0.1 | 84.9±0.4 | 20.5±0.3 | 76.3±0.9 | 13.8±0.3 | 73.7±0.7 | 45.8±0.3 | 91.0±0.1 | 70.0±0.4 | 98.4±0.0 |
| *Playlist W2V* | 17.3±0.2 | 85.5±0.3 | 19.8±0.7 | 75.3±0.2 | 13.5±0.1 | 72.8±0.7 | 45.3±0.6 | 90.9±0.2 | 69.8±0.1 | 98.4±0.0 |
| | | | | | Resnet50 | | | | | |
| *From Scratch* | 14.4±0.2 | 82.9±0.1 | 15.6±0.5 | 71.2±0.6 | 8.9±0.0 | 69.2±0.4 | 40.7±0.3 | 88.8±0.1 | 63.3±0.1 | 97.7±0.1 |
| *SimCLR* | 16.3±0.2 | 84.7±0.3 | 17.4±0.1 | 73.3±0.7 | 12.1±0.3 | 73.0±0.3 | 43.4±0.5 | 90.1±0.2 | 67.4±0.3 | 98.2±0.0 |
| *Artist CO* | **19.0±0.1** | 85.0±0.1 | 21.1±0.4 | 76.5±0.9 | 14.9±0.3 | 74.8±0.7 | 47.0±0.3 | **91.5±0.2** | 73.4±0.2 | 98.6±0.1 |
| *Playlist CO* | 18.7±0.7 | **85.7±0.4** | **21.2±0.** | 76.7±0.9 | 14.8±0.5 | 74.2±0.4 | 46.8±0.2 | 91.4±0.0 | 73.4±0.1 | 98.6±0.0 |
| *Playlist TCO* | 18.9±0.2 | 85.1±0.3 | 20.4±0.7 | 75.4±1.5 | 14.3±0.4 | 73.7±0.7 | **47.0±0.** | 91.3±0.2 | 72.8±0.2 | 98.6±0.0 |
| *Playlist W2V* | **19.0±0.** | 85.4±0.3 | 20.7±0.4 | **77.1±0.4** | **15.0±0.** | **75.1±0.4** | 46.7±0.4 | 91.2±0.2 | **74.1±0.** | **98.7±0.1** |

**Table 5.3:** Metrics in the music classification datasets expressed in macro ROC-AUC and Average Precision. For each architecture, we present the baselines on top and the proposed models below. Metrics statistically equivalent or higher than *Artist CO* according to a one-sided t-test ($p\text{-}value = 0.005$) are marked in light grey. The highest metric per dataset is marked in bold.

| Model | VGGish | Resnet50 | VGGish | Resnet50 |
|---|---|---|---|---|
| | Accuracy | | Average difference | |
| *SimCLR* | 0.699 | 0.672 | 0.007 | 0.009 |
| *Artist CO* | 0.819 | 0.838 | 0.043 | 0.039 |
| *Playlist CO* | **0.852** | **0.845** | **0.077** | **0.064** |
| *Playlist TCO* | 0.793 | 0.813 | 0.052 | 0.046 |
| *Playlist W2V* | 0.831 | 0.818 | 0.067 | 0.041 |

**Table 5.4:** Music similarity accuracy and the average difference between anchor/negative and anchor/positive.

Table 5.4 contains the results of the music similarity evaluation in the dim-sim dataset. We observe that models based on metadata show a stronger correlation with human similarity perception than the baseline SimCLR approach. While *Playlist CO* achieves the best metrics with both architectures, *Playlist TCO* and *Playlist W2V* did not improve the performance as in the classification tasks. We hypothesize that *Top Co-Occurrence* and *Word2Vec representation* reduce the diversity of the positive pairs, which may augment the discriminative capabilities of the latent space at the cost of becoming weaker for similarity. Finally, we observe that pair selection strategies as *Top Co-Occurrence* that were found beneficial in classification, decrease the similarity performance compared to the basic *Playlist CO*, which suggests that augmenting the discriminative capabilities of the latent space may have adverse effects for similarity.

Finally, these results may depend on the nature and sparsity of the available playlists. In our study, we relied on MPD, which contains a curated subset of Spotify playlists filtered by quality and enriched with additional tracks. The playlists were created by US users only between 2010 and 2017 and are not expected to be representative of the overall distribution of playlists in Spotify according to the authors. However, this dataset represents a small fraction of more than 4 billion playlists reportedly existing in Spotify by 2018, which motivates further research on playlist-based pre-training.

## 5.5 Improving the selection of positive pairs

While methods based on playlist information showed promising results, we acknowledge certain limitations in the pair selection strategies. Particularly, it was easy to find examples of playlists with tracks that are not related in terms of genre, mood, or instrumentation. For example, In MPD we could find playlists containing tracks played on films or TV shows, or tracks created in a particular region or country, without any other apparent relationship.

Following this idea, we developed an algorithm called *Informed Co-Occurrence* (*ICO*) that operates jointly with the model being trained to improve the relevance of the selected pairs. This algorithm is a variation of *Co-Occurrence* that operates jointly with the model $B(\cdot)$ being trained to improve the relevance of the selected pairs.

At the beginning of each epoch, the algorithm iterates $P$ randomly dividing each playlist $p_i \in P$ into two halves,

$$p_i = p_{ia} \cup p_{ib} \quad \text{with} \quad p_{ia} \cap p_{ib} = \emptyset \quad \text{and} \quad L = |p_{ia}| = |p_{ib}| = \frac{|p|}{2}. \tag{5.1}$$

For each $p_i$, we compute the similarity among latent representations of the tracks in $p_{ia}$, and $p_{ib}$,

$$S_{j,k} = m(p_{ia,j}, p_{ib,k}). \tag{5.2}$$

Then, for each anchor track $p_{ia,j}$ we create a list of *candidates* $c_{ia,k}$ consisting on the tracks in $p_{ib}$ sorted by similarity $S_{i,0:L}$ in descending order. Our goal is to select a positive sample from the list of candidates that is likely to be similar to the anchor, but without allowing the model to always choose the same pair. We solve this by assigning each candidate a weight $w$ based on a Gaussian density function with $\mu = 0$ and $\sigma = \frac{3}{4}L$.

$$w(k) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{k^2}{2\sigma^2}\right) \tag{5.3}$$

Additionally, we discard candidates that have already been assigned to previous values of $j$. Note that candidates in the first positions (with higher estimated similarity) will have higher weights, making them more likely to be selected. After this, the anchor and positive tracks are added to a list of used samples, so that they are no longer used as candidates in the following playlists. Finally, the pair selection algorithm is run at the beginning of each epoch, allowing the updated versions of $B(\cdot)$ to provide more accurate estimations each time, progressively improving the quality of the selected pairs. Overall, our algorithm favors matching similar pairs within the playlists, while allowing the model to learn from a variety of examples thanks to the initial random split of the playlist and weighted sampling of the candidates.

### 5.5.1 Modifications to the model

The *ICO* algorithm can be easily integrated into the proposed pipeline, by computing similarities on top of the latent representations $z$ or $z'$, e.g., the

| Model | VGGish | | Resnet50 | |
|---|---|---|---|---|
| | *Playlist CO* | Playlist ICO | *Playlist CO* | Playlist ICO |
| | | Genre | | |
| mAP | 17.0±0.1 | 15.3±0.3 | **18.7±0.7** | 18.1±0.2 |
| ROC | 85.4±0.2 | 84.3±0.4 | **85.7±0.4** | 84.6±0.6 |
| | | Instrument | | |
| mAP | 20.2±0.4 | 17.5±0.4 | **21.2±0.7** | 19.3±0.9 |
| ROC | **76.1±0.3** | 73.8±0.7 | 76.7±0.9 | 75.5±1.0 |
| | | Mood | | |
| mAP | **13.3±0.8** | 11.1±0.5 | 14.8±0.5 | 14.2±0.1 |
| ROC | 73.8±0.8 | 72.1±0.8 | **74.2±0.4** | 74.4±0.8 |
| | | MTAT | | |
| mAP | 45.9±0.2 | 43.8±0.7 | **46.8±0.2** | 45.5±0.2 |
| ROC | 90.9±0.0 | 90.3±0.2 | **91.4±0.0** | 91.0±0.1 |
| | | Genre Internal | | |
| mAP | 67.7±0.7 | 59.7±1.8 | **73.4±0.1** | 70.7±0.1 |
| ROC | 98.2±0.1 | 97.4±0.2 | **98.6±0.0** | 98.4±0.0 |

**Table 5.5:** Macro ROC-AUC (ROC) and mean Average Precision (mAP) metrics in the music tagging datasets. The highest metric per dataset marked in bold.

cosine similarity, $S_{j,k} = \frac{\mathbf{z_j} \cdot \mathbf{z_k}}{\|\mathbf{z_j}\| \|\mathbf{z_k}\|}$. In this sense, it is possible to apply the *ICO* algorithm to any of the models we have proposed.

Additionally, we propose modifying the model's architecture by replacing the MLP projector $H(\cdot)$ with a bilinear similarity layer that learns a similarity function between the latent representations of the anchor and candidate, $H(\mathbf{z_{a_j}}, \mathbf{z_{c_k}}) = \mathbf{z_{a_j}}^T \mathbf{W} \mathbf{z_{c_k}}$. Similar to Saeed et al. (2021), we replace the NT-Xent loss with standard cross-entropy applied on top of the similarity scores produced by the new projector. This modification attends practical reasons since having a projector that produces similarity values instead of latent representations allows for a faster execution compared to having to compute these values in a separate step. This aspect was particularly relevant when dealing with a dataset of millions of tracks and playlists, as in our case.

### 5.5.2 Results and conclusions

We evaluated the *ICO* algorithm on the tasks of music tagging and similarity using the same datasets and evaluation protocols as the previous models. Table 5.5 shows the results for the music tagging datasets, and Table shows the results for the music similarity task. We observe that the *ICO* algorithm

| Model | VGGish | Resnet50 | VGGish | Resnet50 |
|---|---|---|---|---|
| | Accuracy | | Average difference | |
| Playlist ICO | 0.852 | 0.845 | 0.77 | **0.064** |
| *Playlist ICO* | **0.857** | **0.857** | **0.112** | 0.054 |

**Table 5.6:** Music similarity accuracy and average difference for *ICO* compared to our previous best model.

achieves lower performance in the music tagging datasets, and better performance in the music similarity task compared to the *Playlist CO* model.

One limitation of our evaluation is that, due to time constraints, we could not conduct further experiments to understand how much the model modifications proposed in Section 5.5 impacted the performance of the *ICO* algorithm. Our results support existing literature suggesting that the standard SimCLR paradigm is superior to the bilinear similarity plus cross-entropy loss for discriminative tasks (Wang et al., 2022). However, the more positive results in the similarity task suggest that models should not be jointly optimized for similarity and discriminative use cases, as optimizing for one task may harm the performance of the other.

## 5.6 Conclusions

In this chapter, we have shown that employing contrastive learning for pretraining neural networks with playlist information is a valuable source of supervision for music tagging. Also, the learned representations achieve a performance comparable to the state of the art for music similarity using a source of supervision that does not require explicit human annotation.

While Chapter 4 focused on editorial metadata, such as the artist name, we found that superior performance can be achieved with consumption metadata consisting of playlist information by relying on track representations obtained from a Word2Vec model trained on the playlist sequences. Also, the representations learned using simple playlist co-occurrences perform significantly better than an unsupervised approach (SimCLR).

Regarding the music similarity task, we found that the proposed models based on playlist information achieve better performance than the SimCLR baseline or than using artist co-occurrences for music similarity. Additionally, we proposed an algorithm called *ICO* that improves the selection of positive pairs by considering the similarity between tracks in the same playlist.

Notably, the main contributions of this chapter were developed as part of an industrial internship at Utopia Music, where the author was able to lever-

age the company's resources and expertise. As main contributions, the *ICO* algorithm was protected by a patent application filed by Utopia Music, and the results presented in Section 5.4 were presented at the 2023 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) in Alonso-Jiménez et al. (2023a).

CHAPTER 6

# Representation learning with transformers and patchout

## 6.1 Introduction

In Chapters 4 and 5 we relied on standard CNN architectures and focused on the impact of the training approach on the downstream tasks. In this chapter, we fix our pre-training task to multi-label classification using Discogs20' music styles and explore how certain design choices can benefit the downstream performance when using transformer architectures.

The universal success of transformers in text (Vaswani et al., 2017), vision (Dosovitskiy et al., 2021), and audio (Gong et al., 2021) tasks motivate further research using this architecture for music representation learning. However, most SOTA models were based CNNs by the time we started this research (Huang et al., 2020; McCallum et al., 2022; Alonso-Jiménez et al., 2022; Huang et al., 2022b). We hypothesized that transformers were not ruling this domain yet because they require large amounts of data and computational power to overcome their convolutional counterparts, while such resources are not always available. To address these challenges, we propose leveraging Discogs20, which contains 3.3M tracks annotated with public-domain metadata from Discogs, and using techniques to train transformers efficiently. Specifically, we focus on PaSST, by Koutini et al. (2022), a method that has demonstrated remarkable performance in the AudioSet (Gemmeke et al., 2017) benchmark. This method uses patchout, a technique consisting of discarding parts of the input to regularize the training process, while also allowing reducing the GPU memory and computations required for training. In this work, we investigate the effectiveness of this technique for music representation learning, considering the impact of specific design aspects.

We focus on the impact of using different combinations of tokens from different blocks of the transformer as embeddings, starting the training from different

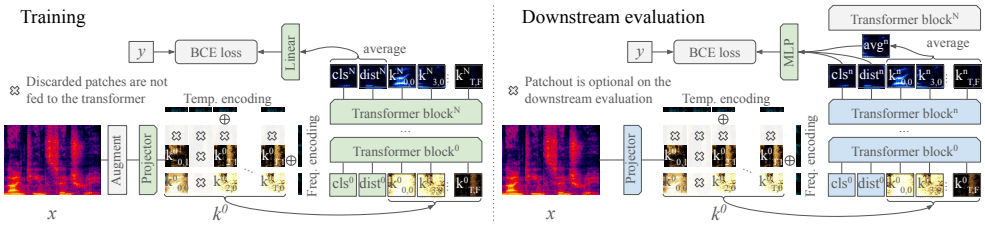


**Figure 6.1:** Illustration of our system at the training and downstream evaluation stages where $x$ is the input spectrogram, $k^0$ is the sequence of tokens after the patchout, $y$ is the target labels, and BCE is the binary cross-entropy loss. Trainable and frozen blocks are colored green and blue respectively.

pre-trained weights from publicly available models, using different input segment lengths, and using patchout at inference time to speed up the embedding extraction. Our experiments show that the best performance is obtained by extracting embeddings from the middle of the transformer and initializing it with weights pre-trained on other audio tasks. Contrary to previous studies based on CNNs, our transformers benefit from long input segments both in training and different downstream scenarios. Finally, we show that, on certain patchout conditions, our transformers are able to double the inference speed of an EfficientNet-B0 baseline while producing embeddings that obtain better performance on downstream tasks. Moreover, this approach has the advantage of being entirely configurable at inference time, allowing the throughput/performance tradeoff to be adapted to the task at hand.

The experiments of this chapter are based on Alonso-Jiménez et al. (2023b).[45]

## 6.2 Method

Our transformer, *MAEST*, has the same architecture as AST (Gong et al., 2021), ViT (Dosovitskiy et al., 2021), or PassT (Koutini et al., 2022), and features 12 blocks of self-attention plus a dense layer resulting in close to 87 million parameters. We use 16 × 16 patches, $x_{t,f}$, with a stride of 10 × 10. Similar to PaSST, we split the positional encoding into time/frequency encodings ($te_t$, $fe_f$) and apply patchout by randomly discarding entire rows and columns from the sliced spectrogram. The input sequence of tokens, $k^0$, is created as a linear projection of the patches plus the correspondent time/frequency encodings, $k^0_{t,f} = P(x_{t,f}) + te_t + fe_f$, where $P(\cdot)$ is a trainable linear layer.[46] $k^1$ to $k^{12}$ represent the output tokens of the respective transformer blocks. Similar to DeiT (Touvron et al., 2021) and PaSST, we extend $k^0$ with

[45]The code and pre-trained models are available at http://github.com/palonso/MAEST/

[46]Since the mel scale is not linear, we considered specialized projectors for each frequency patch. However, this did not improve the performance.

classification ($cls^0$) and distillation ($dist^0$) trainable tokens, which are initialized with the DeiT or PaSST pre-trained weights in the experiments involving these models. [47] We take the average of $cls^{12}$ and $dis^{12}$ tokens to feed a linear classifier targeting $y$.

Given this pipeline, we investigate the following aspects of the model:

- **Initialization weights.** Previous works showed the importance of initializing the transformer to weights pre-trained on ImageNet (Gong et al., 2021). To gain further knowledge, we consider three initialization options: the DeiT B↑384 model pre-trained on ImageNet (Touvron et al., 2021), the PaSST S S16 model pre-trained on mel-spectrograms from AudioSet, and random initialization.
- **Spectrogram segment length**. We consider spectrogram segment lengths of 5 to 30 seconds resulting in the architectures MAEST-5s, MAEST-10s, MAEST-20s, and MAEST-30s. In all cases, we take existing PaSST frequency and temporal encodings and interpolate them to the target shape as an initialization. We use patchout discarding 3 frequency and 15 temporal patches for MAEST-5s and increase the temporal patchout proportionally for models with longer input sequences (e.g., 60 patches for MAEST-20s).

## 6.3 Experiments and results

We train our models using Discogs20. The training task consists of a multi-label classification of the top 400 music styles from Discogs' taxonomy. We compare different training configurations in several downstream tasks by training Multi-Layer Perceptrons (MLP) on representations extracted from the transformers.

### 6.3.1 Dataset and pre-processing

Discogs20 is derived from a pool of 4 M audio tracks mapped to the release information from the Discogs website's public dump. All release metadata, which can include music style tags following a pre-defined taxonomy, is submitted by the community of platform users. *Master releases* group different versions of the same release such as special editions, or remasters. We obtain our training labels, $y$, at the master release level by first aggregating the

[47] We considered a teacher-student approach similar to DeiT by using a pre-trained MAEST-30 to generate pseudo-labels that were targeted by the $dist^{12}$ token in the training stage. We decided to omit the experiment details since it did not achieve a significant improvement.

style tags of all the associated releases and then discarding master releases with more than five style tags or without any style label among the 400 most frequent among our pool of tracks. We keep tracks longer than 20 seconds. Since the style annotations are done at the master release level, the resulting track annotations are expected to be noisy. We generate validation and testing subsets with approximately 40,000 tracks and a training set with 3.3 M tracks, ensuring that every artist appears on a single split. This pre-processing is similar to our previous work (Alonso-Jiménez et al., 2022), and additional details and statistics about the resulting dataset can be found in the repository accompanying this publication. For now on, we refer to this internal dataset as *Discogs20*.

From every track, we sample 30 seconds from the center of the track and downmix it to a mono channel at 16 kHz. We extract 96-bands mel-spectrograms, $x$, using 32 ms windows and a hop size of 16 ms compressed with the expression $log_{10}(1 + 10000x)$ similar to previous works in music tagging (Pons & Serra, 2019a; Alonso-Jiménez et al., 2022). The resulting representations are stored as half-precision floats (16 bits) resulting in 1.3 TB of data. Given that our dataset is in the order of magnitude of AudioSet (1.8 M vs. 3.3 M) and presents similar label density (2.7 average labels in AudioSet and 2.1 in Discogs20), we adopt the sampling strategy used in previous works (Koutini et al., 2022). On every epoch, we take a balanced sample of 200,000 tracks without replacement using the inverse label frequencies as sample weight. We normalize the input to the mean and standard deviation of the training set.

### 6.3.2 Pre-training

We use the Adam Optimizer with a weight decay of $1 \times 10^{-4}$ and train the model for 130 epochs. We warm up the model for 5 epochs and then keep the learning rate at $1 \times 10^{-4}$ until epoch 50. Then the learning rate is linearly decreased to 1e−7 during 50 additional epochs. We consider two sets of weights for inference: those from the last epoch and those obtained by taking the mean of the model's weights every 5 epochs from epoch 50 using Stochastic Weight Averaging (SWA). We pre-compute the mel-spectrograms for efficiency, which limits the set of data augmentations we could apply. We use mixup (Zhang et al., 2018) with $alpha = 0.3$ and SpecAugment (Park et al., 2019) by masking up to 20 groups of 8 timestamps and up to 5 groups of 8 frequency bands.[48]

| Dataset | Size | Lab. | Dur. | Av. | Split |
|---|---|---|---|---|---|
| MTGJ-Genre | 55,215 | 87 | FT | 2.44 | split 0 (Bogdanov et al., 2019c) |
| MTGJ-Inst | 25,135 | 40 | FT | 2.57 | split 0 (Bogdanov et al., 2019c) |
| MTGJ-Moods | 18,486 | 56 | FT | 1.77 | split 0 (Bogdanov et al., 2019c) |
| MTGJ-T50 | 54,380 | 50 | FT | 3.07 | split 0 (Bogdanov et al., 2019c) |
| MTT | 25,860 | 50 | 29s | 2.70 | 12-1-3 (Van Den Oord et al., 2014) |
| MSDs | 241,889 | 50 | 30 | 1.72 | usual (Lee et al., 2018) |
| MSDc | 231,782 | 50 | 30 | 1.31 | CALS (Won et al., 2021) |

**Table 6.1:** Automatic tagging datasets used in the downstream evaluation. The datasets are compared in terms of sample size, number of labels, audio duration (Full Tracks or excerpts of fixed duration), average labels per track, and the splits used in our evaluations.

### 6.3.3 Evaluation

We evaluate our models in several music automatic tagging datasets covering various musical notions. We consider the popular MagnaTagATune (MTT) and the Million Song Dataset (MSD) with the commonly used training, validation, and testing splits used in Van Den Oord et al. (2014) and Lee et al. (2018) respectively. Additionally, we report the performance of our models in the CALS split, which is an artist-filtered version of the MSD ground truth proposed by Won et al. (2021). Finally, we use the MTG-Jamendo Dataset, a dataset of Creative Commons music containing sub-taxonomies with the tags related to genre (MTGJ-Genre), moods and themes (MTGJ-Mood), and instrumentation (MTGJ-Inst), along with the top 50 tags (MTGJ-T50) in the dataset. We use the official split 0 for all the subsets similar to previous works (Won et al., 2020b; Manco et al., 2021; McCallum et al., 2022). Table 6.1 summarizes these datasets in terms of size, number of labels, audio duration, average number of labels per track, and used splits.

We evaluate our models by extracting internal representations from different blocks of the transformer and training MLP classifiers on top. Instead of averaging the $cls^{12}$ and $dist^{12}$ tokens as done in the training stage, we consider three types of representations, $cls^n$, $dist^n$, and the average of the tokens representing the input spectrogram patches ($avg^n$) after $n$ transformer blocks. Additionally, we evaluate the complementarity of these embeddings training MLP classifiers on stacks of the different tokens. To generate the dataset of embeddings, we average the embeddings extracted from half-overlapped segments across the entire audio available for the tracks in the downstream datasets. The same setup is used for the training, validation, and testing stages.

The downstream model is an MLP with a single-hidden layer of 512 dimen-

[48] We trained MAEST using 4 Nvidia 2080 RTX Ti GPUs with 12GB of RAM. The training takes 31 hours for MAEST-5 and 48 hours for MAEST-30.

sions with a ReLU activation and dropout. In the experiments described in Sections 6.3.4, 6.3.5, 6.3.6, and 6.3.8, we use a batch size of 128, drop out of 0.5 and train the model for 30 epochs. In the downstream evaluation from Section 6.3.7, we perform a grid search over the following hyper-parameters for each task:

- **batch size**: {64, 128, 256}
- **epochs**: {30, 40, 50, 60, 70, 80}
- **drop out**: {0.5, 0.75}
- **maximum learning rate**: $\{1 \times 10^{-3}, 1 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-5}\}$

The MLP is trained with the binary cross-entropy loss using the Adam optimizer with a weight decay of $1 \times 10^{-3}$. The learning rate is exponentially raised to its maximum value during the first 10 epochs, kept constant for the number of epochs, and linearly reduced until reaching $1 \times 10^{-7}$ at the end of training. After training, we report the performance on the testing set obtained using the weights from the epoch with the highest validation ROC-AUC.

### 6.3.4 Extracting embeddings from the transformer

We are interested in finding the optimal representations from the transformer to be used as embeddings. To do this, we extract representations $cls^n$, $dist^n$, and $avg^n$ from different transformer blocks $n \in [5, 12]$. To measure the complementarity of these features, we train MLPs fed with stacks of combinations of these representations. In this experiment, we use MAEST-30s initialized with PaSST weights and the MTT dataset.

Figure 6.2 shows mAP scores obtained with different stacks of embeddings extracted from the different transformer blocks. In accordance with previous studies (Castellon et al., 2021), we find that the embeddings with the best performance are found in the middle blocks of the transformer. This contrasts with the typical behavior of CNNs, where the best features are normally towards the last layers of the model, especially, when the downstream task is well aligned with the training task. Also, concatenating the features benefits the performance. In the remaining experiments, we fix our embedding to the stack ($cls^7$, $dist^7$, $avg^7$).

### 6.3.5 Impact of the initial weights

Due to the lack of inductive biases present in architectures such as CNNs, transformers are heavily dependent on pre-training. Because of this, many

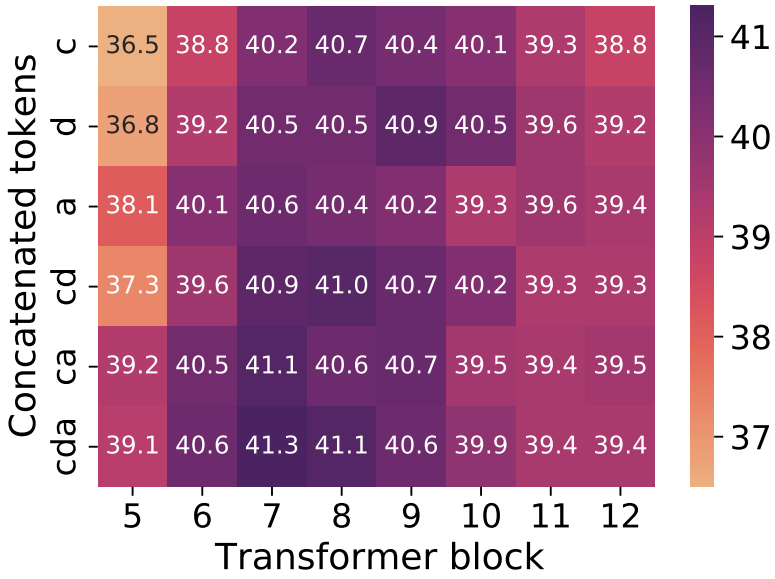


**Figure 6.2:** mAP scores were obtained with our evaluation setup in the MTT dataset using embeddings extracted from different blocks and tokens from the transformer. We evaluate the *cls* (c), *dist* (d), and *avg* (a) tokens and stacks of their combinations extracted from the transformer blocks 5 to 12.

audio transformers are initialized with weights transferred from image tasks (Gong et al., 2021; Koutini et al., 2022). We evaluate the impact of initializing our models from the weights of DeiT (Touvron et al., 2021) (image input), the best single PaSST model (Koutini et al., 2022) (mel-spectrogram input), and random initialization. In this experiment, we use MAEST-10s and its version with SWA weights, MAEST-10s-swa. Although our main focus is to evaluate MAEST on public downstream datasets, we also report their performance on the training task to provide additional insights.

| Model | RW | DeiT | PaSST |
|---|---|---|---|
| *Pre-training task: Discogs20* | | | |
| MAEST-10s | 20.5 | 22.7 | 22.8 |
| MAEST-10s-swa | 20.1 | 23.2 | 23.5 |
| *Downstream task: MTT* | | | |
| MAEST-10s | 38.7 | 40.4 | 41.1 |
| MAEST-10s-swa | 39.0 | 40.2 | 41.0 |

**Table 6.2:** mAP scores obtained in the training and downstream tasks using different initializations. We considered Random Weights, and pre-trained weights from DeiT and PaSST.

| Model | 5s | 10s | 20s | 30s |
|---|---|---|---|---|
| *Pre-training task: Discogs20* | | | | |
| MAEST-$T$ | 21.1 | 22.8 | 24.8 | 26.1 |
| MAEST-$T$-swa | 21.3 | 23.5 | 25.8 | 27.0 |
| *Downstream task: MTT* | | | | |
| MAEST-$T$ | 40.8 | 41.1 | 41.2 | 41.7 |
| MAEST-$T$-swa | 40.9 | 41.0 | 41.2 | 41.5 |

**Table 6.3:** mAP scores obtained in the training and downstream tasks using different spectrogram segment lengths. $T$ represents the spectrogram segment length.

Table 6.2 shows the performance in both, the training (Discogs20), and a downstream (MTT) task. In both cases, the scores are higher when the training is started from pre-trained weights. Since the PaSST weights result in slightly higher performance, we use this initialization for the remaining of this work. Regarding the SWA, we observe a positive effect on the training task when the model is initialized with pre-trained weights. However, we do not observe improvements in the downstream task.

### 6.3.6 Effect of the input segment length

We train MAEST using input segment lengths ranging from 5 to 30 seconds. In our experiments, we keep the frequency patchout constant and proportionally increase the temporal patchout. For our models with segment lengths of 5, 10, 20, and 30 seconds we discard 15, 30, 60, and 90 temporal patches respectively.

Table 6.3 shows the performance of the MAEST models with respect to their input spectrogram segment length in terms of mAP both in the training (Discogs20) and a downstream (MTT) evaluation. While music tagging CNNs tend to reach their peak of performance with receptive fields of 3 to 5 seconds (Choi et al., 2017b), attention-based systems have shown the capability to take advantage of longer temporal contexts (Won et al., 2021). Our models are consistent with this trend, reaching their best performance when trained on segments of 30 seconds. Although even longer segments could be beneficial, we could not use them while keeping the same model size due to GPU memory limitations.

### 6.3.7 Performance in downstream tasks

Considering our previous findings, we extend the evaluation of MAEST to a number of downstream datasets. We evaluate MAEST-10s, MAEST-20s, MAEST-30s,

and a baseline consisting of embeddings from the DiscogsEffNet (EffNet) model introduced in Section 3.3.3. This model consists of an EfficientNet-B0 architecture (Tan & Le, 2019) trained in the same 400 music style tags from Discogs20 following previous work (Alonso-Jiménez et al., 2022). Additionally, we report the performance of SOTA models from the literature considering approaches fully trained in the downstream tasks and based on embeddings plus shallow classifiers.

Table 6.4 shows the results of the different models in terms of ROC-AUC and mAP. [49] We observe that all the MAEST models outperform the baseline in all tasks, confirming the superiority of the proposed approach. Additionally, we achieved a new SOTA for the MTGJ-Genre, MTGJ-Inst, and MSDc datasets, although other models remain superior in the rest of the datasets. Specifically, MuLan (Huang et al., 2022b) obtains higher mAP in MTT, probably because it is trained on a much larger corpus of 40 M tracks. In MTGJ-Moods, MTGJ-T50, MTT, and MSDs, Musicset-Sup, a model trained on a curated dataset of 1.8 M expert annotations, remains superior (McCallum et al., 2022). In both cases, the advantage is likely due to the superiority of the training task. Notably, none of these models is public, which makes MAEST the best open music embedding extractor available.

[49] reference correspondences in the table:
[97] Knox et al. (2020)
[135] Pons & Serra (2019a)
[172] Won et al. (2020b)
[169] Won et al. (2021)
[7] Alonso-Jiménez et al. (2022)
[119] McCallum et al. (2022)

| | MTGJ-Genre | | MTGJ-Inst | | MTGJ-Mood | | MTGJ-T50 | | MTAT | | MSDs | | MSDc | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ROC | mAP | ROC | mAP | ROC | mAP | ROC | mAP | ROC | mAP | ROC | mAP | ROC | mAP |
| *State of the art* | | | | | | | | | | | | | | |
| Fully-trained | - | - | - | - | 77.8 | 15.6 | 83.2 | 29.8 | 90.69 | 38.44 | 92.2 | 38.9 | 89.7 | 34.8 |
| | - | - | - | - | [97] | [97] | [135] | [135] | [172] | [172] | [169] | [169] | [169] | [169] |
| Embeddings | 87.7 | 19.9 | 77.6 | 19.8 | 78.6 | 16.1 | 84.3 | 32.1 | 92.7 | 41.4 | - | - | 90.3 | 36.3 |
| | [7] | [7] | [7] | [7] | [119] † | [119] † | [119] † | [119] † | [83] † | [119] † | - | - | [119] † | [119] † |
| *Baseline* | | | | | | | | | | | | | | |
| EffNet-B0 | 87.7 | 19.9 | 77.6 | 19.8 | 75.6 | 13.6 | 83.1 | 29.7 | 90.2 | 37.4 | 90.4 | 32.8 | 88.9 | 32.8 |
| *Baseline* | | | | | | | | | | | | | | |
| EffNet-B0 | 87.7 | 19.9 | 77.6 | 19.8 | 75.6 | 13.6 | 83.1 | 29.7 | 90.2 | 37.4 | 90.4 | 32.8 | 88.9 | 32.8 |
| *Our models* | | | | | | | | | | | | | | |
| MAEST-10s | 88.1 | 21.1 | 79.7 | 22.4 | 77.9 | 15.1 | 84.0 | 31.3 | 91.8 | 41.0 | 91.5 | 36.9 | 88.9 | 32.7 |
| MAEST-20s | 88.1 | 21.4 | 79.9 | 22.6 | 77.9 | 15.2 | **84.1** | **31.5** | 91.8 | 41.0 | 92.1 | 39.2 | 89.5 | 34.5 |
| MAEST-30s | **88.2** | **21.6** | **80.0** | **22.9** | **78.1** | **15.4** | 84.0 | **31.5** | **92.0** | **41.9** | **92.4** | **40.7** | **89.8** | **35.4** |

**Table 6.4:** ROC-AUC and mAP scores obtained in the downstream tasks. Our baseline consists of an EffNet-B0 architecture trained in Discogs20. Additionally, we report the SOTA results distinguishing models with all parameters trained in the downstream tasks (fully trained) and models evaluated with shallow classifiers. For every task, we mark in bold the best score obtained by a MAEST model and highlight in grey models achieving better performance than the best open alternative. † Models not publicly available.

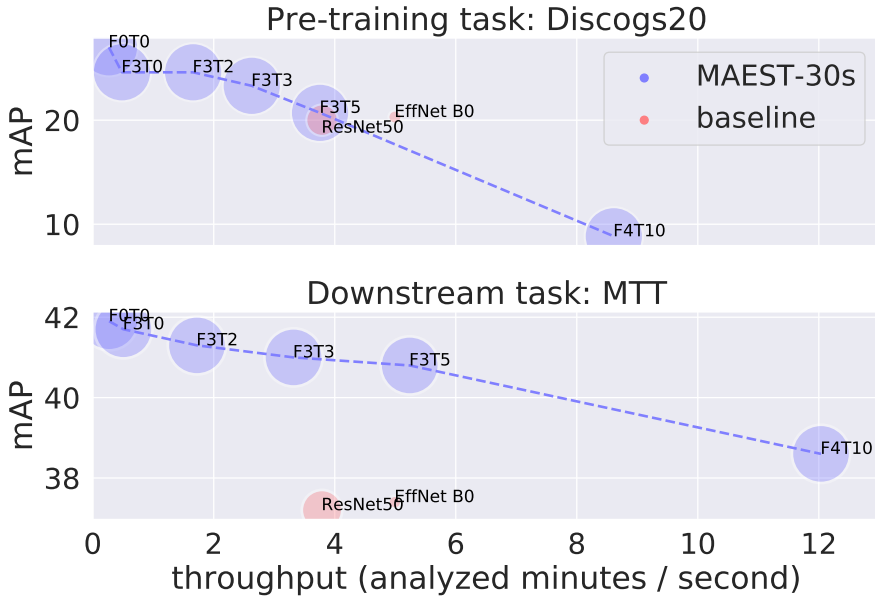


**Figure 6.3:** mAP scores against throughput for MAEST-30s under different amounts of frequency (F) and time (T) patchout. The radius is proportional to the parameter count and the inference is performed on the CPU.

### 6.3.8 Faster feature extraction with inference patchout

Inferring with transformers is typically more computationally expensive than with CNNs. To speed up our models, we consider using two types of patchout at inference time: Time-wise, we keep one out of $T$ spectrogram patches. Frequency-wise, we discard specific rows of patches. We experiment with temporal patchout using $T \in [2, 3, 5, 10]$ and frequency patchout of 3 and 4 patches corresponding to the first and the two last blocks, and the two first and two last blocks respectively. The embeddings obtained under different patchout settings are compared in the training and a downstream task following our downstream evaluation approach on the MTT dataset.

Figure 6.3 shows the mAP scores on the training and downstream tasks under different patchout settings. In the downstream task, even under strong patchout settings, MAEST-30s overcomes the throughput of standard CNN architectures by two to three times while keeping higher mAP. On the training task, this technique is not so effective because the classifier is frozen and cannot adapt to the effects of patchout, and also it operates on tokens from the last block, which requires more computations.

## 6.4 Conclusions

In this chapter, we demonstrate that pure-attention-based transformers can deliver strong performance for music representation learning and study how different design decisions affect the downstream tasks. We set our pre-training objective to music style tagging, and limit our experiments to models fitting in consumer-grade GPUs. We analyze the impact of using activations from different blocks and tokens of the transformer as embeddings, initializing the models with different pre-trained weights, and using input segment lengths ranging from 5 to 30 seconds. Our experiments show that the best embeddings come from a stack of features from the middle blocks of the transformer, initializing from weights pre-trained in audio event recognition provides the best performance, and that longer input segments correlate with better results. We evaluate our models in six popular music tagging datasets, and experiment with patchout at inference time, finding that it allows speeding up significantly the transformer while producing embeddings with better performance/speed trade-offs than our convolutional baselines. Finally, we present MAEST, a family of transformers for music style tagging and embedding extraction, which are publicly available and achieve SOTA performance among currently available music representation models. Chapter 8 discusses the usage of MAEST in real-world applications and demos.

CHAPTER 7

# Sonified prototypical learning for interpretable music classification

## 7.1 Introduction

In previous chapters, we focused on the development of new music representation models from the perspective of the pre-training dataset (Chapter 3), type of supervision (Chapters 4, and 5), and model architecture (Chapter 6). In all these cases, we were able to measure performance improvements by means of standard metrics obtained with auxiliary downstream models. While this approach is an established protocol in the field, all models involved operate as black boxes, failing to provide detailed insights into their inner workings or decision-making process. In this chapter, we focus on the interpretability of music classifiers and propose a new method that allows the creation of interpretable classifiers on top of pre-trained music representations.

*Interpretability* can be understood as the capability of an algorithm or model to be comprehensible, explainable, and understandable, which allows an external observer to decipher its behavior and discern its decisions (AI HLEG, 2019). In the context of sound and music-related applications (such as sound engineering, music production, and music recommendation), faithful human-understandable explanations of model predictions can increase trustworthiness and enhance user experience (Arrieta et al., 2020). From a developer's perspective, an interpretable model could better reveal potential issues of its data or inner workings, allowing the detection of biases, malfunctions, or possible adversarial attacks (Sturm, 2017; Prinz et al., 2021). Ultimately, interpretability can also provide insights into the target problem, thus helping researchers learn more about it.

In this chapter, we show that it is possible to leverage existing prototype learning methods supporting sonifications, such as the metho proposed by Zinemanas et al. (2021), and decouple the training of the autoencoder and the prototypical network, unlocking the possibility to operate on top of arbitrary pre-trained music representations. While the original APNet model was applied to voice commands, environmental sounds, and music instrument recognition tasks, we focus on music genre and instrument recognition to be more aligned with the downstream tasks considered in previous chapters.

Our method allows taking advantage of SOTA representations trained on large datasets, while still providing interpretable models, and competitive classification performance. Additionally, relying on a generative model eliminates the need to transfer information from specific training samples to reconstruct the prototypes as in previous works. In this chapter, we show that it is possible to decouple the training of the autoencoder and the prototype network, which unlocks the possibility of using a wider range of representations, potentially with more discriminative power. Additionally, relying on a generative model to eliminate the need to transfer information from specific training samples to reconstruct the prototypes as in previous work. Finally, we extend the technique of prototype-based audio classification to the task of music genre classification for the first time.

The experiments of this chapter are based on (Alonso-Jiménez et al., 2024), presented in May 2024 at the ICASSP Workshop on Explainable AI for Speech and Audio at Seoul, Korea.[50]

## 7.2 Method

In this section, we introduce Prototype EnCodecMAE (PECMAE), a method inspired by APNet (Audio Prototype Network) (Zinemanas et al., 2021). Prototypical models allow interpretability by measuring the similarity between model inputs and the prototypes in the encoder latent space (Li et al., 2018). Additionally, APNet features an autoencoder architecture that allows to sonify the prototypes for further insights. The key differences of our proposal are that we rely on a pre-trained autoencoder (EnCodecMAE, Pepino et al. (2023)) instead of jointly optimizing it for the classification task and that we use a diffusion decoder to sonify the prototypes instead of using information from specific samples of the training set.

The main components of our system are depicted in Fig. 7.1. The generative autoencoder operates through the embeddings $z$. The prototype network solves

[50]The code and pre-trained models are available at http://github.com/palonso/PECMAE/

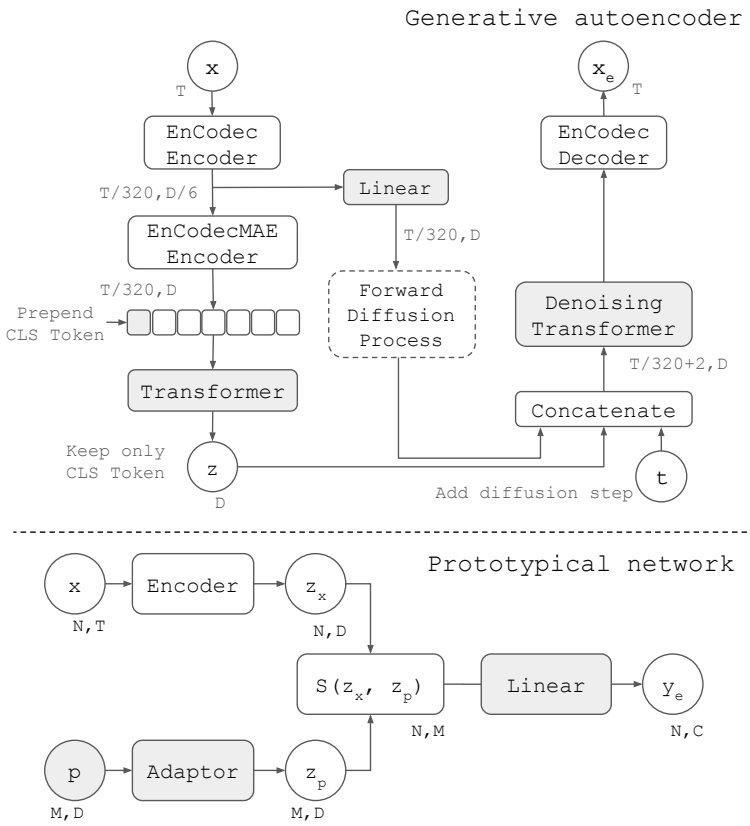


**Figure 7.1:** Diagram of the proposed PECMAE model. The colored boxes indicate trainable modules.

a classification task by learning prototypes $p$ in the embedding space of $z$ so that they can be decoded to audio.

While EnCodecMAE already provides a reconstructable latent space, we observed poor prototype sonification after initial experimentation. We hypothesized that this is because the temporal resolution is excessive and that the system would benefit from summarizing this dimension so that the prototypes were learned in a more abstract space. To this end, we train an autoencoder on top of EnCodecMAE features, consisting of a transformer encoder that summarizes $T = 4$ seconds of audio (a sequence of 300 768-dimensional EnCodecMAE features) into a single vector $z \in \mathbb{R}^D$ with $D = 768$, and a decoder based on a latent diffusion model conditioned with $z$ to generate the EnCodec features corresponding to the original audio. We prepend a CLS token to the input of the transformer encoder and use the corresponding output element as $z$. The conditioning of the diffusion decoder is implemented by prepending $z$ to the noisy EnCodec inputs. With this approach, our compression rate is 28 times higher than that of APNet, facilitating scalability both in terms of the

number of prototypes and the input duration.

The prototypical network works by measuring the similarity, $S$, between the embeddings of a batch of input audio instances $z_x \in \mathbb{R}^{N \times D}$ and the set of prototypes $z_p \in \mathbb{R}^{M \times D}$, where $N$ is the batch size, $M$ is the number of prototypes, $z_p$ is the projection of each prototype $p$ using an optional 1-layer transformer adaptor,[51] and $S = exp^{-||z_x - z_p||_2^2}$. Given $C$ classes, each one is assigned the same number of prototypes, $M/C$. The prototypes corresponding to each class are initialized to centroids obtained by applying k-means clustering over the embeddings $z_x$ of the class samples. Finally, we use a linear layer to map $S$ into class logits. This layer is initialized so that the connection of each prototype with its respective class is 1 but 0 with the others. During training, the prototypes, the prototype transformer adaptor, and the linear layer are optimized while the autoencoder is kept frozen.

PECMAE employs two loss functions. We use binary cross-entropy to optimize the classification task, $L_c$. In addition, we define a prototype loss that minimizes the L2 distance between each prototype and the closest sample among the instances of the same class in the batch, $z_{xc}$,

$$L_p = \frac{1}{M} \sum_{j=1}^{M} \min_i ||z_{xc,ij} - z_{p,ij}||_2^2. \tag{7.1}$$

The goal of this term is to prevent prototypes from diverging too much from real samples to favor interpretable reconstructions. Note that we avoid using an additional loss term minimizing distances between samples and prototypes as in previous works (Zinemanas et al., 2021; Li et al., 2018) since, in our case, the sample representations are not trainable. Finally, the losses are aggregated using a weighting factor $\lambda$, $L = \lambda L_c + (1 - \lambda) L_p$. After training, the decoder can be used to sonify the prototypes.

## 7.3 Experiments and results

We compare the classification accuracy of the proposed model with the SOTA and baseline systems and study the characteristics of learned prototypes on one music instrument and two genre classification datasets (Table 7.1).

### 7.3.1 Datasets

*Medley-Solos-DB* (Lostanlen & Cella, 2016) is an instrument recognition dataset consisting of 3-second recordings for eight instruments: clarinet, distorted

[51] We also report results without this adaptation layer (PECMAE-NA-5).

| Datasets | Classes | Samples | Duration (h) |
|---|---|---|---|
| Medley-solos-DB | 8 | 21,571 | 17.2 |
| GTZAN | 10 | 919 | 7.6 |
| XAI-Genre | 24 | 18,634 | 155.2 |

**Table 7.1:** Considered datasets in terms of number of classes, samples, and total duration in hours.

electric guitar, female singer, flute, piano, tenor saxophone, trumpet, and violin. While our main interest is in genre recognition, we considered evaluating our model in one dataset used in previous works for comparison purposes.

*GTZAN* is a popular genre recognition dataset consisting of 30-second excerpts across 10 broad musical genres. We consider a filtered version of the dataset discarding duplicated and corrupted tracks identified by Sturm (Sturm, 2013). To achieve a genre-balanced split, we use track IDs ending in 8 for validation (e.g., blues.00008) and in 9 for testing.

*XAI-Genre* is an in-house dataset of 30-second audio previews annotated by 24 genre classes, retrieved from the Spotify API,[52] built for this study and our planned future work on evaluation methodologies for interpretable genre recognition. This dataset is 20 times larger than GTZAN in terms of music tracks and includes more than twice the number of classes, adding complexity and diversity to the classification task.

### 7.3.2 Implementation details

We train the diffusion autoencoder in the Free Music Archive dataset (Defferrard et al., 2017), composed of 106,574 30-second music tracks using batches of 128 4-second segments. The autoencoder consists of a transformer with a 2-layer encoder and an 8-layer decoder, which is then trained for 330,000 steps using the AdamW optimizer with a weight decay of 0.05 and a fixed learning rate of $1 \times 10^{-4}$. We apply classifier-free guidance, setting the unconditional probability to 0.1 during training, and use the V-Diffusion objective (Salimans & Ho, 2021). Our experiments use the EnCodecMAE base model, which has 10 transformer layers and 12 attention heads and is trained in a mixture of Audioset, Librilight Medium, and Free Music Archive.

We train PECMAE models in the GTZAN, Medley-Solos-DB, and XAI-Genre datasets featuring 1 to 40 prototypes per class. In all cases, we use z-score normalization and a batch size of 256 samples. We prefer a rather larger batch size, considering that a larger pool of tracks to approximate our embeddings will lead to a better embedding reconstruction. All the models are trained

[52] https://developer.spotify.com/documentation/web-api

for 150,000 steps using the Adam optimizer with a weight decay of $1 \times 10^{-5}$ using the OnceCycleLR learning rate scheduler with a peak value of $1 \times 10^{-3}$. The hyperparameter $\lambda$ controls the weight of the classification and prototype loss components. After preliminary experiments, we set $\lambda$ to 0.25 to favor prototype reconstruction. In all cases, we use fixed training, validation, and testing splits, and test the models using the checkpoint of the step achieving the lowest validation loss. Since our autoencoder operates on 4-second segments, in testing we average the class logits for non-overlapping segments in the track and compute the class-normalized accuracy.

### 7.3.3 Classification Results

Table 7.2 presents the metrics for all compared methods on each of the considered datasets using the same splits. We report SOTA from literature together with a Multi-Layer Perceptron (MLP) trained on SOTA audio embeddings for genre classification (MAEST) (Alonso-Jiménez et al., 2023b) and the original APNet model featuring 5 prototypes per class. Additionally, we train MLPs with EncodecMAE (ECMAE) embeddings and its summarized version (ECMAE-S), referred to as $z$ in Fig. 7.1. Since our prototypes are learned in the ECMAE-S space, these serve as our reference for the performance ceiling. For PECMAE we consider 1, 3, 5, 10, 20, and 40 prototypes per class, plus a version without the transformer adaptation layer (NA).

The results show that ECMAE achieves lower performance compared to methods based on large supervised datasets (Alonso-Jiménez et al., 2023b) in XAI-Genre, or careful feature design (Andén et al., 2019) in Medley-Solos-DB. ECMAE-S has slightly lower accuracies than ECMAE due to information loss associated with the higher compression rate but produces better sonification results.[53] As a general trend, PECMAE performance increases with the number of prototypes and is comparable to or slightly below ECMAE-S. Finally, we find that our method achieves higher classification performance than APNet in XAI-Genre and Medley-Solos-DB, even when fewer prototypes per class are used. We hypothesize that this is due to the powerful representations of EncodecMAE, which had been trained in a much larger data collection and already showed good performance in music-related tasks (Pepino et al., 2023).

### 7.3.4 Effect of the decoder

Since our autoencoder relies on a generative model, its decoder is constrained to synthesize audio close to its training distribution, which can result in reconstruction biases. After preliminary tests, we found that a decoder based

[53] Operating in ECMAE's higher-dimensional space resulted in poor sonification of the prototypes.

| | Params. | GTZAN | Medley | XAI-Genre |
|---|---|---|---|---|
| *State of the Art* | | | | |
| Literature | | 82.1 (Kim et al., 2018c) | 78.0 (Andén et al., 2019) | - |
| MLP MAEST | 300K | 95.6 | - | 62.9 |
| *Baseline* | | | | |
| APNet-5 | 2.7-4.2M | 87.4 | 65.8 | 48.0 |
| *Ceiling* | | | | |
| MLP ECMAE | 100K | 85.7 | 75.7 | 58.0 |
| MLP ECMAE-S | 100K | 85.9 | 72.1 | 56.1 |
| *Ours* | | | | |
| PECMAE-NA-5 | 31-90K | 81.8 | 66.8 | 44.0 |
| PECMAE-1 | 5.5M | 80.8 | 63.9 | 44.0 |
| PECMAE-3 | 5.6M | 82.8 | 67.6 | 48.6 |
| PECMAE-5 | 5.6M | 83.8 | 70.2 | 50.1 |
| PECMAE-10 | 5.6M | **86.9** | 71.1 | 51.8 |
| PECMAE-20 | 5.6M | 85.9 | 69.2 | 52.8 |
| PECMAE-40 | 5.8M | 85.7 | **71.3** | **53.6** |

**Table 7.2:** Normalized classification accuracies.

on V-diffusion was providing more faithful reconstructions than an alternative based on a conditional language model over EnCodec tokens decoder using a GPT2 architecture. We verified that the important class information was not altered in the decoding process by measuring the class predictions for the synthesized prototypes (above 99% accuracy for PECMAE-20).

### 7.3.5 Sonifying the prototypes

As part of developing the proposed models, we conducted iterative listening examinations of the synthesized prototypes.[54] We observe that the sonification of the prototypes results in sounds that are far from resembling real class instances and instead have a sonic texture quality. However, we can identify many of the classes both in the case of instruments and music genres from blind listening to the prototypes. Clearly, the sonification of instrument prototypes is more convincing than genre sonification due to lower sound complexity (monophonic notes vs complete full-mix music tracks). Increasing the number of prototypes tends to provide similar-sounding prototypes, even though we can identify differences in some cases (e.g., pitches or types of timbre). Notably, the autoencoder was able to reconstruct instances from all classes in our datasets with much better quality than the prototypes, which suggests that

[54] Prototype sonifications and complementary results available at https://palonso.github.io/pecmae/

the bottleneck is in the statistical averaging process when learning the prototypes and not in the decoder itself. Overall, the sonification would not be appropriate for end users but is insightful for model developers (Sturm, 2017), especially revealing how adversarial attacks can be devised (Prinz et al., 2021).

## 7.4 Conclusions

In this chapter, we propose an interpretable classification system that learns prototypes in the embedding space of an autoencoder and enables their sonification. Our results show that it is possible to achieve prototype-based models that do not notably degrade classification performance while providing a certain level of interpretability, which proves helpful in developing classification models. Through sonification, we observe that the models learn sonic textures instead of more complex temporal structures as the basis of their classification decisions for most of the classes. The proposed approach motivates new directions for interpretable music audio models. We will consider replacing the self-supervised embeddings with representations optimized for the classification tasks and investigate auto-encoding techniques that handle longer input sequences and increase the diversity of the learned prototypes.

CHAPTER 8

# Applications and outreach

## 8.1 Essentia

Essentia is an open-source C++ library with Python and JavaScript bindings for audio analysis and audio-based MIR developed at the Music Technology Group - Universitat Pompeu Fabra (MTG) since 2006. The library contains an extensive collection of reusable algorithms that implement audio input/output operations, digital signal processing functionalities, and a large variety of spectral, temporal, tonal, and high-level music descriptors. Besides, Essentia can be complemented with a wrapper for inference with TensorFlow models and a set of pre-trained models for audio tagging, music classification, and other tasks. Many of these models rely on representation learning models developed in the context of this thesis.

Essentia is released under the Affero GPLv3 license[55] and is also available under a proprietary license upon request. It has served in several projects including both research activities and large-scale industrial applications. Its most common use cases include music classification, semantic auto-tagging, music similarity and recommendation, visualization, and interaction with music, sound indexing, musical instrument detection, cover detection, beat detection, and acoustic analysis of stimuli for neuroimaging studies.

Essentia is not a framework, but rather a collection of algorithms and models plus the infrastructure required for operating them within a library. It is designed with a focus on the robustness, performance, and optimality of the provided algorithms, including computational speed and memory usage, as well as ease of use. Importantly Essentia is designed so that the provided functionality is easily expandable in terms of new algorithms and models, and it uses standard i/o formats for audio and data interchange, which makes it easy to integrate with other software. For example, Essentia uses FFmpeg[56] for

[55] https://www.gnu.org/licenses/agpl-3.0.en.html
[56] https://ffmpeg.org/

audio decoding and encoding, and NumPy[57] for data interchange with Python. Additionally, a large part of Essentia's algorithms is well-suited for real-time applications.

### 8.1.1 Essentia-TensorFlow

The goal of Essentia-TensorFlow[58] is to extend Essentia to support deep learning models with fast inference times and a capability to run on CPUs or acceleration hardware such as GPUs. While we could have considered Python-based solutions similar to madmom (Böck et al., 2016), we were interested in an integrated C++ solution to take advantage of fast computational speed which is crucial in many applications. The decision to use TensorFlow instead of other options such as PyTorch (Paszke et al., 2017) was motivated by the stability of its C API,[59] its active development to keep up with the state of the art, and a huge availability of existing research relying on it.

To this end, we developed a set of algorithms that allow reading frozen models from Protobuf files, generating tensors from 1D or 2D audio representations and running TensorFlow sessions. The Essentia algorithms implementing TensorFlow support were designed following these criteria:

- **Efficiency.** All dataflow between algorithms for audio feature extraction and model inference should be implemented in C++ without any overhead conversion to Python. We also decided to use TensorFlow frozen models where variables are converted to constants allowing us to remove some training operations.

- **Flexibility.** The deep learning field moves on fast. Therefore, generic support for any TensorFlow architecture should be provided. This can be done by loading both the architecture and the weights from external files instead of hard-coding any particular architecture. Importantly, it is also possible to import the models from other frameworks via intermediate formats such as ONNX.[60]

- **Access to intermediate layers.** Sometimes intermediate layers of a model are valuable as they can be used, for example, as features for other tasks. For this reason, it should be possible to extract the output tensors from any layer.

[57] https://numpy.org/
[58] https://pypi.org/project/essentia-tensorflow/
[59] https://www.tensorflow.org/install/lang_c
[60] https://onnx.ai/

- **Real-time analysis.** Being able to run computations in real-time is one of the key features of Essentia that should be supported by its deep-learning algorithms. The latency and the overall real-time capability ultimately depend on the design of a model, its computational cost for inference, and/or receptive field.

The provided functionality does not include training of the TensorFlow models, only inference. Users can be flexible in selecting the way how to train their models as long as they ensure the compatibility of the input features used for training with their implementation in Essentia for inference. Ideally, users could also use Essentia features on the training stage to ensure the best compatibility. Many deep learning models proposed in research have been trained using features from different software, but they can be also reproduced in Essentia as its algorithms are sufficiently configurable for most input audio features. For example, in the case of mel-spectrograms, Essentia can reproduce virtually any existing common mel implementation.

Most TensorFlow models can then be made compatible with Essentia by freezing and serializing them into Protocol Buffers[61] files or SavedModel[62] format. This is a simple process that can be easily done using available Python scripts.

As an example of the efficiency of our framework, we compared inference times for MusiCNN (Pons & Serra, 2019a) using the original implementation in Python and our algorithms called from Essentia's Python bindings. The original feature extraction time, based on Librosa, took 6.51 seconds compared to 2.30 for Essentia. Loading the model and predicting took 2.07s and 1.66s, respectively. In total, considering the extra overhead of dataflow, the difference is 8.60 to 3.34 seconds, meaning that our framework is 2.5 times faster for the entire end-to-end process from loading audio to inference. These time estimations were done by averaging 10 trials of analysis of a 3:27 MP3 file on an i7 6700 CPU.

Essentia-TensorFlow is available through `pip` [63] and it is compatible with Python 3.6 and above for Linux and macOS. figure 8.1 shows an example of the usage of essentia-tensorflow. internally, the algorithm `tensorflowpredictmaest` is loading, downsampling, and downmixing the audio signal, extracting the mel-spectrogram representations, preparing inference batches with the appropriate shape, and running a TensorFlow session with the model loaded from a protocol buffer file in c++. we provide examples of how to install and use the framework, create TensorFlow frozen models, and run those models to generate predictions on the example of music auto-tagging.[64]

[61] https://protobuf.dev/
[62] https://www.tensorflow.org/guide/saved_model
[63] PyPI installation command: `pip install essentia-tensorflow`
[64] https://mtg.github.io/essentia-labs/

```
from essentia.standard import (
    monoloader,
    tensorflowpredictmaest,
)

x = monoloader(filename="song.wav", samplerate=16000)()

embeddings = tensorflowpredictmaest(
    graphfilename="discogs-maest-30s-pw-1.pb"
)(x)
```

**Figure 8.1:** this code snippet uses essentia-tensorflow algorithms to run a maest model from chapter 6.

the final goal of essentia-tensorflow is to provide fast c++ inference for SOTA deep learning models in essentia suitable for deployment in diverse mir applications. to the best of our knowledge, this is the first effort of its kind to integrate arbitrary deep-learning models into a MIR library. the new functionality for using models is designed to be fast, easy, and flexible, and it is especially attractive for applications requiring computational efficiency, such as large-scale analysis on millions of tracks, real-time processing, or inference on weak devices. additionally, our algorithms are designed in a way that allows using the models as feature extractors, making this solution suitable for transfer learning scenarios.

### 8.1.2 essentia models

most models created throughout this dissertation are available in *essentia models*.[65] this is a hub for the models that can be used with essentia, including end-to-end models and systems combining a representation learning model with a classifier or regressor. to this end, we offer models for specific use cases (auto-tagging, tempo estimation, source separation, and music classification by genre, mood, and instrumentation). the models available cover the tasks of music auto-tagging, instrumentation recognition, MGR, glsmer, voice detection, danceability estimation, music engagement, and approachability regression, arousal and valence estimation, tempo estimation, and source separation. additionally, we provide modes for audio scene recognition. all of these models are publicly available for researchers and practitioners under cc by-nc-sa 4.0,[66] and we plan to add more models in the future. currently essentia models host over 300 models, including the ones developed in this thesis.

figure 8.2 shows a screenshot of the essentia models site. for every model, the

[65] https://essentia.upf.edu/models/

[66] https://creativecommons.org/licenses/by-nc-sa/4.0/

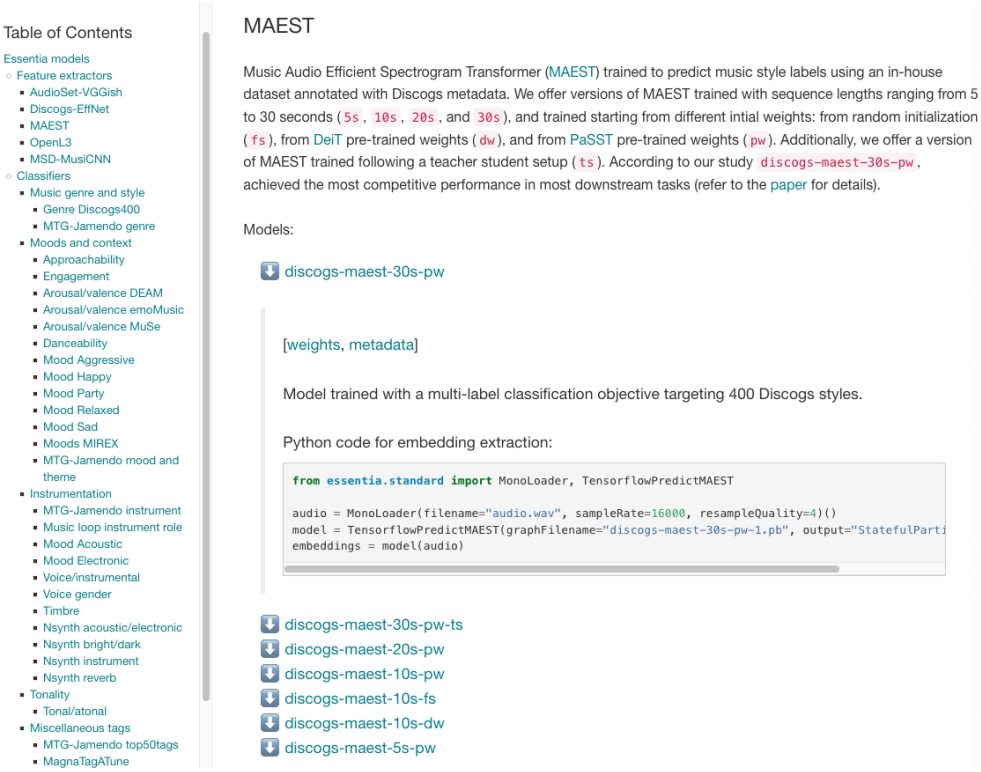


**Figure 8.2:** essentia models site screenshot.

site provides a brief description of the model and the variations available, an explanation of the expected input and output, and example inference python snippet, and links to the weights and metadata files.

### 8.1.3 Essentia.js

Correya et al. (2020)'s Essentia.js is a JavaScript (JS) wrapper library for Essentia. The library's core is powered by Essentia's C++ back-end using WebAssembly[67] built via emscripten[68] along with a high-level JS and TypeScript API and add-on utility modules. Essentia.js allows running an extensive collection of music and audio analysis algorithms and models directly on the web browser or node.js runtime applications.

In addition to the core library, Essentia.js has a few add-on modules to facilitate common MIR tasks. In particular, *Essentia.js-extractor* contains predefined feature extractors to compute bpm, beat positions, pitch, predominant melody, key, chords, chroma features, MFCC, etc. Also, *essentia.js-plot* provides helper

[67] https://webassembly.org/
[68] https://emscripten.org/

functions for visualization of MIR features, allowing both real-time and offline plotting in a given HTML element.

All the algorithms in Essentia.js are configurable similarly to Essentia's C++ or Python APIs. The tools provided with the library allow creating lightweight builds, containing only the specific algorithms required for a particular application. Many of the algorithms and models integrated into Essentia.js support both real-time audio analysis use cases similar to Essentia's C++ core library. See the online documentation for more details.[69]

Our efforts to deploy DNN models on the web were published by Correya et al. (2021a,b) and received the best paper award at the 2021 Web Audio Conference. The ability to deploy client-side DNN models is already a widely supported feature within machine learning frameworks. At the time we started this work, the main tools being actively developed were TensorFlow.js[70] and ONNX.js.[71] We identified the following advantages of using TensorFlow.js for our case:

- It was the most actively maintained project with extensive documentation and example projects.
- It was part of the TensorFlow ecosystem, the same deep-learning library used in Essentia.
- It supported multiple back-end options such as WebGL and WebAssembly for inference on browsers or Node.js, which provides flexibility for future scenarios.
- It provided a tool to convert the format of existing Essentia models to its required input format.

We used the converter provided by TensorFlow.js to port the models. While all our models were stored as TensorFlow v1 frozen protocol buffers, this tool also supports conversion from TensorFlow v2 SavedModels and Keras hdf5 files. Additionally, PyTorch models can be exported in ONNX format and converted to TensorFlow v2 SavedModels with the official tools.[72] Covering the two major machine learning frameworks means that the vast majority of the models developed for research are suitable for web deployment.

The only additional requirement for the model files is to know the name of the inputs and outputs, which can be done with tools such as Netron.[73]

[69] https://mtg.github.io/essentia.js/docs/api/
[70] https://www.tensorflow.org/js
[71] https://github.com/microsoft/onnxjs
[72] https://github.com/onnx/onnx-tensorflow
[73] https://netron.app/

```
// import essentia.js-model and tensorflow.js
import {tensorflowmusicnn} from './essentia.js-model.es.js';
import * as tf from "@tensorflow/tfjs";

// path where the tfjs models are stored
const modelurl = "./autotagging/msd/msd-musicnn-1/model.json";

// create an instance of essentiatensorflowjsmodel
const musicnn = new tensorflowmusicnn(tf, modelurl);

// promise for initializing the model
await musicnn.initialize();

// run model inference on the given feature input
let prediction = await musicnn.predict(inputfeature);
```

**Figure 8.3:** Inference example with musicnn-based models using *essentia.js-model* via es6 style imports.

In the frozen format, the topology and weights are contained in a single binary file, TensorFlow.js models are defined in two files: a human-readable JSON file containing the topology and a binary file with the model weights. None of the weight quantization options offered by the converter were applied. the models are approximately the same size after conversion.

We compared the activations generated by both the original and the converted models finding minimal numerical differences in the range of $1 \times 10^{-5}$. We have also seen similar differences when testing the original models under different computer architectures or TensorFlow versions. After a further inspection of prediction outcomes, we conclude that they are too small to alter the sense of the predictions.

Essentia.js is available under AGPL-3.0 license and can be installed via NPM.[74] Last year, Essentia.js was downloaded over 500,000 times from NPM, and the repository has over 630 stars on GitHub. All the converted models are available for download on the Essentia models site.[75] They can be used for inference on a wide variety of devices, similar to TensorFlow.js. Figure 8.3 shows a code snippet for inference with the MusiCNN model using essentia.js to on the web.

[74] https://www.npmjs.com/package/essentia.js
[75] https://essentia.upf.edu/models

### 8.1.4 Essentia demos

Essentia's site has a dedicated section for web demos showcasing different capabilities of the library.[76] Many of these demos are powered by models developed in the context of this thesis. The demos include local notebooks intended for showcasing the real-time capabilities of our models, web-based demos relying on essentia.js and TensorFlow.js, web-based demos hosted on Replicate,[77] and integrations of our models on Hugging Face's model hub and their `transformers` library.[78]

- **Real-time simultaneous classification notebooks.** This demo showcases the real-time capabilities of our models by running classifiers based on deep representations detecting music danceability, voice activity, presence of aggressive mood, and presence of happy mood simultaneously. As mentioned in Section 1.2.1, the representation learning paradigm allows leveraging a single representation model to feed several shallow models, which prevents the redundancy of having individual end-to-end models for each task, allowing for very efficient inference pipelines. This demo features the audio classification models based on the MusiCNN architecture (Pons & Serra, 2019a), presented in Section 3.3.2, and it is available as a Jupyter Notebook from Essentia's GitHub (https://github.com/MTG/essentia/blob/master/src/examples/python/tutorial_tensorflow_real-time_simultaneous_classifiers.ipynb).
- **Real-time music style classification on the browser.** This demo showcases that it is also possible to run some of the models created in the context of this thesis on the browser through essentia.js. and tensorflow.js. We selected the models trained on Discogs' labels from Section 3.3.3. While for most of our experiments, we relied on the EfficientNet architecture, we noticed a ResNet25 model provided better inference times on the web. Since the EfficientNet architecture was originally designed with inference in mind, it features fewer standard operations compared to the ResNet architectures (depth-wise separable convolutions and squeeze-and-excitation blocks vs. standard convolutions and residual blocks). Thus we speculate that the better performance of the ResNet25 model could be attributed to inefficiencies in the model conversion process (PyTorch to TensorFlow to TensorFlow.js) (https://essentia.upf.edu/essentiajs-discogs).
- **Offline classifiers based on representation learning on the browser.** The difference with the previous demo is that this one is intended for

[76] https://essentia.upf.edu/demos.html
[77] http://replicate.com
[78] https://huggingface.co/

offline use, and it showcases five downstream classifiers trained on the same representation model. The downstream classifiers predict music danceability and happy, sad, relaxed, and aggressive mood presence, and they were presented in Section 3.3.2 (https://mtg.github.io/essentia.js/examples/demos/mood-classifiers/).

- **Replicate models.** Replicate is an inference platform and model hub for machine learning models. It allows users to deploy models so that they can be run on the web, through Replicate's API, or downloaded for offline use. Several models developed in the context of this thesis are available on Replicate, including the new models built on top of the representation learning models we developed.

  First, we provide a demo of the DiscogsEffNet (Section 3.3.3) style classification model (https://replicate.com/mtg/effnet-discogs).

  Then, we created an equivalent demo for the MAEST model (Chapter 6) (https://replicate.com/mtg/maest).

  We provide specific demos for regression models trained on the DiscogsEffNet and MAEST representations. These demos include Arousal and Valence estimation (https://replicate.com/mtg/music-arousal-valence).

  And music approachability and engagement (https://replicate.com/mtg/music-approachability-engagement).

  Finally, the `music-classifiers` demo, incorportes runs all the downstream classifiers available for MusiCNN, VGGish, and the Effnet-Discogs models (https://replicate.com/mtg/music-classifiers).

  Altogether, these demos accumulate over 150,000 runs by replicate users.

- **Hugging Face integration.** Hugging Face is a popular platform for sharing and deploying machine learning models. We have integrated our MAEST (Chapter 6) into this platform, making them available from their popular `transformers`' library. To support reproducibility, all the configurations of MAEST are available for download, which includes models with receptive fields ranging from 5 to 30 seconds, models trained with different types of data initialization, and a version of the weights obtained by teacher-student training. All the models can be used for music style classification and embeddings extraction within the `transformers` library (https://huggingface.co/mtg-upf/) under CC BY-NC-SA 4.0 license.

  Overall, the MAEST models have been downloaded more than 2M times from Hugging Face.

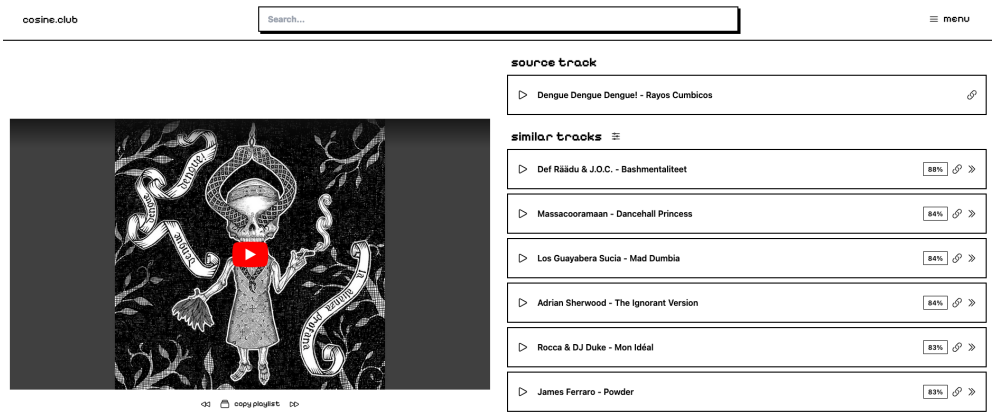


**Figure 8.4:** cosine.club website screenshot. Given a query track, the interface returns the closest tracks in the database based on the cosine distance.

## 8.2 Examples of applications by third parties

In the context of this thesis, we have made an effort to make the models developed available to the community through different means. This includes the integration of the models into the Essentia library, the development of Essentia-TensorFlow, the creation of the Essentia models hub, the integration of the models on Essentia.js, and the creation of web demos showcasing the models on replicate and Hugging Face. In all these cases, the models are openly available under a CC BY-NC-SA 4.0 license. In this section, we present examples of third parties that have integrated or experimented with some of our models for different applications.

### 8.2.1 cosine.club

cosine.club[79] is an electronic music discovery platform powered by representation learning models developed in the context of this thesis. Particularly, it utilizes the contrastive learning models based on editorial metadata (artist associations) presented in Chapter 4 to extract embeddings for a large dataset of electronic music tracks. When a query track is input to cosine.club, its embedding is computed and compared against the database of pre-computed embeddings using cosine distance. Afterward, the closest tracks are presented to the user. Figure 8.4 shows a screenshot of a query with the returned tracks on the cosine.club website.

The original goal of cosine.club was to enable discovery of tracks that were not available on digital platforms such as Spotify. The owners of cosine.club stated that the platform's database is focused on genres like House, Techno,

[79] https://cosine.club/

| Type | Sample size ↑ | All tags ↑ | Some tags ↑ | No tags ↓ |
|---|---|---|---|---|
| Baseline 1 | 308 | 14% | 51% | 36% |
| Baseline 2 | 313 | 22% | 50% | 28% |
| DiscogsEffNet | 18,704 | 18% | 52% | 29% |
| MAEST | **25,626** | **26%** | **58%** | **16%** |

**Table 8.1:** Comparison of user preferences for the different music stlye labelling systems tested by SubmitHub. The sable shows the percentage of users that selected all, some, or no tags when using the different systems.

Funk/Soul, Rare Groove, etc., and not much on mainstream or popular music. Their database is sourced from Discogs, and they plan to continuously update it as more releases are added.

### 8.2.2 SubmitHub

SubmitHub[80] is a music promotion website service where artists submit their music so that it gets distributed to curators, Spotify playlisters, music bloggers, and influencers that match the artist's style and context. The curators can then listen to the music to provide feedback and, optionally, add it to their playlists or promote it on their channels. As of today, SubmitHub has over 10,000 curators and has processed over 38 submissions by 1 million artists.

To match an artist with the relevant curators, SubmitHub relies on the style labels, metadata, and description of the track provided by the artists. One of the limitations of this process is the lack of accuracy when the artists choose the style labels for their music, which hinders the process of finding appropriate curators. To overcome this, SubmitHub experimented with the integration of our DiscogsEffnet (Section 3.3.3) and MAEST (Chapter 6) models to automatically suggest style labels to the users. [81]

Table 8.1 shows a comparison of music-style labeling systems tested by SubmitHub. While the experimental conditions were not disclosed by the company, and the sample size values vary significantly across models, the results suggest that MAEST could outperform other systems in terms of user preferences.

[80] https://www.submithub.com/

[81] This test was informally conducted by SubmitHub and the results were shared with the author. As for now, none of the models developed in this thesis is being used in production by SubmitHub.

CHAPTER 9

# Conclusions and future perspectives

## 9.1 Introduction

Throughout this dissertation, we have explored different strategies to create music representations models able to improve the performance of certain music classification tasks. We have experimented with different sources of supervision based on music metadata publicly available on the internet, and focused on track-level multi-label downstream classification tasks. To pre-train our models, we have collected a large pool of 4 million tracks matched to Discogs metadata, and for our downstream models, we have used a combination of in-house and public datasets covering the high-level music description tasks such as genre recognition, mood and theme detection, and instrument recognition. Through these experiments, we can now draw some insights related to the research questions posed in Section 1.4. After this, we will summarize the main contributions of this thesis, and discuss some open issues and future perspectives.

- **Q1**. Our first research question is related to the benefits of using pre-trained representation models instead of solving the downstream tasks from scratch. In Chapter 3, we experiment with a number of pre-trained models to address this question. We found that relying on pre-trained models can significantly improve the performance of the downstream tasks, especially when the amount of downstream data is limited. Additionally, our cross-collection evaluation suggests that the resulting models tend to be more robust and generalize better to unseen data.

- **Q2**. Our second research question is related to the type of supervision providing the best performance. In Chapters 4 and 5, we explored

the use of editorial and consumption metadata as supervision signals for pre-training music representation models. In both cases, we found that models trained with contrastive learning approaches learning to attract tracks associated with metadata notions tend to outperform models trained with classification-based targets under the same data and parameter count conditions. Nevertheless, the best overall performance on most of the downstream tasks was obtained with the MAEST model, which was pre-trained to classify music style labels from Discogs. An important observation is that MAEST relies on a transformer architecture with a much higher parameter count, and relied on a number of regularization techniques to be trained efficiently. While we could especulate that training a MAEST model with a contrastive learning loss and editorial metadata could improve the performance of the model, we did not have the computational resources to test this hypothesis. This observation highlights one of the main benefits of label-based classification. While other approaches can provide better performance, it results very efficient compared to metric learning or other self-supervised approaches. Following this idea, the self-supervised approaches considered in this thesis could not compete with the metadata or label-based approaches in terms of performance at the parameter counts, compute resources, and data sizes considered in this thesis.

- **Q3**. The third research question is related to the best architecture to use for music representation learning. In Chapter 6, we explored the use of transformers as feature extractors for music classification tasks. We found that making transformers work for music classification tasks is not straightforward, and it requires considerable engineering tricks to be able to train efficiently. In particular, we only achieved good performance when combining weights pre-training on other tasks with strong regularization techniques such as mix-up data augmentation, and patch-out (learning from partial input sequences). By training the transformers this way, we found that it is possible to perform inference from partial input sequences, which allows for reducing the computational cost of the model, achieving better downstream performance/throughput trade-offs than their convolutional counterparts.

- **Q4**. Our last research question is related to the interpretability of the models resulting from pre-trained representation models. In Chapter 7, we show that it is possible to train interpretable audio classifiers based on pre-trained representation models, allowing to gain insights on the decision-making process of the model.

## 9.2 Summary of contributions

With this dissertation, we hope to have contributed to the advancement of the state of the art in the field of music representation learning through different contributions. Apart from the research publications, most of the relevant models developed in the context of this thesis have been published, including training code, weights, and different inference options, including Essentia (C++/Python) as well as implementations of particular models on online APIs such as Replicate.ai, and Hugging Face. With this effort, the author aims to provide the community with tools to facilitate new research in MIR as well as others intersecting fields.

Additionally, our models have the potential to be used in real-world applications beyond the academic scope. For example, the DiscogsEffNet representation model is used in consine.club, an online electronic music similarity search engine, and the MAEST models count with more than 2 million downloads in the Hugging Face model hub.

The main contributions of this thesis are summarized as follows:

- The first contribution of this thesis is the creation of Essentia models, a hub of deep learning models for audio and music analysis integrated with the Essentia Library (Chapter 8.1.2). The original publication also describes the implementation of the C++ algorithms to support using the models in the Essentia framework efficiently, along with an evaluation of the preliminary models available at the time (Alonso-Jiménez et al., 2020a). Subsequent publications serve to extend the models available in the hub, including alternative versions of the existing classifiers with enhanced performance powered by updated representation models.

- We proposed DiscogsEffNet, a convolutional neural network trained on a 3.3 million collection of audio and music style tags coming from Discogs, which achieved state-of-the-art performance in several downstream tasks (Section 3.3.3). DiscogsEffNet is also integrated as part of Essentia models, and available on Replicate.ai for demonstration purposes. [82] DiscogsEffNet and the downstream models based on its representations accumulate over 150,000 runs in this platform.

- We conducted a comprehensive study on the performance of different representations learning models, including DiscogsEffNet, for several music classification tasks with small datasets. The study analyzed different aspects such as the performance of the models on low-data regimes (Section 3.3.5), the effect of combining embeddings from different models to

[82] https://replicate.ai/MTG/effnet-discogs

solve the tasks (Section 3.3.6), and the generalization capabilities of the models to unseen data (Section 3.3.7), and the impact of the pre-training dataset size on the downstream performance (Section 3.3.9).

- In Chapter 4, we proposed a novel approach to pre-train music representation learning model using editorial metadata from Discogs and contrastive learning. The experimental results showed that the proposed approach outperforms the baseline models trained with classification based targets in several music classification tasks, including genre classification, and mood classification. We isolated the behavior of training the models using different metadata notions considering album, artist, and record label properties, finding that the artist metadata provides the features with the best downstream performance.

- Inspired with the success of the metadata-based pre-training methods, we extended this approach to supervision by consumption music metadata. In our experiments we focus on music playlist information, and proposed different strategies to train the models using this data. We obtained the best results by modeling the playlist information with a Word2Vec model, and training a contrastive learning model to align the embeddings of the playlist and the audio features. The results achieved state-of-the-art in the MagnaTagATune (MTAT) dataset, and in general achieved better performance than our previous apprach using artist names as the supervision signal.

- We proposed the first study on the usage of pure self-attention-based transformers as feature extractors for music classification tasks (Chapter6. In this works, we found a number of direction in which transformers behave differently from convolutional neural networks. First they are much more dependent on pre-training. Opposite to convolutional neural networks, transformers can benefit from longer input sequences. Finally, the best representations are obtained from the middle layers instead of the last ones as in the convolutional neural networks. We show that it is possible to train transformers efficiently on supervised tasks by presenting only parts of the input sequence (path out), and the same technique can be used on inference time to reduce the computational cost of the model, achieving better downstream performance/throughput trade-offs than their convolutional counterparts.

- Finally, we proposed a strategy that allows training interpretable audio classifiers based on pre-trained representation models (Chapter 7). Our approach updates the APNet model (Zinemanas et al., 2021), which jointly trains an audio autoencoder and a classifier based on prototypical learning. Our approach, relies on a pre-trained representation model

instead of training the autoencoder, which allows for using additional data to enhance the performance of the model. For the sonification of the prototypes, we use a difussion-based transformer conditioned on the desired features. By decoupling the training of the representation model and the classifier, we can obtain better performance in the downstream tasks, create systems with more prototypes, or train classification models with less data.

## 9.3 Open issues and future perspectives

To conclude this dissertation, we will discuss some open issues and that remain to be addressed in the field of music representation learning and future perspectives that could be explored in the future.

### 9.3.1 Self-supervised learning perspective

The recent success of AI in the field of NLP has been driven by the development of self-supervised learning techniques, specially thanks to autoregressive token prediction models such as BERT (Devlin et al., 2018) and GPT (Radford et al., 2018). These techniques exploit the sparse nature of natural language by approaching token prediction task as a classification problem, which is not directly possible in continuous domains such as image or audio.

Nevertheless, great advances have been made in the field of self-supervised learning for these domains in recent years. In image, speech, and audio, the combination of self-supervised pre-training with fine-tuning on supervised tasks is the de facto standard, and the trend suggest that self-supervised approaches will continue to be the main driver of progress in the field.

In this thesis, we have experimented with self-supervised techniques derived from SimCLR, the most powerful for music audio representation learning approach according to the benchmarks of Wang et al. (2022) and Meseguer-Brocal et al. (2024). We have shown that adapting this approach to use music metadata information to form the positive pairs, resulted in better representations for the considered downstream tasks. However, it is still an open question if future self-supervise approaches would benefit from the same metadata information, or if the models could learn to extract the relevant information from the audio itself.

### 9.3.2 Multitask learning

The metadata information considered in this thesis opened the door to multitask learning approaches. Although the intuition of the authors is that learning

from multiple sources could benefit the quality of the learned features, the multitask experiments conducted in Section 4.3.6 did not show a clear advantage of this approach. It is important to mention that these experiments could not be fairly compared with its single-task counterparts, as the limited amount to GPU ram forced us to reduce the effective batch size per task. Contrastive learning approaches are known to be sensitive to the batch size, and the results could be affected by this limitation. Because of this, we could not draw a clear conclusion on the benefits of multitask learning in the context of music representation learning and the proposed metadata sources.

### 9.3.3 Multi-modal learning

Multi-modal learning is a promising research direction that could benefit the field of music representation learning. In the area of audio scene recognition, audio-video models have been able to outperform their audio-only counterparts recently (Gong et al., 2023). While the usage of the video modality would be a direct extension of the audio modality, the most direct way of doing this is through video clips, where the audio and video are not necessarily synchronized and the benefits have yet to be assessed. Additionally, other modalities that could be useful include lyrics, music scores, or even album covers.

In this dissertation, we proposed one representation learning approach that treats playlist information as a different modality, and it was shown to provide the best performance in the downstream evaluation (Section 5.3). This result suggests that multi-modal learning could be a promising research direction in the field of music representation learning.

### 9.3.4 Representation learning and time resolution

As mentioned in Section 6.4, common NLP self-supervised representation learning are able to learn useful information with token-level resolution. However, all proposed supervision approaches investigated in this thesis are based on track-level supervision. Consequently, all the considered evaluation tasks consider track-level labels, and since all the considered architectures operate with receptive fields shorter than most tracks, track-level metrics are derived by statistical aggregation of the chunk level predictions. While this is a common practice in the field, it can be regarded as a limitation for the development of more nuanced music understanding systems.

Approaches based on masked token prediction, such as (Huang et al., 2022a), are a direct way to address this issue. These approaches, can be combined with a discrete quantization methods such as K-Nearest Neighbors (KNN), or random codebook projection (Yizhi et al., 2023; Won et al., 2024) allowing

to obtain competitive results both in track-level and high temporal resolution downstream music understanding tasks.

Investigating if these new models could benefit from the metadata information considered in this thesis remain an open question. Connecting this idea with Section 9.3.3, we could also consider if overall, metadata-driven objectives could be combines with token-level reconstruction objectives to obtain better representations for music understanding tasks.

### 9.3.5 Music understanding evaluation

As mentioned in Section 9.3.4, advancing the field of music understanding requires the development of more sophisticated evaluation tasks and metrics. In Section 3.2.3, we propose a cross-collection evaluation, methodology to assess the generalization capabilities of the considered models. While this approach supposes a step into expanding the approaches to evaluate representation models, it is still limited and difficult to scale.

### 9.3.6 Interpretability in representation learning

Finally, we address the issue of interpretability in representation learning. While the majority of DNN-based systems operate as black-box models, those relying on data-driven representations could be considered even more opaque than their hand-crafted counterparts. In this thesis, we proposed a strategy to train interpretable audio classifiers based on pre-trained representation models (Chapter 7). However, we did not assess the influence of different representations in the sonifications of the prototypes, and for now, we can only speculate that this approach could be used to obtain understanding of the information encoded in the representations.

Pablo Alonso Jiménez.
Barcelona, July 17th, 2024.

# Bibliography


AI HLEG (2019). Ethics Guidelines for Trustworthy AI. [Cited on page 97.]

Alonso-Jiménez, P., Bogdanov, D., Pons, J., & Serra, X. (2020a). Tensorflow audio models in Essentia. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 266–270. [Cited on pages 3, 8, 11, 19, 30, and 119.]

Alonso-Jiménez, P., Bogdanov, D., & Serra, X. (2020b). Deep embeddings with Essentia models. *International Society for Music Information Retrieval Conference (ISMIR) Late-Braeking/Demo*. [Cited on pages 11, 20, 30, and 71.]

Alonso-Jiménez, P., Favory, X., Foroughmand, H., Bourdalas, G., Serra, X., Lidy, T., & Bogdanov, D. (2023a). Pre-training strategies using contrastive learning and playlist information for music classification and similarity. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 11, 72, and 84.]

Alonso-Jiménez, P., Pepino, L., Batlle-Roca, R., Zinemanas, P., Bogdanov, D., Serra, X., & Rocamora, M. (2024). Leveraging pre-trained autoencoders for interpretable prototype learning of music audio. In *ICASSP Workshop on Explainable AI for Speech and Audio (XAI-SA)*. [Cited on pages 11 and 98.]

Alonso-Jiménez, P., Serra, X., & Bogdanov, D. (2023b). Efficient supervised training of audio transformers for music representation learning. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 3, 5, 11, 86, 102, and 147.]

Alonso-Jiménez, P., Serra, X., & Dmitry, B. (2022). Music representation learning based on editorial metadata from Discogs. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 11, 59, 77, 85, 88, 93, and 94.]

Andén, J., Lostanlen, V., & Mallat, S. (2019). Joint time–frequency scattering. *IEEE Transactions on Signal Processing*. [Cited on pages 102 and 103.]

Andrade, N. & Figueiredo, F. (2016). Exploring the latent structure of collaborations in music recordings: A case study in jazz. In *ISMIR*. [Cited on page 25.]

Arandjelovic, R. & Zisserman, A. (2017). Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 609–617. [Cited on page 20.]

Arrieta, A. B., Díaz-Rodríguez, N., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*. [Cited on page 97.]

Batlle-Roca, R., Gómez, E., Liao, W., Serra, X., & Mitsufuji, Y. (2023). Transparency in music-generative AI: A systematic literature review. *preprint (under reivew)*. [Cited on page 27.]

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. [Cited on pages 2 and 4.]

Berenzweig, A. L., Ellis, D. P., & Lawrence, S. (2002). Using voice segments to improve artist classification of music. In *Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*. Audio Engineering Society. [Cited on page 2.]

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., & Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 6, 15, 17, and 18.]

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., & Widmer, G. (2016). Madmom: A new Python audio and music signal processing library. In *ACM International Conference on Multimedia (ACMMM)*, pp. 1174–1178. [Cited on page 106.]

Böck, S. & Schedl, M. (2011). Enhanced beat tracking with context-aware neural networks. In *Proc. Int. Conference Digital Audio Effects*, pp. 135–139. [Cited on page 16.]

Bogdanov, D., Haro, M., Fuhrmann, F., Xambó, A., Gómez, E., & Herrera, P. (2013a). Semantic audio content-based music recommendation and visualization based on user preference examples. *Information Processing & Management*, *49*(1), 13–33. [Cited on page 8.]

Bogdanov, D. & Herrera, P. (2012). Taking advantage of editorial metadata to recommend music. In *International Symposium on Computer Music Modeling and Retrieval (CMMR)*. [Cited on page 25.]

Bogdanov, D., Lizarraga-Seijas, X., Alonso-Jiménez, P., & Serra, X. (2022). Musav: A dataset of relative arousal-valence annotations for validation of audio models. In *International Society for Music Information Retrieval Conference*. [Not cited.]

Bogdanov, D., Porter, A., Boyer, H., Serra, X. et al. (2016). Cross-collection evaluation for music classification tasks. In *Devaney J, Mandel MI, Turnbull D, Tzanetakis G, editors. ISMIR 2016. Proceedings of the 17th International Society for Music Information Retrieval Conference; 2016 Aug 7-11; New York City (NY).[Canada]: ISMIR; 2016. p. 379-85.* International Society for Music Information Retrieval (ISMIR). [Cited on pages 29, 31, and 49.]

Bogdanov, D., Porter, A., Schreiber, H., Urbano, J., & Oramas, S. (2019a). The acousticbrainz genre dataset: Multi-source, multi-level, multi-label, and large-scale. In *Proceedings of the 20th International Society for Music Information Retrieval (ISMIR)*. [Cited on page 25.]

Bogdanov, D. & Serra, X. (2017). Quantifying music tends and facts using editorial metadata from the Discogs database. In *International Society for Music Information Retrieval Conference (ISMIR)*, p. 7. [Cited on pages 25 and 40.]

Bogdanov, D., Serrà, J., Wack, N., Herrera, P., & Serra, X. (2011). Unifying low-level and high-level music similarity measures. *IEEE Transactions on Multimedia*, *13*(4), 687–701. [Cited on page 8.]

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J. R., & Serra, X. (2013b). Essentia: an audio analysis library for music information retrieval. In *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 493–498. [Cited on page 7.]

Bogdanov, D., Won, M., Tovstogan, P., Porter, A., & Serra, X. (2019b). The MTG-Jamendo dataset for automatic music tagging. In *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML)*. [Cited on pages 34, 49, 63, and 76.]

Bogdanov, D., Won, M., Tovstogan, P., Porter, A., & Serra, X. (2019c). The MTG-Jamendo dataset for automatic music tagging. In *International Conference on Machine Learning (ICML)*. [Cited on page 89.]

Bour, V. (2021). Frequency dependent convolutions for music tagging. In *MediaEval 2021 Multimedia Benchmark Workshop*. [Cited on page 52.]

Budner, P. & Grahl, J. (2016). Collaboration networks in the music industry. *arXiv preprint arXiv:1611.00377*. [Cited on page 25.]

Buisson, M., Alonso-Jiménez, P., & Bogdanov, D. (2022). Ambiguity modelling with label distribution learning for music classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Not cited.]

Burke, J., Rygaard, R., & Yellin-Flaherty, Z. (2014). CLAM!: Inferring genres in the Discogs collaboration network. [Cited on page 25.]

Castellon, R., Donahue, C., & Liang, P. (2021). Codified audio language modeling learns useful representations for music information retrieval. *arXiv preprint arXiv:2107.05677*. [Cited on pages 20, 30, 41, 65, 66, and 90.]

Chen, C.-W., Lamere, P., Schedl, M., & Zamani, H. (2018). RecSys challenge 2018: Automatic music playlist continuation. In *ACM Conference on Recommender Systems*. [Cited on pages 12, 24, 71, 73, and 75.]

Chen, S., Wu, Y., Wang, C., Liu, S., Tompkins, D., Chen, Z., & Wei, F. (2022). BEATs: Audio pre-training with acoustic tokenizers. *arXiv preprint arXiv:2212.09058*. [Cited on page 26.]

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR. [Cited on pages 4, 12, 21, 74, and 75.]

Choi, K., Fazekas, G., Cho, K., & Sandler, M. (2018). The effects of noisy labels on deep convolutional neural networks for music tagging. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *2*, 139–149. [Cited on page 17.]

Choi, K., Fazekas, G., & Sandler, M. (2016). Automatic tagging using deep convolutional neural networks. *arXiv preprint arXiv:1606.00298*. [Cited on pages 17 and 34.]

Choi, K., Fazekas, G., Sandler, M., & Cho, K. (2017a). Convolutional recurrent neural networks for music classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2392–2396. [Cited on page 17.]

Choi, K., Fazekas, G., Sandler, M., & Cho, K. (2017b). Transfer learning for music classification and regression tasks. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 18, 30, and 92.]

Chong, D., Wang, H., Zhou, P., & Zeng, Q. (2022). Masked spectrogram prediction for self-supervised audio pre-training. *arXiv preprint arXiv:2204.12768*. [Cited on pages 4, 22, and 26.]

Correya, A., Alonso-Jiménez, P., Marcos-Fernández, J., Serra, X., & Bogdanov, D. (2021a). Essentia tensorflow models for audio and music processing on the web. In *Web Audio Conference (WAC)*. [Cited on pages 110 and 147.]

Correya, A. A., Bogdanov, D., Alonso Jiménez, P., & Serra, X. (2022). Essentia API: a web API for music audio analysis. *International Society for Music Information Retrieval Conference (ISMIR) Late-Braeking/Demo*. [Not cited.]

Correya, A. A., Bogdanov, D., Joglar-Ongay, L., & Serra, X. (2020). Essentia.js: a javascript library for music and audio analysis on the web. In *Proceedings of the 21st International Society for Music Information Retrieval Conference*. [Cited on page 109.]

Correya, A. A., Marcos-Fernández, J., Joglar-Ongay, L., Alonso-Jiménez, P., Serra, X., & Bogdanov, D. (2021b). Audio and music analysis on the web using essentia.js. *Transactions of the International Society of Music Information Retrieval*, *4*, 167–181. [Cited on page 110.]

Cramer, J., Wu, H.-H., Salamon, J., & Bello, J. P. (2019). Look, listen, and learn more: Design choices for deep audio embeddings. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3852–3856. IEEE. [Cited on pages 20 and 37.]

Dannenberg, R. B., Thom, B., & Watson, D. (1997). A machine learning approach to musical style recognition. [Cited on page 16.]

Defferrard, M., Benzi, K., Vandergheynst, P., & Bresson, X. (2016). FMA: A dataset for music analysis. *arXiv:1612.01840v3 [cs.SD]*. [Cited on page 63.]

Defferrard, M., Benzi, K., Vandergheynst, P., & Bresson, X. (2017). FMA: A dataset for music analysis. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on page 101.]

Défossez, A., Copet, J., Synnaeve, G., & Adi, Y. (2022). High fidelity neural audio compression. *preprint arXiv:2210.13438*. [Cited on page 22.]

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. [Cited on page 121.]

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., & Sutskever, I. (2020). Jukebox: A generative model for music. *arXiv:2005.00341v1 [eess.AS]*. [Cited on pages 20 and 41.]

Dieleman, S. & Schrauwen, B. (2014a). End-to-end learning for music audio. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6964–6968. IEEE. [Cited on page 17.]

Dieleman, S. & Schrauwen, B. (2014b). End-to-end learning for music audio. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on page 27.]

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations (ICLR)*. [Cited on pages 26, 85, and 86.]

Eck, D., Lamere, P., Bertin-Mahieux, T., & Green, S. (2007). Automatic generation of social tags for music recommendation. *Advances in neural information processing systems*, *20*, 385–392. [Cited on page 17.]

Elizalde, B., Deshmukh, S., Al Ismail, M., & Wang, H. (2023). CLAP learning audio concepts from natural language supervision. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. [Cited on page 22.]

Favory, X., Drossos, K., Virtanen, T., & Serra, X. (2020). COALA: Co-aligned autoencoders for learning semantically enriched audio representations. *ArXiv*, *abs/2006.08386*. [Cited on pages 22 and 24.]

Favory, X., Drossos, K., Virtanen, T., & Serra, X. (2021). Learning contextual tag embeddings for cross-modal alignment of audio and tags. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 22 and 24.]

Feichtenhofer, C., Fan, H., Malik, J., & He, K. (2019). Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*. [Cited on page 21.]

Feng, Y., Zhuang, Y., & Pan, Y. (2003). Popular music retrieval by detecting mood. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 375–376. [Cited on page 2.]

Ferraro, A., Favory, X., Drossos, K., Kim, Y., & Bogdanov, D. (2021). Enriched music representations with multiple cross-modal contrastive learning. *IEEE Signal Processing Letters*. [Cited on pages 22, 24, and 71.]

Fonseca, E., Favory, X., Pons, J., Font, F., & Serra, X. (2021). Fsd50k: an open dataset of human-labeled sound events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *30*, 829–852. [Cited on page 22.]

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, *14*(771-780), 1612. [Cited on page 17.]

Fricke, K. R., Greenberg, D. M., Rentfrow, P. J., & Herzberg, P. Y. (2018). Computer-based music feature analysis mirrors human perception and can be used to measure individual music preference. *Journal of Research in Personality*, *75*, 94–102. [Cited on page 8.]

Fu, Z., Lu, G., Ting, K. M., & Zhang, D. (2010). A survey of audio-based music classification and annotation. *IEEE transactions on multimedia*, *13*(2), 303–319. [Cited on page 15.]

Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., & Ritter, M. (2017). Audio Set: An ontology and human-labeled dataset for audio events. In *International Conference on acoustics, speech and signal processing (ICASSP)*. [Cited on pages 22, 34, and 85.]

Gienapp, L., Kruckenberg, C., & Burghardt, M. (2021). Topological properties of music collaboration networks: The case of jazz and hip hop. *DHQ: Digital Humanities Quarterly*. [Cited on page 25.]

Gómez Cañón, J. S., Cano, E., Boyer, H., Gómez Gutiérrez, E. et al. (2020). Joyful for you and tender for us: the influence of individual characteristics and language on emotion labeling and classification. In *Cumming J, Ha Lee J, McFee B, Schedl M, Devaney J, McKay C, Zagerle E, de Reuse T, editors. Proceedings of the 21st International Society for Music Information Retrieval Conference; 2020 Oct 11-16; Montréal, Canada.[Canada]: ISMIR; 2020.* International Society for Music Information Retrieval (ISMIR). [Cited on page 8.]

Gong, Y., Chung, Y., & Glass, J. R. (2021). AST: audio spectrogram transformer. In *22nd Annual Conference of the Intn. Speech Communication Association (Interspeech)*. [Cited on pages 26, 85, 86, 87, and 91.]

Gong, Y., Rouditchenko, A., Liu, A. H., Harwath, D., Karlinsky, L., Kuehne, H., & Glass, J. R. (2023). Contrastive audio-visual masked autoencoder. In *The Eleventh International Conference on Learning Representations*. [Cited on page 122.]

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M. et al. (2020).

Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*. [Cited on page 21.]

Grollmisch, S., Cano, E., Kehling, C., & Taenzer, M. (2021). Analyzing the potential of pre-trained embeddings for audio classification tasks. *2020 28th European Signal Processing Conference (EUSIPCO)*, pp. 790–794. [Cited on page 38.]

Hamel, P., Davies, M., Yoshii, K., & Goto, M. (2013). Transfer learning in MIR: Sharing learned latent representations for music audio classification and similarity. In *ISMIR*. [Cited on page 18.]

Hamel, P. & Eck, D. (2010). Learning features from music audio with deep belief networks. In *International Society for Music Information Retrieval Conference (ISMIR)*, vol. 10, pp. 339–344. [Cited on page 18.]

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., & Girshick, R. (2022). Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009. [Cited on page 4.]

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [Cited on pages 4 and 21.]

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. [Cited on pages 40, 73, and 74.]

Hennequin, R., Khlif, A., Voituret, F., & Moussallam, M. (2020). Spleeter: a fast and efficient music source separation tool with pre-trained models. *Journal of Open Source Software*, *5*(50), 2154. [Cited on page 38.]

Hennequin, R., Royo-Letelier, J., & Moussallam, M. (2018). Audio Based Disambiguation Of Music Genre Tags. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 25 and 40.]

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B. et al. (2017). Cnn architectures for large-scale audio classification. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 131–135. IEEE. [Cited on pages 19, 34, 65, 73, 74, and 77.]

Hillecke, T., Nickel, A., & Bolay, H. V. (2005). Scientific perspectives on music therapy. *Annals of the New York Academy of Sciences*, *1060*(1), 271–282. [Cited on page 2.]

Huang, P.-Y., Xu, H., Li, J., Baevski, A., Auli, M., Galuba, W., Metze, F., & Feichtenhofer, C. (2022a). Masked autoencoders that listen. In *NeurIPS*. [Cited on page 122.]

Huang, Q., Jansen, A., Lee, J., Ganti, R., Li, J. Y., & Ellis, D. P. (2022b). MuLan: A joint embedding of music audio and natural language. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 27, 85, 93, and 94.]

Huang, Q., Jansen, A., Zhang, L., Ellis, D. P., Saurous, R. A., & Anderson, J. (2020). Large-scale weakly-supervised content embeddings for music recommendation and tagging. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8364–8368. [Cited on pages 19, 20, 24, 71, and 85.]

Hung, Y.-N., Wang, J.-C., Song, X., Lu, W.-T., & Won, M. (2022). Modeling beats and downbeats with a time-frequency transformer. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 401–405. [Cited on page 5.]

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., & Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*. [Cited on page 62.]

Joorabchi, A. & Mahdi, A. E. (2011). An unsupervised approach to automatic classification of scientific literature utilizing bibliographic metadata. *Journal of Information Science*. [Cited on page 23.]

Kaminsky, I. & Materka, A. (1995). Automatic source identification of monophonic musical instrument sounds. In *Proceedings of the International Conference on Neural Networks*, pp. 189–194 vol.1. [Cited on page 2.]

Kereliuk, C., Sturm, B. L., & Larsen, J. (2015). Deep learning and music adversaries. *IEEE Transactions on Multimedia*, *17*(11), 2059–2071. [Cited on page 52.]

Kim, J., Urbano, J., Liem, C. C., & Hanjalic, A. (2020a). One deep music representation to rule them all? a comparative analysis of different representation learning strategies. *Neural Computing and Applications*, *32*(4), 1067–1093. [Cited on page 19.]

Kim, J., Urbano, J., Liem, C. C. S., & Hanjalic, A. (2019a). Are nearby neighbors relatives?: Diagnosing deep music embedding spaces. *ArXiv*, *abs/1904.07154*. [Cited on page 19.]

Kim, J., Urbano, J., Liem, C. C. S., & Hanjalic, A. (2019b). One deep music representation to rule them all? a comparative analysis of different representation learning strategies. *Neural Computing and Applications*, *32*, 1067–1093. [Cited on page 20.]

Kim, J., Urbano, J., Liem, C. C. S., & Hanjalic, A. (2020b). One deep music representation to rule them all? a comparative analysis of different representation learning strategies. *Neural Computing and Applications*. [Cited on pages 23, 24, and 71.]

Kim, J., Won, M., Serra, X., & Liem, C. C. S. (2018a). Transfer learning of artist group factors to musical genre classification. *Companion Proceedings of the Web Conference 2018*. [Cited on pages 7, 23, 24, 71, and 77.]

Kim, J. W., Salamon, J., Li, P., & Bello, J. P. (2018b). Crepe: A convolutional representation for pitch estimation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 161–165. IEEE. [Cited on pages 2 and 16.]

Kim, T., Lee, J., & Nam, J. (2018c). Sample-level CNN architectures for music auto-tagging using raw waveforms. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on page 103.]

Knox, D., Greer, T., Ma, B., Kuo, E., Somandepalli, K., & Narayanan, S. (2020). Mediaeval 2020 emotion and theme recognition in music task: Loss function approaches for multi-label music tagging. In *MediaEval 2020 Multimedia Benchmark Workshop*. [Cited on pages 93 and 94.]

Koh, E. S. & Dubnov, S. (2021). Comparison and analysis of deep audio embeddings for music emotion recognition. *CoRR*. [Cited on page 19.]

Korzeniowski, F. & Widmer, G. (2018). Genre-agnostic key classification with convolutional neural networks. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on page 5.]

Koutini, K., Schlüter, J., Eghbal-zadeh, H., & Widmer, G. (2022). Efficient training of audio transformers with patchout. In *23rd Annual Conference of the International Speech Communication Association (Interspeech)*. [Cited on pages 26, 85, 86, 88, and 91.]

Kumar, R., Seetharaman, P., Luebs, A., Kumar, I., & Kumar, K. (2023). High-fidelity audio compression with improved RVQGAN. *preprint arXiv:2306.06546*. [Cited on page 22.]

Laden, B. & Keefe, D. H. (1989). The representation of pitch in a neural net model of chord classification. *Computer Music Journal*, *13*(4), 12–26. [Cited on pages 2, 15, and 16.]

Laurier, C. (2011). *Automatic Classification of Musical Mood by Content-Based Analysis*. Ph.D. thesis, Universitat Pompeu Fabra. [Cited on page 8.]

Law, E., West, K., Mandel, M. I., Bay, M., & Downie, J. S. (2009). Evaluation of algorithms using games: The case of music tagging. In *ISMIR*, pp. 387–392. Citeseer. [Cited on pages 12, 63, and 76.]

Lee, J., Bryan, N. J., Salamon, J., Jin, Z., & Nam, J. (2020a). Disentangled multidimensional metric learning for music similarity. In *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 76 and 77.]

Lee, J., Bryan, N. J., Salamon, J., Jin, Z., & Nam, J. (2020b). Metric learning vs classification for disentangled music representation learning. In *ISMIR*. [Cited on page 19.]

Lee, J., Park, J., Kim, K., & Nam, J. (2018). Samplecnn: End-to-end deep convolutional neural networks using very small filters for music classification. *Applied Sciences*, *8*(1), 150. [Cited on pages 19, 52, and 89.]

Lee, J., Park, J., & Nam, J. (2019a). Representation learning of music using artist, album, and track information. *ArXiv*, *abs/1906.11783*. [Cited on page 23.]

Lee, J., Park, J., & Nam, J. (2019b). Representation learning of music using artist, album, and track information. In *International Conference on Machine Learning (ICML) 2019, Machine Learning for Music Discovery Workshop*. [Cited on pages 24, 58, 59, 65, 71, and 77.]

Leung, J. K., Griva, I., & Kennedy, W. G. (2020). Making use of affective features from media content metadata for better movie recommendation making. *arXiv preprint arXiv:2007.00636*. [Cited on page 23.]

Levitin, D. J. & Tirovolas, A. K. (2009). Current advances in the cognitive neuroscience of music. *Annals of the New York Academy of Sciences*, *1156*(1), 211–231. [Cited on page 2.]

Li, O., Liu, H., Chen, C., & Rudin, C. (2018). Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *The 32nd AAAI Conference on Artificial Intelligence*. [Cited on pages 98 and 100.]

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. [Cited on pages 21 and 27.]

Loiseau, R., Bouvier, B., Teytaut, Y., Vincent, E., Aubry, M., & Landrieu, L. (2022). A model you can hear: Audio identification with playable prototypes. In *International Conference on Music Information Retrieval (ISMIR)*. [Cited on page 27.]

Lostanlen, V. & Cella, C.-E. (2016). Deep convolutional networks on the pitch spiral for musical instrument recognition. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on page 100.]

Manco, I., Benetos, E., Quinton, E., & Fazekas, G. (2021). Learning music audio representations via weak language supervision. *arXiv:2112.04214v1 [cs.SD]*. [Cited on pages 22, 63, 65, and 89.]

Marolt, M., Kavcic, A., & Privosnik, M. (2002). Neural networks for note onset detection in piano music. In *Proceedings of the 2002 International Computer Music Conference*. Citeseer. [Cited on pages 2 and 16.]

Matityaho, B. & Furst, M. (1995). Neural network based model for classification of music type. In *Eighteenth Convention of Electrical and Electronics Engineers in Israel*, pp. 4–3. IEEE. [Cited on pages 2, 15, and 16.]

McCallum, M. C., Korzeniowski, F., Oramas, S., Gouyon, F., & Ehmann, A. F. (2022). Supervised and unsupervised learning of audio representations for music understanding. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 6, 21, 85, 89, 93, and 94.]

Meseguer-Brocal, G., Desblancs, D., & Hennequin, R. (2024). An experimental comparison of multi-view self-supervised methods for music tagging. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1141–1145. [Cited on pages 21 and 121.]

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. [Cited on page 73.]

Mishra, S., Sturm, B. L., & Dixon, S. (2017). Local interpretable model-agnostic explanations for music content analysis. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on page 27.]

Moffat, D., Ronan, D., & Reiss, J. D. (2015). An evaluation of audio feature extraction toolboxes. In *International Conference on Digital Audio Effects (DAFx)*. [Cited on page 7.]

Niizumi, D., Takeuchi, D., Ohishi, Y., Harada, N., & Kashino, K. (2021). Byol for audio: Self-supervised learning for general-purpose audio representation. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. [Cited on page 21.]

Palanisamy, K., Singhania, D., & Yao, A. (2020). Rethinking cnn models for audio classification. *ArXiv, abs/2007.11154*. [Cited on page 19.]

Park, D. S., Chan, W., Zhang, Y., Chiu, C., Zoph, B., Cubuk, E. D., & Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)*. [Cited on page 88.]

Park, J., Lee, J., Ha, J.-W., & Nam, J. (2018). Representation learning of music using artist labels. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 7, 12, 23, 24, 58, 71, and 77.]

Park, J., Lee, J., Park, J., Ha, J.-W., & Nam, J. (2017). Representation learning using artist labels for audio classification tasks. [Cited on page 23.]

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in pytorch. [Cited on page 106.]

Pepino, L., Riera, P., & Ferrer, L. (2023). EnCodecMAE: Leveraging neural codecs for universal audio representation learning. *preprint arXiv:2309.07391*. [Cited on pages 22, 24, 98, and 102.]

Plachouras, C., Alonso-Jiménez, P., & Bogdanov, D. (2023). mir_ref: A representation evaluation framework for music information retrieval tasks. In *37th Conference on Neural Information Processing Systems (NeurIPS), Machine Learning for Audio Workshop*. [Not cited.]

Pons, J., Lidy, T., & Serra, X. (2016). Experimenting with musically motivated convolutional neural networks. In *International Workshop on Content-Based Multimedia Indexing (CBMI)*. [Cited on page 17.]

Pons, J., Nieto, O., Prockup, M., Schmidt, E., Ehmann, A., & Serra, X. (2017). End-to-end learning for music audio tagging at scale. *arXiv preprint arXiv:1711.02520*. [Cited on page 17.]

Pons, J., Serrà, J., & Serra, X. (2019). Training neural audio classifiers with few data. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 16–20. IEEE. [Cited on page 19.]

Pons, J. & Serra, X. (2019a). musicnn: Pre-trained convolutional neural networks for music audio tagging. *arXiv preprint arXiv:1909.06654*. [Cited on pages 34, 62, 65, 66, 88, 93, 94, 107, and 112.]

Pons, J. & Serra, X. (2019b). Randomly weighted cnns for (music) audio classification. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 336–340. [Cited on page 20.]

Porter, A., Bogdanov, D., Kaye, R., Tsukanov, R., & Serra, X. (2015). Acousticbrainz: a community platform for gathering music information obtained from audio. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 8, 33, and 35.]

Prinz, K., Flexer, A., & Widmer, G. (2021). On end-to-end white-box adversarial attacks in music information retrieval. *Transactions of the International Society for Music Information Retrieval*, *4*(1). [Cited on pages 97 and 104.]

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, *63*(10), 1872–1897. [Cited on page 4.]

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. [Cited on pages 4 and 121.]

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9. [Cited on page 4.]

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation. Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science. [Cited on page 15.]

Saeed, A., Grangier, D., & Zeghidour, N. (2021). Contrastive learning of general-purpose audio representations. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. [Cited on pages 21, 22, 58, and 82.]

Salimans, T. & Ho, J. (2021). Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*. [Cited on page 101.]

Scaringella, N. & Zoia, G. (2004). A real-time beat tracker for unrestricted audio signals. In *Journées d'informatique musicale*. [Cited on page 2.]

Schreiber, H. & Meinard, M. (2018). A single-step approach to musical tempo estimation using a convolutional neural network. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 2 and 25.]

Schreiber, H. & Müller, M. (2019). Musical tempo and key estimation using convolutional neural networks with directional filters. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 47–54. Málaga, Spain. [Cited on page 37.]

Simonyan, K. & Zisserman, A. (2014a). Very deep convolutional networks for large-scale image recognition. [Cited on page 34.]

Simonyan, K. & Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. [Cited on page 74.]

Smith, L. N. (2017). Cyclical learning rates for training neural networks. In *Winter Conference on Applications of Computer Vision (WACV)*. [Cited on page 76.]

Spijkervet, J. & Burgoyne, J. A. (2021). Contrastive learning of musical representations. In *International Society for Music Information Retrieval Conference (ISMIR'21)*. [Cited on pages 21, 22, and 66.]

Sturm, B. L. (2013). The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *arXiv:1306.1461v2 [cs.SD]*. [Cited on pages 38, 52, and 101.]

Sturm, B. L. (2017). The "horse" inside: Seeking causes behind the behaviors of music content analysis systems. *Comput. Entertain.*, *14*(2). [Cited on pages 97 and 104.]

Tan, M. & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR. [Cited on pages 40, 41, 62, and 93.]

Tan, M. & Le, Q. (2021). Efficientnetv2: Smaller models and faster training. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. [Cited on page 40.]

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., & Isola, P. (2020). What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems*. [Cited on page 22.]

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*. [Cited on pages 86, 87, and 91.]

Tovstogan, P., Bogdanov, D., & Porter, A. (2021). MediaEval 2021: Emotion and theme recognition in music using Jamendo. [Cited on page 65.]

Turian, J., Shier, J., Khan, H. R., Raj, B., Schuller, B. W., Steinmetz, C. J., Malloy, C., Tzanetakis, G., Velarde, G., McNally, K. et al. (2022). HEAR 2021: Holistic evaluation of audio representations. *arXiv:2203.03022v2 [cs.SD]*. [Cited on page 24.]

Van Den Oord, A., Dieleman, S., & Schrauwen, B. (2014). Transfer learning by supervised pre-training for audio-based music classification. In *Conference of the International Society for Music Information Retrieval (ISMIR 2014)*. [Cited on pages 6, 63, 76, and 89.]

van den Oord, A., Dieleman, S., & Schrauwen, B. (2014). Transfer learning by supervised pre-training for audio-based music classification. In *ISMIR*. [Cited on page 18.]

Van den Oord, A., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv e-prints*, pp. arXiv–1807. [Cited on page 21.]

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*. [Cited on page 85.]

Volk, A., Wiering, F., & Van Kranenburg, P. (2011). Unfolding the potential of computational musicology. In *Proceedings of the 13th International Conference on Informatics and Semiotics in Organisations*, pp. 137–144. [Cited on page 2.]

Wack, N., Guaus, E., Laurier, C., Marxer, R., Bogdanov, D., Serrà, J., & Herrera, P. (2009). Music Type Groupers (MTG): Generic Music Classification Algorithms. In *Music Information Retrieval Evaluation Exchange (MIREX'09)*. [Cited on page 8.]

Wack, N., Laurier, C., Meyers, O., Marxer, R., Bogdanov, D., Serra, J., Gomez, E., & Herrera, P. (2010). Music classification using high-level models. In *Music Information Retrieval Evaluation Exchange (MIREX)*. [Cited on page 8.]

Wang, L., Luc, P., Wu, Y., Recasens, A., Smaira, L., Brock, A., Jaegle, A., Alayrac, J.-B., Dieleman, S., Carreira, J. et al. (2022). Towards learning universal audio representations. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. [Cited on pages 4, 21, 22, 24, 74, 75, 83, and 121.]

Weck, B., Favory, X., Drossos, K., & Serra, X. (2021). Evaluating Off-the-Shelf Machine Listening and Natural Language Models for Automated Audio Captioning. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, pp. 60–64. [Cited on page 38.]

Won, M., Choi, K., & Serra, X. (2021). Semi-supervised music tagging transformer. In *International Society for Music Information Retrieval Conference (ISMIR)*. [Cited on pages 89, 92, 93, and 94.]

Won, M., Chun, S., Nieto, O., & Serrc, X. (2020a). Data-driven harmonic filters for audio representation learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 536–540. [Cited on page 18.]

Won, M., Chun, S., & Serra, X. (2019). Toward interpretable music tagging with self-attention. *arXiv preprint arXiv:1906.04972*. [Cited on pages 18 and 27.]

Won, M., Ferraro, A., Bogdanov, D., & Serra, X. (2020b). Evaluation of cnn-based automatic music tagging models. *arXiv preprint arXiv:2006.00751*. [Cited on pages 17, 41, 63, 65, 89, 93, and 94.]

Won, M., Hung, Y.-N., & Le, D. (2024). A foundation model for music informatics. In *ICASSP*. [Cited on page 122.]

Yao, D., Zhao, Z., Zhang, S., Zhu, J., Zhu, Y., Zhang, R., & He, X. (2022). Contrastive learning with positive-negative frame mask for music representation. In *Proceedings of the ACM Web Conference 2022*. [Cited on page 21.]

Yizhi, L., Yuan, R., Zhang, G., Ma, Y., Chen, X., Yin, H., Xiao, C., Lin, C., Ragni, A., Benetos, E. et al. (2023). MERT: Acoustic music understanding model with large-scale self-supervised training. In *The Twelfth International Conference on Learning Representations*. [Cited on pages 22, 27, and 122.]

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., & Tagliasacchi, M. (2021). SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *30*. [Cited on page 22.]

Zhang, H., Cissé, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*. [Cited on page 88.]

Zhao, H., Zhang, C., Zhu, B., Ma, Z., & Zhang, K. (2022). S3t: Self-supervised pre-training with swin transformer for music classification. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [Cited on pages 21, 27, and 66.]

Zhao, Y. & Guo, J. (2021). Musicoder: A universal music-acoustic encoder based on transformer. In *International Conference on Multimedia Modeling*, pp. 417–429. [Cited on page 22.]

Zinemanas, P., Rocamora, M., Miron, M., Font, F., & Serra, X. (2021). An interpretable deep learning model for automatic sound classification. *Electronics*, *10*(7). [Cited on pages 13, 27, 98, 100, and 120.]

# Appendix A: contributions by the author

## Peer-reviewed publications

- Alonso-Jiménez, P., Bogdanov, D., Pons, J., & Serra, X. (2020a). Tensorflow audio models in Essentia. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 266–270
- Correya, A., Alonso-Jiménez, P., Marcos-Fernández, J., Serra, X., & Bogdanov, D. (2021a). Essentia tensorflow models for audio and music processing on the web. In *Web Audio Conference (WAC)*
- Alonso-Jiménez, P., Serra, X., & Dmitry, B. (2022). Music representation learning based on editorial metadata from Discogs. In *International Society for Music Information Retrieval Conference (ISMIR)*
- Alonso-Jiménez, P., Favory, X., Foroughmand, H., Bourdalas, G., Serra, X., Lidy, T., & Bogdanov, D. (2023a). Pre-training strategies using contrastive learning and playlist information for music classification and similarity. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*
- Alonso-Jiménez, P., Serra, X., & Bogdanov, D. (2023b). Efficient supervised training of audio transformers for music representation learning. In *International Society for Music Information Retrieval Conference (ISMIR)*
- Alonso-Jiménez, P., Pepino, L., Batlle-Roca, R., Zinemanas, P., Bogdanov, D., Serra, X., & Rocamora, M. (2024). Leveraging pre-trained autoencoders for interpretable prototype learning of music audio. In *ICASSP Workshop on Explainable AI for Speech and Audio (XAI-SA)*

## Collaborations in peer-reviewed publications

- Correya, A. A., Marcos-Fernández, J., Joglar-Ongay, L., Alonso-Jiménez, P., Serra, X., & Bogdanov, D. (2021b). Audio and music analysis on the web using essentia.js. *Transactions of the International Society of Music Information Retrieval*, *4*, 167–181

- Buisson, M., Alonso-Jiménez, P., & Bogdanov, D. (2022). Ambiguity modelling with label distribution learning for music classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*

- Bogdanov, D., Lizarraga-Seijas, X., Alonso-Jiménez, P., & Serra, X. (2022). Musav: A dataset of relative arousal-valence annotations for validation of audio models. In *International Society for Music Information Retrieval Conference*

- Plachouras, C., Alonso-Jiménez, P., & Bogdanov, D. (2023). mir_ref: A representation evaluation framework for music information retrieval tasks. In *37th Conference on Neural Information Processing Systems (NeurIPS), Machine Learning for Audio Workshop*

## Other relevant publications

- Alonso-Jiménez, P., Bogdanov, D., & Serra, X. (2020b). Deep embeddings with Essentia models. *International Society for Music Information Retrieval Conference (ISMIR) Late-Braeking/Demo*

- Correya, A. A., Bogdanov, D., Alonso Jiménez, P., & Serra, X. (2022). Essentia API: a web API for music audio analysis. *International Society for Music Information Retrieval Conference (ISMIR) Late-Braeking/Demo*

## Ongoing research

- "ManyMusic dataset." A project with the goal of establishing a massive multi-modal open-access dataset for music-evoked emotions with paired audio embeddings and brain activity recordings.

- "Large-scale Self-supervised Audio Representation Models for Music Understanding." A project funded by a Barcelona Supercomputing Center grant of 500.00 hours of computing resources to continue the research direction of the thesis, July to October 2024.

## Industrial exploitation and outreach

- Patent (outcomes of research visit at Utopia Music): "Training methodology for contrastive learning based on associative metadata" (filed).

- Indústria del Coneixement 2023 Llavor program and María de Maeztu DTIC UPF program: funding for implementation of a web API service reutilizing the DNN models proposed on this thesis.
- Integration of the models into Essentia software library and online web demos. 1,200+ monthly downloads on PyPI, 200,000 runs of demos on Replicate, and 2 million runs of demos on Hugging Face.

## Recognition and awards

- Web Audio Conference 2021, "best paper award" (Correya et al., 2021a).
- Dolby Barcelona Scientific Paper Award 2023, "runner-up award" (Alonso-Jiménez et al., 2023b).