Increasing trust in complex machine learning systems

Studies in the music domain

Kim, Jaehun

DOI
10.4233/uuid:01ba927d-28e7-4abd-8193-e4ebef3b8218

Publication date
2021

Document Version
Final published version

Citation (APA)
Kim, J. (2021). *Increasing trust in complex machine learning systems: Studies in the music domain*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:01ba927d-28e7-4abd-8193-e4ebef3b8218

Increasing trust in complex machine learning systems

studies in the music domain

INCREASING TRUST IN COMPLEX MACHINE LEARNING SYSTEMS

STUDIES IN THE MUSIC DOMAIN

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates
to be defended publicly on
Wednesday 19 May 2021 at 15:00 o'clock

by

Jaehun KIM

Master of Science in Digital Contents and Information Studies,
born in Seoul, Republic of Korea.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. A. Hanjalic,	Delft University of Technology, promotor
Dr. ir. C.C.S. Liem,	Delft University of Technology, copromotor
Independent members:	
Prof. dr. ir. M.J.T. Reinders	Delft University of Technology
Prof. dr. A. van Deursen	Delft University of Technology
Prof. dr. M.A. Larson	Radboud University
Prof. dr. P. Tonella	Università della Svizzera italiana (USI), Switzerland
Dr. B.L.T. Sturm	KTH Royal Institute of Technology, Sweden

Keywords: Trustworthy Machine Learning, Music Information Retrieval, Transfer Learning, Recommender Systems

Printed by: Ridderprint

Front & Back: Beautiful cover art that captures the entire content of this thesis in a single illustration.

ISBN 978-94-6366-418-9

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

This thesis is dedicated to my family: my parents, grandmother, and JC.

Contents

Summary **xi**

Samenvatting **xiii**

1 Introduction **1**

1.1 The “Renaissance Era” of Machine Learning 2
1.2 Trustworthiness Concerns for Machine Learning 2
1.3 Principles of Trustworthy Machine Learning 3
1.4 Trustworthy Machine Learning for Music 4
1.5 Reliability and Explainability in Music IR 5
1.6 Thesis Contribution 5
1.7 Publication List 8
References 8

2 One deep music representation to rule them all? A comparative analysis of different representation learning strategies **13**

2.1 Introduction 14
2.2 Framework for Deep Representation Learning 17
2.2.1 Problem Definition 17
2.2.2 Learning Sources 19
2.2.3 Latent Factor Preprocessing 24
2.3 Representation Network Architectures 24
2.3.1 Base Architecture 25
2.3.2 Multi-Source Architectures with Various Degrees of Shared Information 26
2.3.3 MTL Training Procedure 28
2.3.4 Implementation Details 30
2.4 Evaluation 30
2.4.1 Target Datasets 31
2.4.2 Baselines 34
2.4.3 Experimental Design 35
2.4.4 Implementation Details 37
2.5 Results and Discussion 39
2.5.1 Single-Source and Multi-Source Representation 40
2.5.2 Effect of Number of Learning Sources and Fusion Strategy 40
2.5.3 Single-Source vs. Multi-Source 45
2.5.4 Compactness 47
2.5.5 Multiple Explanatory Factors 48
2.6 Conclusion 50
References 51

3 Transfer learning of artist group factors to musical genre classification 59
3.1 Introduction 59
3.2 Initial data exploration 61
3.3 Methodology 61
3.3.1 Artist Group Factors 61
3.3.2 Network Architectures 62
3.3.3 Pre-processing 65
3.3.4 Implementation Details 65
3.4 Experiments 65
3.5 Results 67
3.5.1 Multiple Learning Tasks in STN vs. MTN 67
3.5.2 Networks for Multiple Learning Tasks vs. Large Network on a Single Task 68
3.6 Discussion & Conclusion 68
References 69

4 Are Nearby Neighbors Relatives?: Testing Deep Music Embeddings 73
4.1 Introduction 73
4.2 Learning 75
4.2.1 Tasks 75
4.2.2 Network Architectures 77
4.2.3 Architecture and Learning Details 78
4.3 Measuring Distance Consistency 78
4.3.1 Distance Consistency 80
4.3.2 Distance Measures 81
4.3.3 Transformations 81
4.4 Experiment 83
4.4.1 Audio Pre-processing 83
4.4.2 Baseline 83
4.4.3 Dataset. 84
4.4.4 Performance Measures. 84
4.5 Results 85
4.5.1 Task-Specific Performance 85
4.5.2 Within-Space Consistency 86
4.5.3 Between-Space Consistency 88
4.5.4 Sensitivity to Imperceptible Transformations 91
4.6 Discussion and Conclusion 93
4.6.1 Effect of the Encoder 93
4.6.2 Effect of the Learning Task 93
4.6.3 Effect of the Transformation 94
4.6.4 Are Nearby Neighbors Relatives? 94
4.6.5 Towards Reliable Deep Music Embeddings 95
References 96

5 Evaluation Framework for Model-Agnostic Explainers: A Case Study on Recommender Systems 101
5.1 Introduction . . . 101
5.2 Framework . . . 103
5.3 Use case: Recommender Systems . . . 105
5.3.1 Recommenders . . . 105
5.3.2 Explainers . . . 108
5.3.3 Datasets . . . 110
5.4 Results . . . 112
5.4.1 Recommendation . . . 112
5.4.2 Explainability . . . 112
5.4.3 Discussion . . . 116
5.5 Conclusion & Future Work . . . 117
References . . . 118

6 Conclusion 123
6.1 On Reliable Use of Pre-trained Neural Networks . . . 124
6.2 On Making the ML Systems better explainable . . . 125
6.3 The way forward . . . 126
References . . . 128

Acknowledgements 129

Curriculum Vitæ 131

List of Publications 133

SUMMARY

Machine learning (ML) has become a core technology for many real-world applications. Modern ML models are applied to unprecedentedly complex and difficult challenges, including very large and subjective problems. For instance, applications towards multimedia understanding have been advanced substantially. Here, it is already prevalent that cultural/artistic objects such as music and videos are analyzed and served to users according to their preference, enabled through ML techniques.

One of the most recent breakthroughs in ML is Deep Learning (DL), which has been immensely adopted to tackle such complex problems. DL allows for higher learning capacity, making end-to-end learning possible, which reduces the need for substantial engineering effort, while achieving high effectiveness. At the same time, this also makes DL models more complex than conventional ML models. Reports in several domains indicate that such more complex ML models may have potentially critical hidden problems: various biases embedded in the training data can emerge in the prediction, extremely sensitive models can make unaccountable mistakes. Furthermore, the black-box nature of the DL models hinders the interpretation of the mechanisms behind them. Such unexpected drawbacks result in a significant impact on the trustworthiness of the systems in which the ML models are equipped as the core apparatus.

In this thesis, a series of studies investigates aspects of trustworthiness for complex ML applications, namely the reliability and explainability. Specifically, we focus on music as the primary domain of interest, considering its complexity and subjectivity. Due to this nature of music, ML models for music are necessarily complex for achieving meaningful effectiveness. As such, the reliability and explainability of music ML models are crucial in the field.

The first main chapter of the thesis investigates the transferability of the neural network in the Music Information Retrieval (MIR) context. Transfer learning, where the pre-trained ML models are used as off-the-shelf modules for the task at hand, has become one of the major ML practices. It is helpful since a substantial amount of the information is already encoded in the pre-trained models, which allows the model to achieve high effectiveness even when the amount of the dataset for the current task is scarce. However, this may not always be true if the “source” task which pre-trained the model shares little commonality with the “target” task at hand. An experiment including multiple “source” tasks and “target” tasks was conducted to examine the conditions which have a positive effect on the transferability. The result of the experiment suggests that the number of source tasks is a major factor of transferability. Simultaneously, it is less evident that there is a single source task that is universally effective on multiple target tasks. Overall, we conclude that considering multiple pre-trained models or pre-training a model employing heterogeneous source tasks can increase the chance for successful transfer learning.

The second major work investigates the robustness of the DL models in the transfer learning context. The hypothesis is that the DL models can be susceptible to imperceptible noise on the input. This may drastically shift the analysis of similarity among inputs, which is undesirable for tasks such as information retrieval. Several DL models pretrained in MIR tasks are examined for a set of plausible perturbations in a real-world setup. Based on a proposed sensitivity measure, the experimental results indicate that all the DL models were substantially vulnerable to perturbations, compared to a traditional feature encoder. They also suggest that the experimental framework can be used to test the pretrained DL models for measuring robustness.

In the final main chapter, the explainability of black-box ML models is discussed. In particular, the chapter focuses on the evaluation of the explanation derived from model-agnostic explanation methods. With black-box ML models having become common practice, model-agnostic explanation methods have been developed to explain a prediction. However, the evaluation of such explanations is still an open problem. The work introduces an evaluation framework that measures the quality of the explanations employing fidelity and complexity. Fidelity refers to the explained mechanism's coherence to the black-box model, while complexity is the length of the explanation.

Throughout the thesis, we gave special attention to the experimental design, such that robust conclusions can be reached. Furthermore, we focused on delivering machine learning framework and evaluation frameworks. This is crucial, as we intend that the experimental design and results will be reusable in general ML practice. As it implies, we also aim our findings to be applicable beyond the music applications such as computer vision or natural language processing.

Trustworthiness in ML is not a domain-specific problem. Thus, it is vital for both researchers and practitioners from diverse problem spaces to increase awareness of complex ML systems' trustworthiness. We believe the research reported in this thesis provides meaningful stepping stones towards the trustworthiness of ML.

SAMENVATTING

Machine learning (ML) is een kerntechnologie geworden voor veel toepassingen in het dagelijks leven. Hedendaagse ML-methoden worden toegepast op steeds complexere en moeilijkere uitdagingen, waaronder grootschalige, subjectieve problemen. Toepassingen om multimedia beter te begrijpen hebben bijvoorbeeld grote vooruitgangen geboekt. Het is tegenwoordig al gebruikelijk dat culturele en kunstzinnige objecten zoals muziek en videos geanalyseerd en aan gebruikers aangeboden worden op grond van hun voorkeur, dankzij ML-technieken.

Een van de meest recente doorbraken in ML is deep learning (DL), wat zeer uitgebreid is toegepast om zulke complexe problemen aan te pakken. DL heeft een grotere leercapaciteit. Hierdoor wordt end-to-end learning mogelijk, wat de noodzaak voor uitgebreide handmatige technische aanpassingen vermindert, terwijl grote effectiviteit bereikt kan worden. Tegelijkertijd zijn DL-modellen complexer dan traditionele ML-modellen. Rapportages uit verschillende vakgebieden geven aan dat zulke complexere ML-modellen mogelijk kritische verborgen problemen kunnen hebben: ongelijke verhoudingen en bias in trainingdata kunnen doorwerken in de voorspelling, en zeer gevoelige modellen kunnen onverwachte fouten maken, zonder duidelijke verantwoordelijkheid. De black-box-karakteristieken van DL-modellen maken interpretatie van onderliggende mechanismen ook moeilijker. Zulke onverwachte nadelen hebben aanzienlijke invloed op de betrouwbaarheid van systemen, waarin ML-modellen de kern vormen.

In deze dissertatie zullen verschillende aspecten van betrouwbaarheid in complexe ML-toepassingen worden bestudeerd, met name betrouwbaarheid en uitlegbaarheid. We richten ons specifiek op het muziekdomein, gezien de complexiteit en subjectiviteit van problemen in dit domein. Door deze karakteristieken van muziek, zijn complexe ML-modellen vaak nodig om betekenisvolle effectiviteit te krijgen. Dit betekent echter ook dat betrouwbaarheid en uitlegbaarheid van muziekgeoriënteerde ML-modellen cruciaal zijn.

Het eerste kernhoofdstuk van de dissertatie richt zich op overdraagbaarheid (transferability) van neurale netwerken in de Music Information Retrieval (MIR)-context. Transfer learning, waarin eerder getrainde ML-modellen worden ingezet als basismodules voor een gegeven taak, is een van de belangrijkste ML-praktijken. Deze praktijk is behulpzaam wanneer een substantiële hoeveelheid informatie in de eerder getrainde modellen geëncodeerd is. Hierdoor kan een model grote effectiviteit krijgen, zelfs als trainingdata voor de taak zelf schaars is. Dit geldt echter niet, als de “brontaak” van het eerder getrainde model weinig overeenkomsten heeft met de “doeltaak”. Een experiment met meerdere “brontaken” en “doeltaken” werd uitgevoerd, om de situaties te herkennen die positief effect op overdraagbaarheid hebben. De resultaten suggereren dat het aantal brontaken een belangrijke factor voor overdraagbaarheid is. Tegelijkertijd is het minder duidelijk of er een enkele brontaak bestaat, die universeel effectief is voor meerdere doeltaken. In het algemeen concluderen we dat het meenemen van meerdere eerder getrainde modellen,

of het eerder trainen van een model op basis van heterogene brontaken, de kans op succesvolle transfer learning doet toenemen.

Het tweede kernproject bestudeert de robuustheid van DL-modellen in de context van transfer learning. De hypothese is dat DL-modellen gevoelig kunnen zijn voor nietwaarneembare ruis in invoerdata. Dit kan vergelijkingen tussen inputs drastisch beïnvloeden, wat ongewenst is in taken als information retrieval. We bestuderen verschillende eerder getrainde DL-modellen voor MIR-taken, met plausibele perturbaties van data, die in het dagelijks leven kunnen voorkomen. Op basis van een voorgestelde maat voor gevoeligheid, tonen de experimentele resultaten aan dat DL-modellen gevoeliger zijn voor perturbaties, dan een traditionele kenmerkextractor. Ze suggereren ook dat het voorgestelde experimentele raamwerk gebruikt kan worden om eerder getrainde DL-modellen te testen op robuustheid.

In het laatste kernhoofdstuk wordt uitlegbaarheid van black-box ML models besproken. Het hoofdstuk focust in het bijzonder op de evaluatie van uitleg, die afgeleid is door model-agnostische uitlegbaarheidsmethoden. Nu black-box-modellen gebruikelijk zijn geworden, zijn zulke model-agnostische uitlegbaarheidsmethoden voorgesteld, om een voorspelling uit te kunnen leggen. Het is echter nog een open vraagstuk hoe deze uitleg geëvalueerd moet worden. Ons werk introduceert een evaluatieraamwerk, dat de kwaliteit van uitleg kwantificeert, op grond van getrouwheid (fidelity) en complexiteit. Getrouwheid wordt bepaald aan de hand van de coherentie tussen de uitgelegde mechanismen en het black-box model, waar complexiteit de lengte van de uitleg beschouwt.

In de hele dissertatie geven we speciale aandacht aan experimenteel ontwerp, opdat robuuste conclusies getrokken kunnen worden. Hiernaast richten we ons ook in het bijzonder op het afleveren van raamwerken voor machine learning en ML-evaluatie. Dit is cruciaal, aangezien we de bedoeling hebben dat het experimentele ontwerp en de resultaten herbruikbaar zullen zijn in algemene ML-praktijken. Onze intentie is dat onze uitkomsten ook toepasbaar zijn buiten het muziekdomein, bijvoorbeeld in computer vision en natural language processing.

Betrouwbaarheid van ML is geen domeinspecifiek probleem. Hierom is het van vitaal belang dat onderzoekers en beoefenaars uit verschillende probleemdomeinen het bewustzijn rond betrouwbaarheid van complexe ML-systemen vergroten. We menen dat het onderzoek in deze dissertatie op betekenisvolle wijze een springplank kan bieden, die aan deze discussies kan bijdragen.

1

INTRODUCTION

1.1. The "Renaissance Era" of Machine Learning

Statistical modeling forms one of the primary tools to predict and understand complex real-world phenomena. Nowadays, this technique is frequently applied in a computationally driven form, known as Machine Learning (ML). ML allows one to automate functionalities that are too complicated to be manually engineered. For instance, complex tasks, such as visual object recognition, can be automated due to highly capable underlying ML models.

The effectiveness of ML can be achieved when several conditions are met: 1) a sufficient amount of the data sampled from the complex phenomena to be modeled, 2) broad availability of ML models whose learning capacity and flexibility is sufficient to accommodate complex patterns at scale, and 3) significant computational capacity required to fit those models. For instance, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [1, 2] has been one of the most influential ML challenges. In particular, it confirmed the effectiveness of complex ML models such as Deep Convolutional Neural Network (DCNN) on the complex problem of visual object recognition [3–5]. For long, it had been infeasible to fit DCNNs on large-scale datasets like those offered by the ILSVRC, until the recent development of the General-Purpose Graphics Processing Unit (GPGPU), which allows for massive parallel execution of the mathematical operations that are crucial for fitting complex ML models.

High learning capacity of neural networks stems from their non-linearity and an immense number of parameters that can be trained, while the flexibility comes from a vast number of ways how to do this training. This makes neural network (NN) models capable of handling a wide range of data, ML tasks, and applications. Due to the high capacity and flexibility, it had been shown that even with relatively simple neural network architecture, any arbitrary function could be approximated [6]. This property allows much more efficient learning of highly complex problems when "deep" structure is introduced.

1.2. Trustworthiness Concerns for Machine Learning

Unlike other engineering artifacts that humans have invented, complex ML systems are artifacts for which the internal mechanism is increasingly unknown—even to their developers. For instance, NN models are highly complex, due to their non-linearity and many layers. Even while technically spoken, the complete mathematical description (i.e. coefficients, functions, inputs) of NN models can be considered transparent, exact decision-making processes within these models are extremely obscure to human observers. This problem will be magnified for systems encompassing off-the-shelf, pre-trained ML components, which already is common practice today.

Empirical reports have revealed that there are unexplained corners of currently deployed ML models, which may have negative impact on applying these models in practice. For instance, Sturm [7] showed that the NN models can seem to work correctly, while they actually capture hidden, irrelevant patterns (confounders), leading to wrong associations drawn from data. In this way, high accuracy that is often achieved by modern ML models can be misleading [8, 9]. For example, it can happen that the addition of imperceptible noise to input data deceives a NN model, resulting in a wrong decision. One of the well-studied special cases of this phenomenon considers noise structured "adversarially" to an expected model's decision [10, 11]. Wrong decisions also can happen because of hidden biases embedded in external factors surrounding the ML model training; for example, input data that got biased during its collection, or experimental setups not considering confounding factors [9, 12].

The role of ML and artificial intelligence (AI) in our socio-economic technical infrastructure is expected to grow in the future. If cases discussed above are encountered in practice, problems might arise regarding the broad adoption of ML-based solutions. It is therefore imperative to invest sufficient effort in making ML technology *trustworthy*.

1.3. PRINCIPLES OF TRUSTWORTHY MACHINE LEARNING

In recognition of the problems mentioned in the previous section, several initiatives have attempted to propose principles of trustworthy ML, which are meant to serve as guidelines for ML practice [13–17]. These studies focused on varying sets of values that an ML system must comply with, in order to prevent unexpected socio-technical or ethical failures. However, their terminologies and definitions are not aligned, which may potentially lead to confusion. In this thesis, we therefore only focus on two principles associated with trustworthy ML, which were implicitly pointed out by the examples given in the previous sections: *reliability* and *explainability*.

A *reliable* machine learning system minimizes the probability of a surprising failure [18]. Such failures can happen due to various reasons. For example, unidentified confounders can increase the chance that the resulting model shows unexplainable behavior. Furthermore, unexpected data distribution differences between the training and deployment phase may cause a system to be incapable of addressing changed contexts under which an ML system needs to operate (environment shift) [18]. For instance, a model trained on data collected from a specific time and sub-population might not be generalized to a different time and sub-population. Even when no such inherent distributional mismatches exist, the data acquisition process may not be representative, which can induce unexpected failures of resulting ML models. In this sense, reliability can be related to *robustness* or *consistency*, which in particular focus on the (in)consistent behavior of an ML system due to input noise [11, 19] or adversarial input [19, 20]. However, reliability can also be related to *transferability*: the extent to which a pre-trained model is useful for a newly applied "environment", such as a new task or ML system coupled with it.

Miller [21] defines *explainability*[1] as "the degree to which an observer can understand the cause behind a decision". Explainability becomes increasingly important with the growing perception of ML models as complex black boxes, in which it is less and less possible to link model elements to their influence on the model predictions. An explainable ML system brings many benefits: easier assessment of compliance to relevant legislation, more transparent entrances to verify and improve the system, and the possibility to enhance the trust between the user and the system [24]. What needs to be considered here, however, is a potential issue of conflicting principles, such as explainability and accuracy. Due to a larger learning capacity, a more complex, and thus less explainable model (e.g. a deep neural network) may achieve higher accuracy than an explainable, often less com-

[1] It often equates to the term *interpretability* in a number of works [21–23]. In this thesis, the two terms are used interchangeably as well.

plex (e.g. linear) model. Having said this, the increasing complexity of the ML models in order to secure the desired level of accuracy makes achieving a high level of explainability (and thus, trustworthiness) less and less trivial. More in-depth understanding of such systems is thus a crucial diagnostic step, such that unexpected violations of trustworthiness can be recognized and eventually handled. Focusing on this understanding also promotes the necessary critical discussions at different stages of devising trustworthy ML systems, making sure that the claim regarding trustworthiness is based on solid foundations. Our intention to contribute to these discussions served as the basis of the work presented in this thesis.

1.4. TRUSTWORTHY MACHINE LEARNING FOR MUSIC

The incorporation of principles of trustworthiness into ML practice cannot be done without a context. Different tasks, applications and data domains may emphasize some principles above others. The more complex data and use cases, the more emphasis should be given to explainability and reliability [7, 11]. Recently, complicated models are strongly prioritized to tackle such complex problems, which require large-scale datasets for effective training. As we will argue below, music, both as an application and data domain, provides several desirable properties on conducting trustworthy ML research under such settings.

Music, as multimedia category, manifests high complexity rooted in its multi-faceted nature [25]. Music can move one to appreciate and enjoy artistic creation and performance, but can also serve to "just" entertain us. Beyond active, immersive appreciation, there also is evidence that music can be used as support to other contextual activities [26]. Furthermore, as a physical phenomenon, music is a complex, typically multi-modal signal. It combines multiple sources of sound, which are organized in a particular way both in time and frequency. Most of Western popular music and a subset of classical music contain lyrics. Furthermore, music videos have become an important medium to express the intention of the artist and the music itself, enriching the music's impact on users. Music Information Retrieval (MIR) technology both focuses on describing music data and making it digitally accessible, largely through ML-based techniques [27].

MIR technology forms the backbone of today's music services, especially as music catalogues have grown extremely large. It therefore drives and steers our preferences in a domain with which we daily and heavily interact, and actively is studied by academic and industrial players alike. While humans have intuitive understanding of their preferences and perception of music, it is hard to pinpoint how human interpretation exactly relates to patterns observed in music data. Especially when high-capacity models are employed, the implicit complexity and subjectivity behind human music perception and preference can easily lead to many alternative, highly sensitive ML models. These may mimic human judgements, but upon closer looks, they may have picked up patterns that humans would certainly not have picked up. As such, principles of reliability and explainability are both challenging and natural for this field. Indeed, literature has both raised concerns on hidden reliability issues of ML-based music systems [7, 9, 28], and articulated the need for explainable MIR systems [29, 30]. Adversarial learning issues observed in the music domain [28] also were influential to the early development of the field in the broader ML domain [20, 31].

1.5. Reliability and Explainability in Music IR

In [9, 32], many of the reliability issues in MIR-related ML tasks were linked to the unavoidable subjective aspects in annotating and consuming music, which is difficult to capture by training data and incorporate in ML models. Similarly to computer vision [11], weak robustness to input noise can also be a grave issue in MIR, as audio can be "polluted" by environmental noise or various other degradations [33, 34]. Finally, due to the rapidly increasing deployment of transfer learning in MIR R&D practice, the transferability of pre-trained neural networks (e.g., for tasks such as music auto-tagging or recommendation) has become critical in the MIR domain [35–38].

Explainability has increasingly been addressed in music-related ML tasks as well. Choi et al. [39] adopted deconvolution [40] to identify sub-components of the input music signal, processed by each individual neuron within a convolutional neural network. The results suggest that one can identify a subset of units correlated with the specific functionality relevant to a given task. As an alternative to deconvolution, a saliency map [41], computed through the gradient of the neuron activation of interest, with respect to the input dimension, was also found effective for this purpose. Han et al. [42] deployed the technique to identify the region of interest of a subset of the neurons that are trained for the musical instrument recognition task, leading to a similar conclusion as in [39]. The concept of attention [43], where a gating mechanism conditioned on the input is learned through the training, is often considered as another way to improve the explainability of a deep neural network. Examples of deploying this mechanism in MIR are given in [44, 45]. Recent works of Slizovskaia et al. [46] and Chowdhury et al. [30] tried to interpret the latent activation of deep neural networks by investigating the correlation between this activation and transparent low-level or mid-level features. Finally, Mishra et al. [29] suggested a way to adopt local linear approximation methods [47] with a music audio signal by segmenting it along the frequency and/or the time axis, which is an analogy to the concept of a "super-pixel" in image processing, as suggested in [47].

1.6. Thesis Contribution

Despite the prior works presented above, trustworthy ML for music is still a newly emerging field, that has not necessarily been the center of attention. Instead, it would normally be a by-product of solutions to a specific ML task. In this thesis, we seek to put questions of reliability and explainability of music ML more front and center, and focus on related methodological practice that both allow more systematic study, while also being practically adoptable by R&D practitioners.

Aspects of reliability and explainability for ML in MIR are addressed throughout the technical chapters of the thesis, considering various music-related ML application scenarios. In particular, related to reliability, we consider *transferability* and *robustness*. These aspects are specifically investigated under the scenario of *transfer learning*, where off-the-shelf sub-networks are used as equivalents of a feature encoder, of which the reliability is not clearly known up front. Regarding explainability, we focus on model-agnostic explainers, the evaluation of which still is not clearly established in general. While studying this in the context of recommendation tasks, we hypothesize that our findings generalize beyond these. In the following, we elaborate in more detail about the contribution per chapter:

1. Chapter 2 focuses on the *transferability* of music ML. In modern ML practice, especially in computer vision and natural language processing, it is common that a neural-network-based learning model is trusted and reused beyond the original scope of the task it was trained for. In other words, such a pre-trained model is often “transferred" to a future unseen task. The most common transfer-learning practice is to pre-train a network on a single source task and deploy it for another single target task. The core assumption is that the pre-trained patterns, typically from a large-scale dataset, would be effective to the new task at hand. However, the assumption might not necessarily hold if the source task is not related, or in other words, if it shares little commonality with the target task. In that case, the effect of the pre-trained network on the overall learning performance can be sub-optimal, and even negatively influence this performance. It is, unfortunately, common that such potential incompatibility is hardly validated in practice. This may be due to the limited number of reasonably pre-trained networks being at the disposal of the R&D community, or due to the often limited resources to explore the alternatives (e.g., training new task-specific neural networks from scratch).

   In this chapter, we propose an approach to shed light onto the transferability of the networks pre-trained on a single task to a new target task. Furthermore, we investigate how to improve the overall reliability of transfer learning in the music domain. We build our approach on the intuitive hypothesis that increasing the number and diversity of source tasks on which pre-training is done is beneficial for improving the transfer-learning reliability. We investigate different ways of making use of these multiple source tasks, either by using them simultaneously to pre-train a single network, or by aggregating multiple networks each trained on a single, but different source task. We then deploy a range of target tasks to examine the conditions under which the transfer is successful. Our results indicate that the number and heterogeneity of tasks used in the pre-training process indeed has a positive effect on transferability. In Chapter 3, we verified the findings from Chapter 2 in practice by applying them to a real-world music classification problem.

2. In Chapter 4, we remain in the transfer learning scenario in the music domain, but now focusing on the *robustness* of the neural network. Transferability relates to the effectiveness of a pre-trained network. Robustness, on the other hand, relates to the extent to which a pre-trained network is capable to operate reliably in the presence of perturbation of the new input data. Robustness cannot be taken for granted due to highly complex non-linear transformations taking place in a neural network. It is well studied that such high complexity can lead to unexpected erroneous results if triggered by even the smallest perturbations at their input. Ideally, one could study the robustness of the pre-trained network by testing its performance on the task it was initially trained for various input perturbations. In practice, however, this is difficult to do, as the original dataset may not be accessible. Also, testing the robustness on the source task might not accurately reflect the actual robustness on the target task.

   In this chapter, we report the results of our search for an effective, practical solution to assess robustness of a pre-trained network upfront, even before testing the network on a target task. The suggested assessment, thus, does not require the access to the datasets corresponding either to the source or target task. We achieve it by a testing framework that focuses on the analysis of the internal representation of the given pre-trained network. The hypothesis underlying our proposed testing framework is that small, barely perceptible perturbations should not drastically shift the corresponding latent representation of the data points from their original position. We deployed our framework on a range of pre-trained neural networks, which revealed that all the tested networks are vulnerable to input data perturbations. Deploying our framework on a network at hand can provide quick insight into the robustness of the network for transfer learning.

3. In Chapter 5, we propose an evaluation framework for assessing the quality of the explanation given by post-hoc *model explainers* on "black-box systems". We refer to a black-box system as a model which is so complex, that human interpretation of its internal decision-making mechanisms is virtually impossible. Several attempts have been reported in literature to achieve post-hoc explanation of such models, by fitting an interpretable model to mimic the complex behavior of a target black-box model. However, the systematic evaluation of those explanation methods, in terms of the reliability of the generated explanations, has received insufficient attention.

   In this chapter, we propose a method to assess post-hoc model explainers. We do so by looking at the correlation between the input-output behavior of the given black-box system and the behavior of the explainer at hand, and at the complexity of the generated explanation. The underlying intuition is that the best explainer to be selected is the one with maximum correlation (fidelity) and minimum complexity. This can be seen as a special case of the formalism of the explainability from [47]. We demonstrate the effectiveness of our evaluation method by conducting an experiment where ML-based recommender systems are applied to music recommendation and book recommendation.

In the studies reported throughout the thesis, we gave special attention to the experimental design, ensuring sufficient variability within the ML scenario each study considered. In particular, the studies deliberately chose a wide range of datasets and ML models, such that the conclusions do not depend on a specific experimental choice.

Although the problems we address in the thesis are inspired by ML applications in the music domain, it was also our intention that the experimental designs, results, and findings are reusable in general ML practice. In this respect, one of the main aims of this thesis is to provide ML practitioners with feasible frameworks encompassing trustworthy ML. This includes the formulation of the measurements that closely relate to the principles of trustworthy ML, and research frameworks that utilize those measurements to pursue trust in ML. Using these measurements and frameworks, the same questions we posed in the MIR context may be answered in other fields, such as the computer vision or the natural language processing, with minimal adaptation.

1.7. PUBLICATION LIST

9. Ahn, H., Kim, J., Kim, K., & Oh, S. (2020). Generative Autoregressive Networks for 3D Dancing Move Synthesis from Music. IEEE Robotics and Automation Letters.

8. Kim, J., Urbano, J., Liem, C. C. S. & Hanjalic, A. (2020). One deep music representation to rule them all? A comparative analysis of different representation learning strategies. Neural Comput & Applic 32, 1067–1093.

7. Kim, J., Demetriou, A. M., Manolios, S., & Liem, C. C. S. (2019). Beyond Explicit Reports: Comparing Data-Driven Approaches to Studying Underlying Dimensions of Music Preference. In Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization (pp. 285-293).

6. Kim, J., Urbano, J., Liem, C. C. S., & Hanjalic, A. (2019). Are Nearby Neighbors Relatives?: Are Nearby Neighbors Relatives?: Testing Deep Music Embeddings. Frontiers in Applied Mathematics and Statistics, 5, 53.

5. Kim, J., Picek, S., Heuser, A., Bhasin, S., & Hanjalic, A. (2019). Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems, 148-179.

4. Picek, S., Samiotis, I. P., Kim, J., Heuser, A., Bhasin, S., & Legay, A. (2018). On the performance of convolutional neural networks for side-channel analysis. In International Conference on Security, Privacy, and Applied Cryptography Engineering (pp. 157-176). Springer, Cham.

3. Kim, J., Won, M., Liem, C. C. S., & Hanjalic, A. (2018). Towards seed-free music playlist generation: Enhancing collaborative filtering with playlist title information. In Proceedings of the ACM Recommender Systems Challenge 2018 (pp. 1-6).

2. Kim, J., Won, M., Serra, X., & Liem, C. C. S. (2018). Transfer Learning of Artist Group Factors to Musical Genre Classification. In Companion Proceedings of the The Web Conference 2018 (WWW '18). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1929–1934.

1. Kim, C. W., Kim, J., Kim, K., & Won, M. (2017). Single and Multi-Column Neural Networks for Content-based Music Genre Recognition. In MediaEval.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, *ImageNet: A large-scale hierarchical image database,* in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2009, 20-25 June 2009, Miami, Florida, USA* (IEEE Computer Society, 2009) pp. 248–255.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, *ImageNet large scale visual recognition challenge,* Int. J. Comput. Vis. **115**, 211 (2015).

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks,* Commun. ACM **60**, 84 (2017).

[4] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition,* in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, June 27-30, 2016, Las Vegas, NV, USA* (IEEE Computer Society, 2016) pp. 770–778.

[5] M. Tan and Q. V. Le, *EfficientNet: Rethinking model scaling for convolutional neural networks,* in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Proceedings of Machine Learning Research, Vol. 97, edited by K. Chaudhuri and R. Salakhutdinov (PMLR, 2019) pp. 6105–6114.

[6] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).

[7] B. L. Sturm, *The "horse" inside: Seeking causes behind the behaviors of music content analysis systems,* Comput. Entertain. **14**, 3:1 (2016).

[8] M. Lagrange and M. Rossignol, *Computational experiments in Science: Horse wrangling in the digital age,* in *Research workshop on "Horses" in Applied Machine Learning* (London, United Kingdom, 2016).

[9] B. L. Sturm, *Classification accuracy is not enough - on the evaluation of music genre recognition systems,* J. Intell. Inf. Syst. **41**, 371 (2013).

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2015).

[11] A. Kurakin, I. J. Goodfellow, and S. Bengio, *Adversarial examples in the physical world,* in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings* (OpenReview.net, 2017).

[12] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J. Crespo, and D. Dennison, *Hidden technical debt in machine learning systems,* in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, edited by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (2015) pp. 2503–2511.

[13] T. Hagendorff, *The ethics of AI ethics: An evaluation of guidelines,* Minds Mach. **30**, 99 (2020).

[14] B. Mittelstadt, *Principles alone cannot guarantee ethical AI,* Nat Mach Intel **1**, 501 (2019).

[15] High-Level Expert Group on AI, *Ethics guidelines for trustworthy AI,* Report (European Commission, Brussels, 2019).

[16] E. Toreini, M. Aitken, K. P. L. Coopamootoo, K. Elliott, V. G. Zelaya, P. Missier, M. Ng, and A. van Moorsel, *Technologies for trustworthy machine learning: A survey in a socio-technical context,* CoRR **abs/2007.08911** (2020), arXiv:2007.08911 .

[17] L. Floridi and J. Cowls, *A unified framework of five principles for AI in society,* Harvard Data Science Review **1** (2019), 10.1162/99608f92.8cd550d1.

[18] S. Saria and A. Subbaswamy, *Tutorial: Safe and reliable machine learning,* CoRR **abs/1904.07204** (2019), arXiv:1904.07204 .

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, *Intriguing properties of neural networks,* in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2014).

[20] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, *Technical report on the CleverHans v2.1.0 adversarial examples library,* arXiv preprint arXiv:1610.00768 (2018).

[21] T. Miller, *Explanation in artificial intelligence: Insights from the social sciences,* Artif. Intell. **267**, 1 (2019).

[22] C. Molnar, *Interpretable Machine Learning* (2019).

[23] B. D. Mittelstadt, C. Russell, and S. Wachter, *Explaining explanations in AI,* in *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, Atlanta, GA, USA, January 29-31, 2019* (ACM, 2019) pp. 279–288.

[24] F. Doshi-Velez, M. Kortz, R. Budish, C. Bavitz, S. Gershman, D. O'Brien, S. Schieber, J. Waldo, D. Weinberger, and A. Wood, *Accountability of AI under the law: The role of explanation,* CoRR **abs/1711.01134** (2017), arXiv:1711.01134 .

[25] C. C. S. Liem, *Multifaceted approaches to music information retrieval*, Ph.D. thesis, Delft University of Technology (2015).

[26] K. Yadati, *Music in Use: Novel perspectives on content-based music Retrieval*, Ph.D. thesis, Delft University of Technology (2019).

[27] M. Schedl, E. Gómez, and J. Urbano, *Music information retrieval: Recent developments and applications,* Found. Trends Inf. Retr. **8**, 127 (2014).

[28] B. L. Sturm, *A simple method to determine if a music information retrieval system is a "horse",* IEEE Trans. Multimedia **16**, 1636 (2014).

[29] S. Mishra, B. L. Sturm, and S. Dixon, *Local interpretable model-agnostic explanations for music content analysis,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, edited by S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull (2017) pp. 537–543.

[30] S. Chowdhury, A. Vall, V. Haunschmid, and G. Widmer, *Towards explainable music emotion recognition: The route via mid-level features,* in *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, edited by A. Flexer, G. Peeters, J. Urbano, and A. Volk (2019) pp. 237–243.

[31] I. J. Goodfellow, *Defense against the dark arts: An overview of adversarial example security research and future research directions,* CoRR **abs/1806.04169** (2018), arXiv:1806.04169 .

[32] K. Choi, G. Fazekas, K. Cho, and M. B. Sandler, *The effects of noisy labels on deep convolutional neural networks for music tagging,* IEEE Trans. Emerg. Top. Comput. Intell. **2**, 139 (2018).

[33] M. Mauch and S. Ewert, *The audio degradation toolbox and its application to robustness evaluation,* in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, edited by A. de Souza Britto Jr., F. Gouyon, and S. Dixon (2013) pp. 83–88.

[34] J. Urbano, D. Bogdanov, P. Herrera, E. Gómez, and X. Serra, *What is the effect of audio quality on the robustness of MFCCs and chroma features?* in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, edited by H. Wang, Y. Yang, and J. H. Lee (2014) pp. 573–578.

[35] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, *Transfer learning for music classification and regression tasks,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, edited by S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull (2017) pp. 141–149.

[36] S. Dieleman, P. Brakel, and B. Schrauwen, *Audio-based music classification with a pretrained convolutional network,* in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, edited by A. Klapuri and C. Leider (University of Miami, 2011) pp. 669–674.

[37] A. van den Oord, S. Dieleman, and B. Schrauwen, *Deep content-based music recommendation,* in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2015. December 5-8, 2013, Lake Tahoe, Nevada, United States*, edited by C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger (2013) pp. 2643–2651.

[38] D. Liang, M. Zhan, and D. P. W. Ellis, *Content-aware collaborative music recommendation using pre-trained neural networks,* in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, edited by M. Müller and F. Wiering (2015) pp. 295–301.

[39] K. Choi, G. Fazekas, and M. B. Sandler, *Explaining deep convolutional neural networks on music classification,* CoRR **abs/1607.02444** (2016), arXiv:1607.02444 .

[40] M. D. Zeiler and R. Fergus, *Visualizing and understanding convolutional networks,* in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, Lecture Notes in Computer Science, Vol. 8689, edited by D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Springer, 2014) pp. 818–833.

[41] K. Simonyan, A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps,* in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, edited by Y. Bengio and Y. LeCun (2014).

[42] Y. Han, J. Kim, and K. Lee, *Deep convolutional neural networks for predominant instrument recognition in polyphonic music,* IEEE ACM Trans. Audio Speech Lang. Process. **25**, 208 (2017).

[43] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2015).

[44] S. Lee, J. Lee, and K. Lee, *Content-based feature exploration for transparent music recommendation using self-attentive genre classification,* CoRR **abs/1808.10600** (2018), arXiv:1808.10600 .

[45] M. Won, S. Chun, and X. Serra, *Toward interpretable music tagging with self-attention,* CoRR **abs/1906.04972** (2019), arXiv:1906.04972 .

[46] O. Slizovskaia, E. Gómez, and G. Haro, *A case study of deep-learned activations via hand-crafted audio features,* CoRR **abs/1907.01813** (2019), arXiv:1907.01813 .

[47] M. T. Ribeiro, S. Singh, and C. Guestrin, *"why should I trust you?": Explaining the predictions of any classifier,* in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, edited by B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi (ACM, 2016) pp. 1135–1144.

2

One deep music representation to rule them all? A comparative analysis of different representation learning strategies

Inspired by the success of deploying deep learning in the fields of Computer Vision and Natural Language Processing, this learning paradigm has also found its way into the field of Music Information Retrieval. In order to benefit from deep learning in an effective, but also efficient manner, deep transfer learning has become a common approach. In this approach, it is possible to reuse the output of a pre-trained neural network as the basis for a new learning task. The underlying hypothesis is that if the initial and new learning tasks show commonalities and are applied to the same type of input data (e.g. music audio), the generated deep representation of the data is also informative for the new task. Since, however, most of the networks used to generate deep representations are trained using a single initial learning source, their representation is unlikely to be informative for all possible future tasks. In this paper, we present the results of our investigation of what are the most important factors to generate deep representations for the data and learning tasks in the music domain. We conducted this investigation via an extensive empirical study that involves multiple learning sources, as well as multiple deep learning architectures with varying levels of information sharing between sources, in order to learn music representations. We then validate these representations considering multiple target datasets for evaluation. The results of our experiments yield several insights on how to approach the design of methods for learning widely deployable deep data representations in the music domain.

This chapter was published in Neural Computing and Applications **32** (4), (2020) [1]

2.1. Introduction

In the Music Information Retrieval (MIR) field, many research problems of interest involve the automatic description of properties of musical signals, employing concepts that are understood by humans. For this, tasks are derived that can be solved by automated systems. In such cases, algorithmic processes are employed to map raw music audio information to humanly understood descriptors (e.g. genre labels or descriptive tags). To achieve this, historically, the raw audio would first be transformed into a *representation* based on *hand-crafted features*, which are engineered by humans to reflect dedicated semantic signal properties. The feature representation would then serve as input to various statistical or Machine Learning (ML) approaches [2].

The framing as described above can generally be applied to many applied ML problems: complex real-world problems are abstracted into a relatively simpler form, by establishing tasks that can be computationally addressed by automatic systems. In many cases, the task involves making a prediction based on a certain observation. For this, modern ML methodologies can be employed, that automatically can infer the logic for the prediction directly from (a numeric representation of) the given data, by optimizing an objective function defined for the given task.

However, music is a multimodal phenomenon, that can be described in many parallel ways, ranging from objective descriptors to subjective preference. As a consequence, in many cases, while music-related tasks are well understood by humans, it often is hard to pinpoint and describe where the truly 'relevant' information is in the music data used for the tasks, and how this properly can be translated into numeric representations that should be used for prediction. While research into such proper translations can be conducted per individual task, it is likely that informative factors in music data will be shared across tasks. As a consequence, when seeking to identify informative factors that are not explicitly restricted to a single task, Multi-Task Learning (MTL) is a promising strategy. In MTL, a single learning framework hosts multiple tasks at once, allowing for models to perform better by sharing commonalities between involved tasks [3]. MTL has been successfully used in a range of applied ML works [4–11], also including the music domain [12, 13].

Following successes in the fields of Computer Vision (CV) and Natural Language Processing (NLP), deep learning approaches have recently also gained increasing interest in the MIR field, in which case *deep representations* of music audio data are directly learned from the data, rather than being hand-crafted. Many works employing such approaches reported considerable performance improvements in various music analysis, indexing and classification tasks [14–21].

In many deep learning applications, rather than training a complete network from scratch, pre-trained networks are commonly used to generate deep representations, which can be either directly adopted or further adapted for the current task at hand. In CV and NLP, (parts of) certain pre-trained networks [22–25] have now been adopted and adapted in a very large number of works. These 'standard' deep representations have typically been obtained by training a network for a single learning task, such as visual object recognition, employing large amounts of training data. The hypothesis on why these representations are effective in a broader of spectrum of tasks than they originally were trained for, is that *deep transfer learning (DTL)* is happening: information initially picked up by the network is beneficial also for new learning tasks performed on the same type of raw input data.

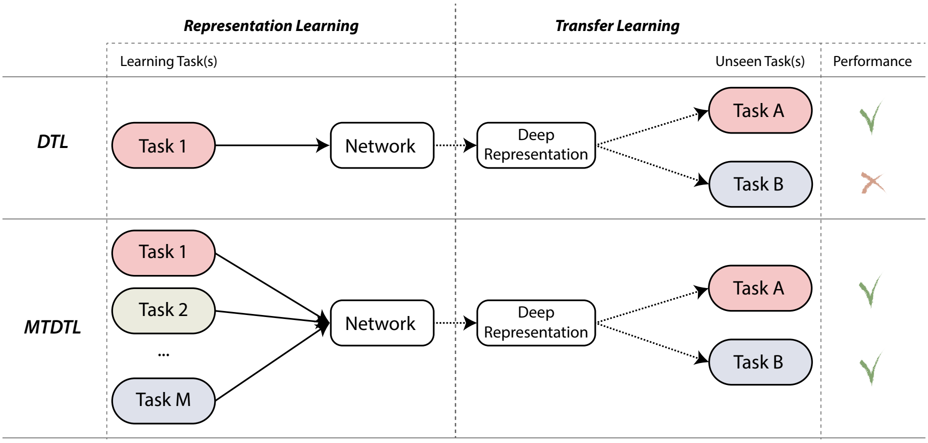


Figure 2.1: Simplified illustration of the conceptual difference between traditional deep transfer learning (DTL) based on a single learning task (above) and multi-task based deep transfer learning (MTDTL) (below). The same color used for a learning and an target task indicates that the tasks have commonalities, which implies that the learned representation is likely to be informative for the target task. At the same time, this representation may not be that informative to another future task, leading to a low transfer learning performance. The hypothesis behind MTDTL is that relying on more learning tasks increases robustness of the learned representation and its usability for a broader set of target tasks.

Clearly, the validity of this hypothesis is linked to the extent to which the new task can rely on similar data characteristics as the task on which the pre-trained network was originally trained.

Although a number of works deployed DTL for various learning tasks in the music domain[26–29], to our knowledge, however, transfer learning and the employment of pre-trained networks are not as standard in the MIR domain as in the CV domain. Again, this may be due to the broad and partially subjective range and nature of possible music descriptions. Following the considerations above, it may then be useful to combine deep transfer learning with multi-task learning.

Indeed, in order to increase robustness to a larger scope of new learning tasks and datasets, the concept of MTL also has been applied in training deep networks for representation learning, both in the music domain [12, 13] and in general [4, p. 2]. As the model learns several tasks and datasets in parallel, it may pick up commonalities among them. As a consequence, the expectation is that a network learned with MTL will yield robust performance across different tasks, by transferring shared knowledge [3, 4]. A simple illustration of the conceptual difference between traditional DTL and deep transfer learning based on MTL (further referred to as *multi-task based deep transfer learning (MTDTL))* is shown in Fig. 2.1.

The mission of this paper is to investigate the effect of conditions around the setup of MTDTL, which are important to yield effective deep music representations. Here, we understand an 'effective' representation to be a representation that is suitable for a wide range of new tasks and datasets. Ultimately, we aim for providing a methodological framework to systematically obtain and evaluate such transferable representations. We pursue this mission by exploring the effectiveness of MTDTL and traditional DTL, as well as concatenations of multiple deep representations, obtained by networks that were independently trained on separate single learning tasks. We consider these representations for multiple choices of learning tasks and considering multiple target datasets.

Our work will address the following research questions:

- **RQ1:** Given a set of learning sources that can be used to train a network, what is the influence of the number and type of the sources on the effectiveness of the learned deep representation?
- **RQ2:** How do various degrees of information sharing in the deep architecture affect the effectiveness of a learned deep representation?

By answering the **RQ1** we arrive at an understanding of important factors regarding the composition of a set of learning tasks and datasets (which in the remainder of this work will be denoted as *learning sources*) to achieve an effective deep music representation, specifically on the number and nature of learning sources. The answer to **RQ2** provides insight in *how to choose the optimal multi-task network architecture* under a MTDTL context. For example, in MTL, multiple sources are considered under a joint learning scheme, that partially shares inferences obtained from different learning sources in the learning pipeline. In MTL applications using deep neural networks, this means that certain layers will be shared between all sources, while at other stages, the architecture will 'branch' out into source-specific layers [3, 6–9, 13, 30]. However, investigation is still needed on where in the layered architecture branching should ideally happen—if a branching strategy would turn out beneficial in the first place.

To reach the aforementioned answers, it is necessary to conduct a systematic assessment to examine relevant factors. For **RQ1**, we investigate different numbers and combinations of learning sources. For **RQ2**, we study different architectural strategies. However, we wish to ultimately investigate effectiveness of the representation with respect to new, target learning tasks and datasets (which in the remainder of this paper will be denoted by *target datasets*). While this may cause combinatorial explosion with respect to possible experimental configurations, we will make strategic choices in the design and evaluation procedure of the various representation learning strategies.

The scientific contribution of this work can be summarized as follows:

- We provide insight into the effectiveness of various deep representation learning strategies under the multi-task learning context.
- We offer in-depth insight into ways to evaluate desired properties of a deep representation learning procedure.
- We propose and release several pre-trained music representation networks, based on different learning strategies for multiple semantic learning sources.

The rest of this work is presented as following: a formalization of this problem, as well as the global outline of how learning will be performed based on different learning tasks from different sources, will be presented in Section 2.2. Detailed specifications of the deep architectures we considered for the learning procedure will be discussed in Section 2.3. Our strategy to *evaluate* the effectiveness of different representation network variants by employing various *target datasets* will be the focus of Section 2.4. Experimental results will be discussed in Section 2.5, after which general conclusions will be presented in Section 2.6.

2.2. Framework for Deep Representation Learning

In this section, we formally define the deep representation learning problem. As Fig. 2.2 illustrates, any domain-specific MTDTL problem can be abstracted into a formal task, which is instantiated by a specific dataset with specific observations and labels. Multiple tasks and datasets are involved to emphasize different aspects of the input data, such that the learned representation is more adaptable to different future tasks. The learning part of this scheme can be understood as the MTL phase, which is introduced in Section 2.2.1. Subsequently in Section 2.2.2, we discuss learning sources involved in this work, which consist of various tasks and datasets to allow investigating their effects on the transfer learning. Further, we introduce the label preprocessing procedure that is applied in this work in Section 2.2.3, ensuring that the learning sources are more regularized, such that their comparative analysis is clearer.

2.2.1. Problem Definition

A machine learning problem, focused on solving a specific task t, can be formulated as a minimization problem, in which a model function f_t must be learned that minimizes a loss function $\mathscr{L}$ for given dataset $\mathscr{D}_t = \{(x_t^{(i)}, y_t^{(i)}) \mid i \in \{1, \cdots, I\}\}$, comparing the model's predictions given by the input x_t and actual task-specific learning labels y_t. This can be formulated using the following expression:

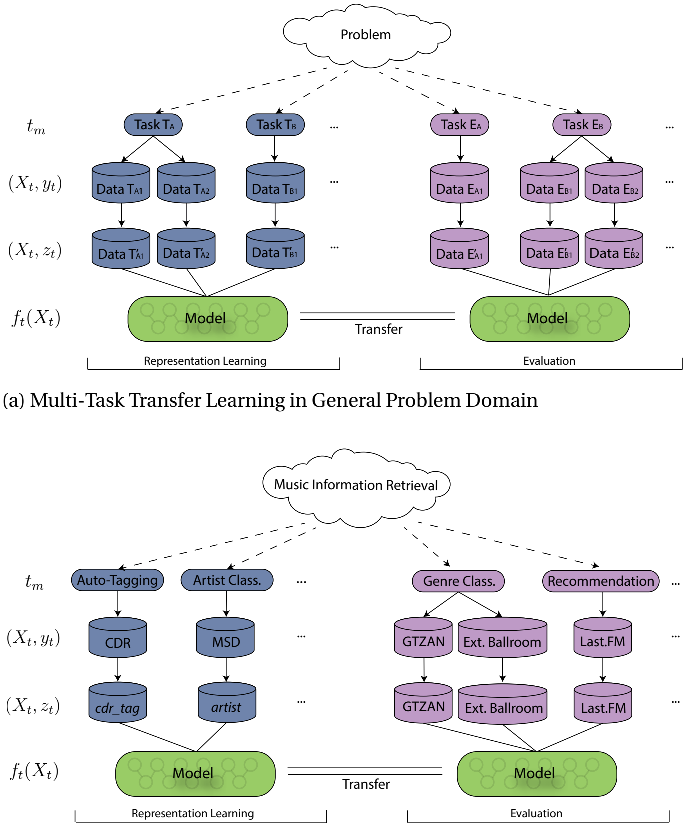
$$\hat{\theta} = \arg\min \mathbb{E}_{\mathscr{D}_t} \mathscr{L}(y_t, f_t(x_t; \theta)) \tag{2.1}$$

where $x_t \in \mathbb{R}^d$ is, traditionally, a hand-crafted d-dimensional feature vector and θ is a set of model parameters of f.

When deep learning is employed, the model function f denotes a learnable network. Typically, the network model f is learned in an end-to-end fashion, from raw data at the input to the learning label. In the speech and music field, however, using true end-to-end learning is still not a common practice. Instead, raw data is typically transformed first, before serving as network input. More specifically, in the music domain, common input to function f would be $X \in \mathbb{R}^{c \times n \times b}$, replacing the originally hand-crafted feature vector $x \in \mathbb{R}^d$ from (2.1) by a time-frequency representation of the observed music data, usually obtained through the Short-Time Fourier Transform (STFT), with potential additional filter bank applications (e.g. mel-filter bank). The dimensions c, n, b indicate channels of the audio signal, time steps, and frequency bins respectively.

If such a network still is trained for a specific single machine learning task t, we can now reformulate (2.1) as follows:

$$\hat{\theta} = \arg\min \mathbb{E}_{\mathscr{D}_t} \mathscr{L}(y_t, f_t(X_t; \theta)). \tag{2.2}$$



(a) Multi-Task Transfer Learning in General Problem Domain

(b) Multi-Task Transfer Learning in Music Information Retrieval Domain

Figure 2.2: Schematic overview of what this work investigates. The upper scheme illustrates a general problem solving framework in which multi-task transfer learning is employed. The tasks $t \in \{t_0, t_1, \cdots, t_M\}$ are derived from a certain problem domain, which are instantiated by datasets, that often are represented as sample pairs of observations and corresponding labels (X_t, y_t). Sometimes, the original dataset is processed further into simpler representation forms (X_t, z_t), to filter out undesirable information and noise. Once a model or system $f_t(X_t)$ has learned the necessary mappings within the learning sources, this knowledge can be transferred to another set of target datasets, leveraging commonalities already obtained by the pre-training. Below the general framework, we show a concrete example, in which the broad MIR problem domain is abstracted into various sub-problems with corresponding tasks and datasets.

In MTL, in the process of learning the network model f, different tasks will need to be solved in parallel. In case of deep neural networks, this is usually realized by having a network in which lower layers are shared for all tasks, but upper layers are task-specific. Given m different tasks t, each having the learning label y_t, we can formulate the learning objective of the neural network in a MTL scenario as follows:

$$\hat{\theta}^s, \hat{\theta}^* = \arg\min \mathbb{E}_{t \in \mathscr{T}} \mathbb{E}_{\mathscr{D}_t} \mathscr{L}(y_t, f_t(X_t; \theta^s, \theta^t)) \tag{2.3}$$

Here, $\mathscr{T} = \{t_1, t_2, ..., t_m\}$ is a given set of tasks to be learned and $\theta^* = \{\theta^1, \theta^2, ..., \theta^m\}$ indicates a set of model parameters θ^t with respect to each task. Since the deep architecture initially shares lower layers and branches out to task-specific upper layers, the parameters of shared layers and task-specific layers are referred to separately as θ^s and θ^t, respectively. Updates for all parameters can be achieved through standard back-propagation. Further specifics on network architectures and training configurations will be given in Section 2.3.

Given the formalizations above, the first step in our framework is to select a suitable set $\mathscr{T}$ of learning tasks. These tasks can be seen as multiple concurrent descriptions or transformations of the same input fragment of musical audio: each will reflect certain semantic aspects of the music. However, unlike the approach in a typical MTL scheme, solving multiple specific learning tasks is actually not our main goal; instead, we wish to learn an effective *representation* that captures as many semantically important factors in the low-level music representation as possible. Thus, rather than using learning labels y_t, our representation learning process will employ reduced learning labels z_t, which capture a reduced set of semantic factors from y_t. We then can reformulate (2.3) as follows:

$$\hat{\theta}^s, \hat{\theta}^* = \arg\min \mathbb{E}_{t \in \mathscr{T}} \mathbb{E}_{\mathscr{D}_t} \mathscr{L}(z_t, f_t(X_t; \theta^s, \theta^t)) \tag{2.4}$$

where $z_t \in \mathbb{R}^k$ is a k-dimensional vector that represents reduced learning label for a specific task t. Each z_t will be obtained through task-specific factor extraction methods, as described in Section 2.2.3.

2.2.2. LEARNING SOURCES

In the MTDTL context, a training dataset can be seen as the 'source' to learn the representation, which will be further transferred to the future 'target' dataset. Different learning sources of different nature can be imagined, that can be globally categorized as *Algorithm* or *Annotation*. As for the *Algorithm* category, by employing traditional feature extraction or representation transformation algorithms, we will be able to automatically extract semantically interesting aspects from input data. As for the *Annotation* category, these include different types of label annotations of the input data by humans.

The dataset used as resource for our learning experiments is the Million Song Dataset (MSD) [31]. In its original form, it contains metadata and precomputed features for a million songs, with several associated data resources, e.g. considering `Last.fm` social tags and listening profiles from `the Echo Nest`. While the MSD does not distribute audio due to copyright reasons, through the API of the `7digital` service, 30-second audio previews can be obtained for the songs in the dataset. These 30-second previews will form the source for our raw audio input.

Using the MSD data, we consider several subcategories of learning sources within the *Algorithm* and *Annotation* categories; below, we give an overview of these, and specify what information we considered exactly for the learning labels in our work.

Algorithm

- ***Self.*** The music track is the learning source itself; in other words, intrinsic information in the input music track should be captured through a learning procedure, without employing further data. Various unsupervised or auto-regressive learning strategies can be employed under this category, with variants of Autoencoders, including the Stacked Autoencoder [32, 33], Restricted Boltzmann Machines (RBM) [34], Deep Belief Networks (DBN) [35] and Generative Adversarial Networks (GAN) [36]. As another example within this category, variants of the Siamese networks for similarity learning can be considered [37–39].

  In our case, we will employ the Siamese architecture to learn a metric that measures whether two input music clips belong to the same track, or two different tracks. This can be formulated as follows:

$$\hat{\theta}^{self}, \hat{\theta}^{s} = \arg\min \mathbb{E}_{X_l, X_r \sim \mathcal{D}_{self}} \mathcal{L}(y_{self}, f_{self}(X_l, X_r; \theta^{self}, \theta^{s})) \tag{2.5}$$

$$y_{self} = \begin{cases} 1, & \text{if } X_l \text{ and } X_r \text{ sampled from same track} \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

  where X_l and X_r are a pair of randomly sampled short music snippets (taken from the 30-second MSD audio previews) and f_{self} is a network for learning a metric between given input representations in terms of the criteria imposed by y_{self}. It is composed of one or more fully-connected layers and one output layer with softmax activation. An global outline illustration of our chosen architecture is given in Fig. 2.3. Further specifications of the representation network and sampling strategies will be given in Section 2.3.

- ***Feature.*** Many algorithms exist already for extracting features out of musical audio, or for transforming musical audio representations. By running such algorithms on musical audio, learning labels are automatically computed, without the need for soliciting human annotations. Algorithmically computed outcomes will likely not be perfect, and include noise or errors. At the same time, we consider them as a relatively efficient way to extract semantically relevant and more structured information out of a raw input signal.

  In our case, under this category, we use Beat Per Minute (BPM) information, released as part of the MSD's precomputed features. The BPM values were computed by an estimation algorithm, as part of the `Echo Nest` API.

Annotation

- ***Metadata.*** Typically, metadata will come 'for free' with music audio, specifying side information, such as a release year, the song title, the name of the artist, the corresponding album name, and the corresponding album cover image. Considering

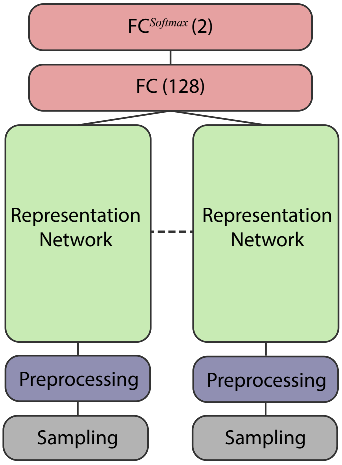


Figure 2.3: Siamese architecture adopted for the *self* learning task. For further details of the Representation Network, see Section 2.3.1 and Fig. 2.4.

that this information describes categorization facets of the musical audio, metadata can be a useful information source to learn a music representation. In our experiments, we use release year information, which is readily provided as metadata with each song in the MSD.

- ***Crowd.*** Through interaction with music streaming or scrobbling services, large numbers of users, also designated as the *crowd*, left explicit or implicit information regarding their perspectives on musical content. For example, they may have created social tags, ratings, or social media mentionings of songs. With many services offering API access to these types of descriptors, crowd data therefore offers scalable, spontaneous and diverse (albeit noisy) human perspectives on music signals.

  In our experiments, we use social tags from `Last.fm`[1] and user listening profiles from the `Echo Nest`.

- ***Professional.*** As mentioned in [2], annotation of music tracks is a complicated and time-consuming process: annotation criteria frequently are subjective, and considerable domain knowledge and annotation experience may be required before accurate and consistent annotations can be made. Professional experts in categorization have this experience, and thus are capable of indicating clean and systematic information about musical content. It is not trivial to get such professional annotations at scale; however, these types of annotations may be available in existing professional libraries.

  In our case, we use professional annotations from the Centrale Discotheek Rotterdam (CDR), the largest music library in The Netherlands, holding all music ever released in the country in physical and digital form in its collection. The CDR collec-

[1] `https://labrosa.ee.columbia.edu/millionsong/lastfm`

tion can be digitally accessed through the online Muziekweb[2] platform. For each musical album in the CDR collection, genre annotations were made by a professional annotator, according to a fixed vocabulary of 367 hierarchical music genres.

As another professional-level 'description', we adopted lyrics information per each track, which is provided in Bag-of-Words format with the MSD. To filter out trivial terms such as stop-words, we applied TF-IDF[40].

- ***Combination.*** Finally, learning labels can be derived from combinations of the above categories. In our experiment, we used combination of artist information and social tags, by making a bag of tags at the artist level as a learning label.

Not all songs in the MSD actually include learning labels from all the sources mentioned above. Clearly, it is another advantage of using MTL that one can use such unbalanced datasets in a single learning procedure, to maximize the coverage of the dataset. However, on the other hand, if one uses an unbalanced number of samples across different learning sources, it is not trivial to compare the effect of individual learning sources. We therefore choose to work with a subset of the dataset, in which equal numbers of samples across learning sources can be used. As a consequence, we managed to collect 46,490 clips of tracks with corresponding learning source labels. A 41,841 / 4,649 split was made for training and validation for all sources from both MSD and CDR. Since we mainly focus on transfer learning, we used the validation set mostly for monitoring the training, to keep the network from overfitting.

[2] `https://www.muziekweb.nl/`

Table 2.1: Properties of learning sources.

Identifier	Category		Data	Dimensionality	Preprocessing
self	Algorithm	Self	MSD - Track	1	
bpm		Feature	MSD - BPM	1	GMM
year	Annotation	Metadata	MSD - Year	1	GMM
tag		Crowd	MSD - Tag	174,156	pLSA
taste		Crowd	MSD - Taste	949,813	pLSA
cdr_tag		Professional	CDR - Tag	367	pLSA
lyrics		Professional	MSD - Lyrics	5,000	pLSA, TF-IDF
artist		Combination	MSD - Artist & Tag	522,366	pLSA

Table 2.2: Examples of Latent Topics extracted with pLSA from MSD social tags

Topic	Strongest social tags
tag1	`indie rock`, `indie`, `british`, `Scottish`
tag2	`pop`, `pop rock`, `dance`, `male vocalists`
tag3	`soul`, `rnb`, `funk`, `Neo-Soul`
tag4	`Melodic Death Metal`, `black metal`, `doom metal`, `Gothic Metal`
tag5	`fun`, `catchy`, `happy`, `Favorite`

2.2.3. LATENT FACTOR PREPROCESSING

Most learning sources are noisy. For instance, social tags include tags for personal playlist management, long sentences, or simply typos, which do not actually show relevant nuances in describing the music signal. The algorithmically extracted BPM information also is imperfect, and likely contains octave errors, in which BPM is under- or overestimated by a factor of 2. To deal with this noise, several previous works using the MSD [17, 27] applied a frequency-based filtering strategy along with top-down domain knowledge. However, this shrinks the available sample size. As an alternative way to handle noisiness, several other previous works [12, 18, 28, 41–43] apply latent factor extraction using various low-rank approximation models to preprocess the label information. We also choose to do this in our experiments.

A full overview of chosen learning sources, their category, origin dataset, dimensionality and preprocessing strategies is shown in Table 2.1. In most cases, we apply probabilistic latent semantic analysis (pLSA), which extracts latent factors as a multinomial distribution of latent topics [44]. Table 2.2 illustrates several examples of strong social tags within extracted latent topics.

For situations in which learning labels are a scalar, non-binary value (BPM and release year), we applied a Gaussian Mixture Model (GMM) to transform each value into a categorical distribution of Gaussian components. In case of the *Self* category, as it basically is a binary membership test, no factor extraction was needed in this case.

After preprocessing, learning source labels y_t are now expressed in the form of probabilistic distributions z_t. Then, the learning of a deep representation can take place by minimizing the Kullback–Leibler (KL) divergence between model inferences $f_t(X)$ and label factor distributions z_t.

Along with the noise reduction, another benefit from such preprocessing is the regularization of the scale of the objective function between different tasks involved in the learning, when the resulting factors have the same size. This regularity between the objective functions is particularly helpful for comparing different tasks and datasets. For this purpose, we used a fixed single value $k = 50$ for the number of factors (pLSA) and the number of Gaussians (GMM). In the remainder of this paper, the datasets and tasks processed in above manner will be denoted by *learning sources* for coherent presentation and usage of the terminology.

2.3. REPRESENTATION NETWORK ARCHITECTURES

In this section, we present the detailed specification of the deep representation neural network architecture we exploited in this work. We will discuss the base architecture of

Table 2.3: Configuration of the base CNN. conv and max-pool indicate a 2-dimensional convolution and max-pooling layer, respectively. We set the stride size with 2 on the time dimension of conv1, to compress dimensionality at the early stage. Otherwise, all strides are set as 1 across all the convolution layers. gap corresponds to the global average pooling used in [23], which averages out all the spatial dimensions of the filter responses. fc is an abbreviation of fully-connected layer. We use dropout with $p = 0.5$ only for the fc-feature layer, where the intermediate latent representation is extracted and evaluated. For simplicity, we omit the batch-size dimension of the input shape.

Layer	Input Shape	Weight Shape	Sub-Sampling	Activation
conv1	2 × 216 × 128	2 × 16 × 5 × 5	2 × 1	ReLU
max-pool1	16 × 108 × 128		2 × 2	
conv2	16 × 54 × 64	16 × 32 × 3 × 3		ReLU
max-pool2	32 × 54 × 64		2 × 2	
conv3	32 × 27 × 32	32 × 64 × 3 × 3		ReLU
max-pool3	64 × 27 × 32		2 × 2	
conv4	64 × 13 × 16	64 × 64 × 3 × 3		ReLU
max-pool4	64 × 13 × 16		2 × 2	
conv5	64 × 6 × 8	64 × 128 × 3 × 3		ReLU
max-pool5	128 × 6 × 8		2 × 2	
conv61	128 × 3 × 4	128 × 256 × 3 × 3		ReLU
conv62	256 × 3 × 4	256 × 256 × 1 × 1		ReLU
gap	256			
fc-feature	256	256 × 256		ReLU
dropout	256			
fc-output	256	learning source specific		Softmax

the network, and further discuss the shared architecture with respect to different fusion strategies that one can take in the MTDTL context. Also, we introduce details on the pre-processing related to the input data served into networks.

2.3.1. BASE ARCHITECTURE

As the deep base architecture for feature representation learning, we choose a Convolutional Neural Network (CNN) architecture inspired by [22], as described in Fig. 2.4 and Table 2.3.

The CNN is one of the most popular architectures in many music-related machine learning tasks [17, 18, 21, 26, 45–56]. Many of these works adopt an architecture having cascading blocks of 2-dimensional filters and max-pooling, derived from well-known works in image recognition [22, 57]. Although variants of CNN using 1-dimensional filters also were suggested by [13, 58–60] to learn features directly from a raw audio signal in an end-to-end manner, not many works managed to use them on music classification tasks successfully [61].

The main difference between the base architecture and [22] is the use of Global Average Pooling (GAP) and the Batch Normalization (BN) layers. BN is applied to accelerate the training and stabilize the internal covariate shift for every convolution layer and the fc-feature layer [62]. Also, global spatial pooling is adopted as the last pooling layer of the cascading convolution blocks, which is known to effectively summarize the spatial dimensions both in the image [23] and music domain [21]. We also applied the approach to ensure the `fc-feature` layer not to have a huge number of parameters.

We applied the Rectified Linear Unit (ReLU) [63] to all convolution layers and the `fc-feature` layer. For the `fc-output` layer, softmax activation is used. For each convolution layer, we applied zero-padding such that the input and the output have the same spatial shape. As for the regularization, we choose to apply drop-out [64] on the `fc-feature` layer. We added $L2$ regularization across all the parameters with the same weight $\lambda = 10^{-6}$.

Audio Preprocessing

We aim to learn a music representation from as-raw-as-possible input data to fully leverage the capability of the neural network. For this purpose, we use the dB-scale mel-scale magnitude spectrum of an input audio fragment, extracted by applying 128-band mel-filter banks on the Short-Time Fourier Transform (STFT). mel-spectrograms have generally been a popular input representation choice for CNNs applied in music-related tasks [17, 18, 21, 27, 42, 65]; besides, it also was reported recently that their frequency-domain summarization, based on psycho-acoustics, is efficient and not easily learnable through data-driven approaches [66, 67]. We choose a 1024-sample window size and 256-sample hop size, translating to about 46 ms and 11.6 ms respectively for a sampling rate of 22 kHz. We also applied standardization to each frequency band of the mel spectrum, making use of the mean and variance of all individual mel spectra in the training set.

Sampling

During the learning process, in each iteration, a random batch of songs is selected. Audio corresponding to these songs originally is 30 seconds in length; for computational efficiency, we randomly crop 2.5 seconds out of each song each time. Keeping stereo channels of the audio, the size of a single input tensor X^* we used for the experiment ended up with $2 \times 216 \times 128$, where the first dimension indicates number of channels, and following dimensions mean time steps and mel-bins, respectively. Along with the computational efficiency, a number of literatures in MIR field reported that using a small chunk of the input not only inflates the dataset, but also shows good performance on the high-level tasks such as music auto-tagging [21, 58, 61]. For the *self* case, we generate batches with equal numbers of songs for both membership categories in y_{self}.

2.3.2. Multi-Source Architectures with Various Degrees of Shared Information

When learning a music representation based on various available learning sources, different strategies can be taken regarding the choice of architecture. We will investigate the following setups:

- As a base case, a ***Single-Source Representation (SS-R)*** can be learned for a single source only. As mentioned earlier, this would be the typical strategy leading to pre-trained networks, that later would be used in transfer learning. In our case, our base architecture from Section 2.3.1 and Fig. 2.4 will be used, for which the layers in the Representation Network also are illustrated in Fig. 2.5a. Out of the `fc-feature` layer, a d-dimensional representation is obtained.

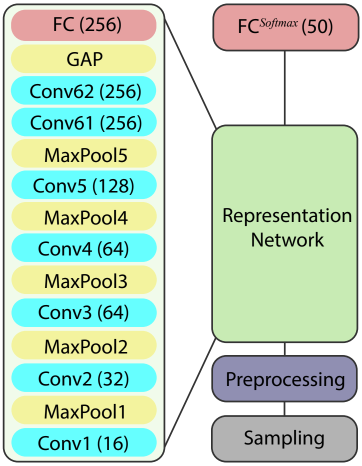


Figure 2.4: Default CNN architecture for supervised single-source representation learning. Details of the Representation Network are presented at the left of the global architecture diagram. The numbers inside the parentheses indicate either the number of filters, or the number of units with respect to the type of layer.

- If multiple perspectives on the same content, as reflected by the multiple learning labels, should also be reflected in the ultimate learned representation, one can learn *SS-R* representations for each learning source, and simply concatenate them afterwards. With d dimensions per source and m sources, this leads to a $d \times m$ ***Multiple Single-Source Concatenated Representation (MSS-CR)***. In this case, independent networks are trained for each of the sources, and no shared knowledge will be transferred between sources. A layer setup of the corresponding Representation Network is illustrated in Fig. 2.5b.

- When applying MTL learning strategies, the deep architecture should involve shared knowledge layers, before branching out to various individual learning sources, whose learned representations will be concatenated in the final $d \times m$-dimensional representation. We call these ***Multi-Source Concatenated Representations (MS-CR)***. As the branching point can be chosen at different stages, we will investigate the effect of various prototypical branching point choices: at the second convolution layer (*MS-CR@2*, Fig. 2.5c), the fourth convolution layer (*MS-CR@4*, Fig. 2.5d), and the sixth convolution layer (*MS-CR@6*, Fig. 2.5e). The later the branching point occurs, the more shared knowledge the network will employ.

- In the most extreme case, branching would only occur at the very last fully connected layer, and a **Multi-Source Shared Representation (MS-SR)** (or, more specifically, *MS-SR@FC*) is learned, as illustrated in Fig. 2.5f. As the representation is obtained from the `fc-feature` layer, no concatenation takes place here, and a d-dimensional representation is obtained.

A summary of these different representation learning architectures is given in Table 2.4. Beyond the strategies we choose, further approaches can be thought of to connect rep-

Table 2.4: Properties of the various categories of representation learning architectures.

	Multi Source	Shared Network	Concatenation	Dimensionality
SS-R	No	No	No	d
MSS-CR	Yes	No	Yes	$d \times m$
MS-CR	Yes	Partial	Yes	$d \times m$
MS-SR	Yes	Yes	No	d

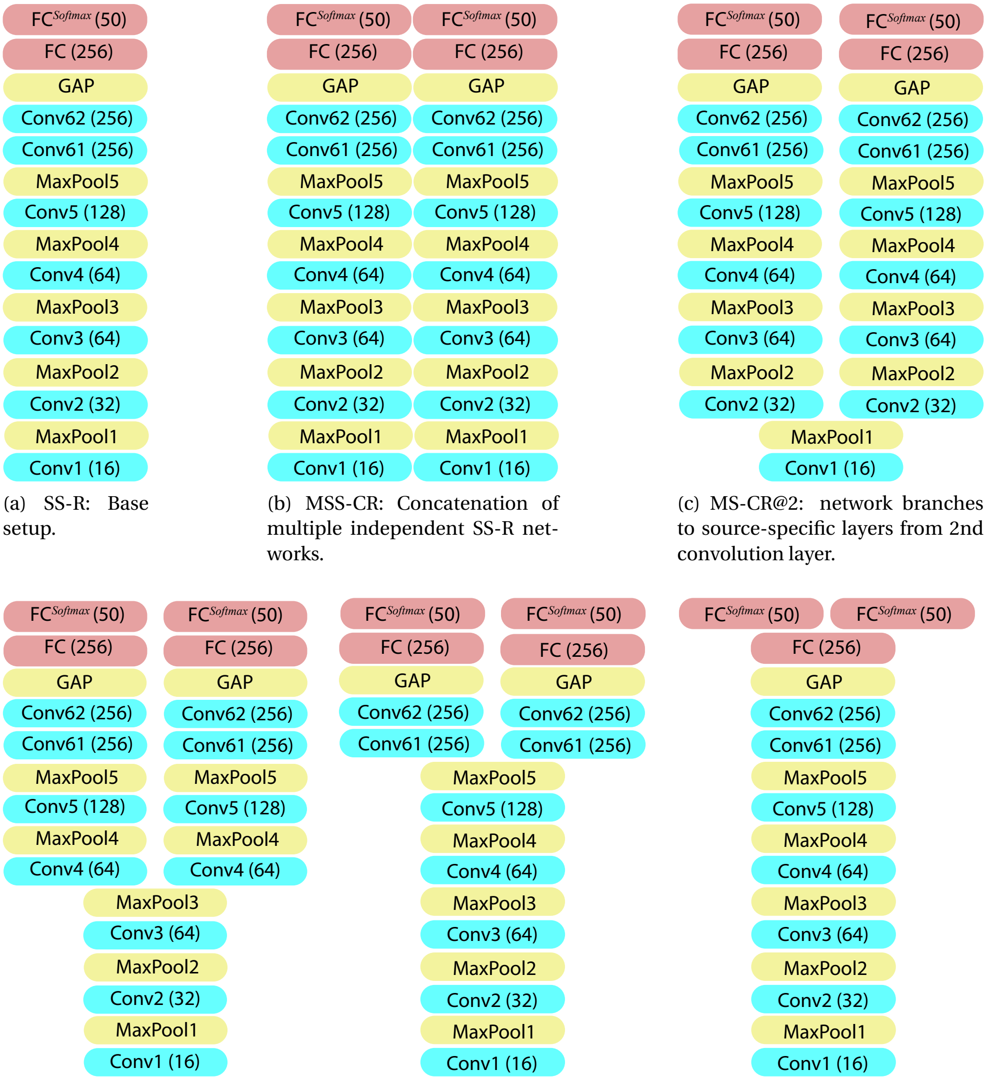
resentations learned for different learning sources in neural network architectures. For example, for different tasks, representations can be extracted from different intermediate hidden layers, benefiting from the hierarchical feature encoding capability of the deep network [27]. However, considering that learned representations are usually taken from a specific fixed layer of the shared architecture, we focus on the strategies as we outlined above.

2.3.3. MTL Training Procedure

Algorithm 1: Training a Multi-Source CNN

```
Initialize $\Theta$: $\{\theta^t, \theta^s\}$ randomly;
for epoch in 1...N do
    for iteration in 1...L do
        Pick a learning source $t$ randomly;
        Pick batch of samples from learning source $t$;
        $(X_l, X_r)$ for self;
        $X$ otherwise;
        Derive learning label $z_t$;
        Sub-sample chunk $X^*$ from track $X$;
        Forward-pass:;
        $\mathcal{L}(y_{self}, \Theta, X_l^*, X_r^*)$ =Eq. 2.5 for self;
        $\mathcal{L}(z_t, \Theta, X^*)$ =Eq. 2.2 otherwise;
        Backward-pass: $\nabla(\Theta)$;
        Update model: $\Theta \leftarrow \Theta - \epsilon\nabla(\Theta)$;
```

Similar to [5, 12], we choose to train the MTL models with a stochastic update scheme as described in Algorithm 1. At every iteration, a learning source is selected randomly. After the learning source is chosen, a batch of observation-label pairs (X, z_t) is drawn. For the audio previews belonging to the songs within this batch, an input representation X^* is cropped randomly from its super-sample X. The updates of the parameters Θ are conducted through back-propagation using the Adam algorithm [68]. For each neural network we train, we set $L = lm$, where l is the number of iterations needed to visit all the training samples with fixed batch size $b = 128$, and m is the number of learning sources used in the training. Across the training, we used a fixed learning rate $\epsilon = 0.00025$. After a



(a) SS-R: Base setup.

(b) MSS-CR: Concatenation of multiple independent SS-R networks.

(c) MS-CR@2: network branches to source-specific layers from 2nd convolution layer.

(d) MS-CR@4: network branches to source-specific layers from 4th convolution layer.

(e) MS-CR@6: network branches to source-specific layers from 6th convolution layer.

(f) MS-SR@FC: heavily shared network, source-specific branching only at final FC layer.

Figure 2.5: The various model architectures considered in the current work. Beyond single-source architectures, multi-source architectures with various degrees of shared information are studied. For simplification, multi-source cases are illustrated here for two sources. The `fc-feature` layer from which representations will be extracted is the FC(256) layer in the illustrations (see Table 2.3).

fixed number of epochs N is reached, we stop the training.

2.3.4. Implementation Details

We used *PyTorch* [69] to implement the CNN models and parallel data serving. For evaluation of models and cross-validation, we made extensive use of functionality in *Scikit-Learn* [70]. Furthermore, *Librosa* [71] was used to process audio files and its raw features including mel spectrograms. The training is conducted with 8 Graphical Processing Unit (GPU) computation nodes, composed of 2 NVIDIA GRID K2 GPUs and 6 NVIDIA GTX 1080Ti GPUs.

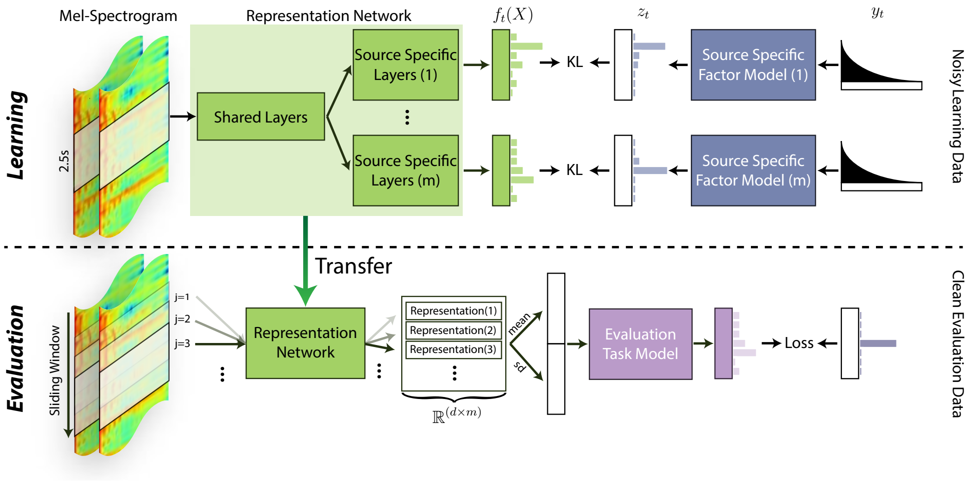


Figure 2.6: Overall system framework. The first row of the figure illustrates the learning scheme, where the representation learning is happening by minimizing the KL divergence between the network inference $f_t(X)$ and the preprocessed learning label z_t. The preprocessing is conducted by the blue blocks which transform the original noisy labels y_t to z_t, reducing noise and summarizing the high-dimensional label space into a smaller latent space. The second row describes the entire evaluation scenario. The representation is first extracted from the representation network, which is transferred from the upper row. The sequence of representation vectors is aggregated as the concatenation of their means and standard deviations. The purple block indicates a machine learning model employed to evaluate the representation's effectiveness.

2.4. Evaluation

So far, we discussed the details regarding the learning phase of this work, which corresponds to the upper row of Fig. 2.6. This included various choices of sources for the representation learning, and various choices of architecture and fusion strategies. In this section, we present the evaluation methodology we followed, as illustrated in the second row of Fig. 2.6. First, we will discuss the chosen target tasks and datasets in Section 2.4.1, followed in Section 2.4.2 by the baselines against which our representations will be compared. Section 2.4.3 explains our experimental design, and finally we discuss the implementation of our evaluation experiments in Section 2.4.4.

2.4.1. Target Datasets

In order to gain insight into the effectiveness of learned representations with respect to multiple potential future tasks, we consider a range of *target datasets*. In this work, our target datasets are chosen to reflect various semantic properties of music, purposefully chosen semantic biases, or popularity in the MIR literature. Furthermore, the representation network should not be configured or learned to explicitly solve the chosen target datasets.

While for the learning sources, we could provide categorizations on where and how the learning labels were derived, and also consider algorithmic outcomes as labels, existing popular research datasets mostly fall in the *Professional* or *Crowd* categories. In our work, we choose 7 evaluation datasets commonly used in MIR research, which reflect three conventional types of MIR tasks, namely classification, regression and recommendation:

Table 2.5: Properties of target datasets used in our experiments. Because of time constraints, we sampled the Lastfm dataset as described in Section 2.4.1; the original size appears between parentheses. In case particular data splits are defined by an original author or follow up study, we apply the same split, including the reference in which the split is introduced. Otherwise, we applied either a random split stratified by the label (Ballroom), or simple filtering based on reported faulty entries (IRMAS).

Task	Data		#Tracks	#Class	Split Method
Classification	FMA[72]	Genre	25,000	16	Artist Filtered [72]
Classification	GTZAN[73]	Genre	1,000	10	Artist Filtered [74]
Classification	Ext. Ballroom[75, 76]	Genre	3,390	13	N/A
Classification	IRMAS[77]	Instrument	6,705	11	Song Filtered
Regression	Music Emotion[78]	Arousal	744		Genre Stratified[78]
Regression	Music Emotion[78]	Valence	744		Genre Stratified[78]
Recommendation	Lastfm*[79]	Listening Count	27,093 (961,416)		N/A

- ***Classification.*** Different types of classification tasks exist in MIR. In our experiments, we consider several datasets used for genre classification and instrument classification.

  For genre classification, we chose the GTZAN [73] and FMA [72] datasets as main exemplars. Even though GTZAN is known for its caveats [80], we deliberately used it, because its popularity can be beneficial when comparing with previous and future work. We note though that there may be some overlap between the tracks of GTZAN and the subset of the MSD we use in our experiments; the extent of this overlap is unknown, due to the lack of a confirmed and exhaustive track listing of the GTZAN dataset. We choose to use a fault-filtered data split for the training and evaluation, which is suggested in [74]. The split originally includes a training, validation and evaluation split; in our case, we also included the validation split as training data.

  Among the various packages provided by the FMA, we chose the top-genre classification task of FMA-Medium [72]. This is a classification dataset with an unbalanced genre distribution. We used the data split provided by the dataset for our experiment, where the training is validation set are combined as the training.

  Considering another type of genre classification, we selected the Extended Ballroom dataset [75, 76]. Because the classes in this dataset are highly separable with regard to their BPM [81], we specifically included this 'purposefully biased' dataset as an example of how a learned representation may effectively capture temporal dynamics properties present in a target dataset, as long as learning sources also reflected these properties. Since no pre-defined split is provided or suggested by other literature, we used stratified random sampling based on the genre label.

  The last dataset we considered for classification is the training set of the IRMAS dataset [77], which consists of short music clips annotated with the predominant instruments present in the clip. Compared to the genre classification task, instrument classification is generally considered as less subjective, requiring features to separate timbral characteristics of the music signal as opposed to high-level semantics like genre. We split the dataset to make sure that observations from the same music track are not split into training and test set.

  As performance metric for all these classification tasks, we used classification accuracy.

- ***Regression.*** As exemplars of regression tasks, we evaluate our proposed deep representations on the dataset used in the MediaEval Music Emotion prediction task [78]. It contains frame-level and song-level labels of a two-dimensional representation of emotion, with valence and arousal as dimensions [82]. Valence is related to the positivity or negativity of the emotion, and arousal is related to its intensity [78]. The song-level annotation of the V-A coordinates was used as the learning label. In similar fashion to the approach taken in [27], we trained separate models for the two emotional dimensions. As for the dataset split, we used the split provided by the dataset, which is done by the random split stratified by the genre distribution.

  As evaluation metric, we measured the coefficient of determination R^2 of each model.

- ***Recommendation.*** Finally, we employed the 'Last.fm - 1K users' dataset [79] to evaluate our representations in the context of a content-aware music recommendation task (which will be denoted as *Lastfm* in the remaining of the paper). This dataset contains 19 million records of listening events across 961,416 unique tracks collected from 992 unique users. In our experiments, we mimicked a cold-start recommendation problem, in which items not seen before should be recommended to the right users. For efficiency, we filtered out users who listened to less than 5 tracks and tracks known to less than 5 users.

  As for the audio content of each track, we obtained the mapping between the MusicBrainz Identifier (MBID) with the Spotify identifier (SpotifyID) using the `MusicBrainz API`[3]. After cross-matching, we collected 30 seconds previews of all track using the `Spotify API`[4]. We found that there is a substantial amount of missing mapping information between the SpotifyID and MBID in the `MusicBrainz` database, where only approximately 30% of mappings are available. Also, because of the substantial amount of inactive users and unpopular tracks in the dataset, we ultimately acquired a dataset of 985 unique users and 27,093 unique tracks with audio content.

  Similar to [29], we considered the *outer matrix* performance for un-introduced songs; in other words, the model's recommendation accuracy on the items newly introduced to the system [29]. This was done by holding out certain tracks when learning user models, and then predicting user preference scores based on all tracks, including those that were held out, resulting in a ranked track list per user. As evaluation metric, we consider Normalized Discounted Cumulative Gain (*nDCG*@500), only treating held-out tracks that were indeed liked by a user as relevant items. Further details on how hold-out tracks were chosen are given in Section 2.4.4.

A summary of all evaluation datasets, their origins and properties, can be found in Table 2.5.

2.4.2. Baselines

We examined three baselines to compare with our proposed representations:

- ***Mel-Frequency Cepstral Coefficients (MFCC).*** These are some of the most popular audio representations in MIR research. In this work, we extract and aggregate MFCC following the strategy in [27]. In particular, we extracted 20 coefficients and also used their first- and second-order derivatives. After obtaining the sequence of MFCCs and its derivatives, we performed aggregation by taking the average and standard deviation over the time dimension, resulting in a 120-dimensional vector representation.

- ***Random Network Feature (Rand).*** We extracted the representation at the `fc-feature` layer without any representation network training. With random initialization, this representation therefore gives a random baseline for a given CNN architecture. We refer to this baseline as *Rand*.

[3] `https://musicbrainz.org/`

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

- ***Latent Representation from Music Auto-Tagger (Choi).*** The work in [27] focused on a music auto-tagging task, and can be considered as yielding a state-of-the-art deep music representation for MIR. While the model's focus on learning a representation for music auto-tagging can be considered as our *SS-R* case, there are a number of issues that complicate direct comparisons between this work and ours. First, the network in [27] is trained with about 4 times more data samples than in our experiments. Second, it employed a much smaller network than our architecture. Further, intermediate representations were extracted, which is out of the scope of our work, as we only consider representations at the `fc-feature` layer. Nevertheless, despite these caveats, the work still is very much in line with ours, making it a clear candidate for comparison. Throughout the evaluation, we could not fully reproduce the performance reported in the original paper [27]. When reporting our results, we therefore will report the performance we obtained with the published model, referring to this as *Choi*.

2.4.3. Experimental Design

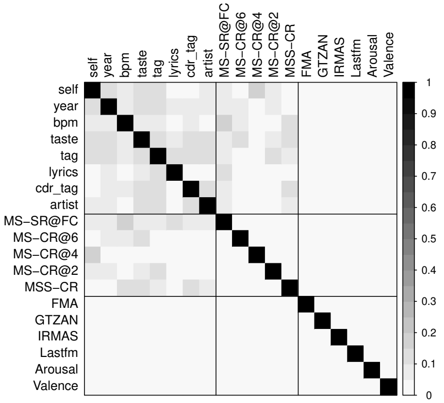


Figure 2.7: Aliasing among main effects in the final experimental design.

In order to investigate our research questions, we carried out an experiment to study the effect of the number and type of learning sources on the effectiveness of deep representations, as well as the effect of the various architectural learning strategies described in Section 2.3.2. For the experimental design we consider the following factors:

- Representation strategy, with 6 levels: *SS-R*, *MS-SR@FC*, *MS-CR@6*, *MS-CR@4*, *MS-CR@2*, and *MSS-CR*).

- 8 2-level factors indicating the presence or not of each of the 8 learning sources: *self*, *year*, *bpm*, *taste*, *tag*, *lyrics*, *cdr_tag* and *artist*.

- Number of learning sources present in the learning process (1 to 8). Note that this is actually calculated as the sum of the eight factors above.

- Target dataset, with 7 levels: Ballroom, FMA, GTZAN, IRMAS, Lastfm, Arousal and Valence.

Given a learned representation, fitting dataset-specific models is much more efficient than learning the representation, so we decided to evaluate each representation on all 7 target datasets. The experimental design is thus restricted to combinations of representation and learning sources, and for each such combination we will produce 7 observations. However, given the constraint of *SS-R* relying on a single learning source, that there is only one possible combination for n = 8 sources, as well as the high unbalance in the number of sources[5], we proceeded in three phases:

1. We first trained the *SS-R* representations for each of the 8 sources, and repeated 6 times each. This resulted in 48 experimental runs.

2. We then proceeded to train all five multi-source strategies with all sources, that is, $n = 8$. We repeated this 5 times, leading to 25 additional experimental runs.

3. Finally, we ran all five multi-source strategies with $n = 2, \ldots, 7$. The full design matrix would contain 5 representations and 8 sources, for a total of 1,230 possible runs. Such an experiment was unfortunately infeasible to run exhaustively given available resources, so we decided to follow a fractional design. However, rather than using a pre-specified optimal design with a fixed amount of runs [84], we decided to run sequentially for as long as time would permit us, generating at each step a new experimental run on demand in a way that would maximize desired properties of the design up to that point, such as balance and orthogonality[6].

   We did this with the greedy Algorithm 2. From the set of still remaining runs $\mathscr{A}$, a subset $\mathscr{O}$ is selected such that the expected unbalance in the augmented design $\mathscr{B} \cup \{o\}$ is minimal. In this case, the unbalance of a design is defined as the maximum unbalance found between the levels of any factor, except for those already exhausted[7]. From $\mathscr{O}$, a second subset $\mathscr{P}$ is selected such that the expected aliasing in the augmented design is minimal, here defined as the maximum absolute aliasing between main effects[8]. Finally, a run p is selected at random from $\mathscr{P}$, the corresponding representation is learned, and the algorithm iterates again after updating $\mathscr{A}$ and $\mathscr{B}$.

   Following this on demand methodology, we managed to run another 352 experimental runs from all the 1,230 possible.

[5] For instance, from the 255 possible combinations of up to 8 sources, there are 70 combinations of $n = 4$ sources, but 28 with $n = 2$, or only 8 for $n = 7$. Simple random sampling from the 255 possible combinations would lead to a very unbalanced design, that is, a highly non-uniform distribution of observation counts across the levels of the factor (n in this case). A balanced design is desired to prevent aliasing and maximize statistical power. See section 15.2 in [83] for details on unbalanced designs.

[6] An experimental design is orthogonal if the effects of any factor balance out across the effects of the other factors. In a non-orthogonal design effects may be aliased, meaning that the estimate of one effect is partially biased with the effect of another, the extent of which ranges from 0 (no aliasing) to 1 (full aliasing). Aliasing is sometimes referred to as confounding. See sections 8.5 and 9.5 in [83] for details on aliasing.

[7] For instance, let a design have 20 runs for *SS-R*, 16 for *MS-SR@FC*, and 18 for all other representations. The unbalance in the representation factor is thus 20 − 16 = 4. The total unbalance of the design is defined as the maximum unbalance found across all factors.

[8] See section 2.3.7 in [84] for details on how to compute an alias matrix.

Algorithm 2: Sequential generation of experimental runs.

```
Initialize $\mathscr{A}$ with all possible 1,230 runs to execute;
Initialize $\mathscr{B} \leftarrow \emptyset$ for the set of already executed runs;
while time allows do
    Select $\mathscr{O} \subseteq \mathscr{A}$ s.t. $\forall o \in \mathscr{O}$, the unbalance in $\mathscr{B} \cup \{o\}$ is minimal;
    Select $\mathscr{P} \subseteq \mathscr{O}$ s.t. $\forall p \in \mathscr{P}$, the aliasing in $\mathscr{B} \cup \{p\}$ is minimal;
    Select $p \in \mathscr{P}$ at random;
    Update $\mathscr{A} \leftarrow \mathscr{A} - \{p\}$;
    Update $\mathscr{B} \leftarrow \mathscr{B} \cup \{p\}$;
    Learn the representation coded by $p$;
```

After going through the three phases above, the final experiment contained $48+25+352=425$ experimental runs, each producing a different deep music representation. We further evaluated each representation on all 7 target datasets, leading to a grand total of $42 \times 7 = 2,975$ datapoints. Fig. 2.7 plots the alias matrix of the final experimental design, showing that the aliasing among main factors is indeed minimal. The final experimental design matrix can be downloaded along with the rest of the supplemental material.

Each considered representation network was trained using the CNN representation network model from Section 2.3, based on the specific combination of learning sources and deep architecture as indicated by the experimental run. In order to reduce variance, we fixed the number of training epochs to $N = 200$ across all runs, and applied the same base architecture, except for the branching point. This entire training procedure took approximately 5 weeks with given computational hardware resources introduced in Section 2.3.4.

2.4.4. IMPLEMENTATION DETAILS

In order to assess how our learned deep music representations perform on the various target datasets, transfer learning will now be applied, to consider our representations in the context of these new target datasets.

As a consequence, new machine learning pipelines are set up, focused on each of the target datasets. In all cases, we applied the pre-defined split if it is feasible. Otherwise, we randomly split the dataset in a 80% training and 20% test set. For every dataset, we repeated the training and evaluation for 5 times, using different train/test splits. In most of our evaluation cases, validation will take place on the test set; in case of the the recommendation problem, the test set represents a set of tracks to be held out during user model training, and re-inserted for validation. In all cases, we will extract representations from evaluation dataset audio as detailed in Section 2.4.4, and then learn relatively simple models based on them, as detailed in Section 2.4.4. Employing the metrics as mentioned in the previous section, we will then take average performance scores over the 5 different train-test splits for final performance reporting.

FEATURE EXTRACTION AND PREPROCESSING

Taking raw audio from the evaluation datasets as input, we take non-overlapping slices out of this audio with a fixed length of 2.5 seconds. Based on this, we apply the same preprocessing transformations as discussed in Section 2.3.1. Then, we extract a deep representation from this preprocessed audio, employing the architecture as specified by the given experimental run. As in the case of Section 2.3.2, representations are extracted from the `fc-feature` layer of each trained CNN model. Depending on the choice of architecture, the final representation may consist of concatenations of representations obtained by separate representation networks.

Input audio may originally be (much) longer than 2.5 seconds; therefore, we aggregate information in feature vectors over multiple time slices by taking their *mean* and *standard deviation* values. As a result, we get a representation with averages per learned feature dimension, and another representation with standard deviations per feature dimension. These will be concatenated, as illustrated in Fig. 2.6.

Target Dataset-Specific Models

As our goal is not to over-optimize dataset-specific performance, but rather perform a comparative analysis between different representations (resulting from different learning strategies), we keep the model simple, and use fixed hyper-parameter values for each model across the entire experiment.

To evaluate the trained representations, we used different models according to the target dataset. For classification and regression tasks, we used Multi Layer Perceptron (MLP) model [85]. More specifically, the MLP model has two hidden layers, whose dimensionality is 257. As for the non-linearity, we choose ReLU [63] for all nodes, and the model is trained with ADAM optimization technique [68] for 200 iterations. In evaluation, we used the *Scikit-Learn*'s implementation for ease of distributed computing on multiple CPU computation nodes.

For the recommendation task, we choose a similar model as suggested in [29, 86], in which the learning objective function $\mathscr{L}$ is defined as

$$\hat{U}, \hat{V}, \hat{W} = \arg\min ||P - UV^T||_C + \frac{\lambda^V}{2}||V - XW|| + \frac{\lambda^U}{2}||U|| + \frac{\lambda^W}{2}||W|| \tag{2.7}$$

where $P \in \mathbb{R}^{u \times i}$ is a binary matrix indicating whether there is interaction between users u and items i, $U \in \mathbb{R}^{u \times r}$ and $V \in \mathbb{R}^{i \times r}$ are r dimensional user factors and item factors for the low-rank approximation of P. P is derived from the original interaction matrix $R \in \mathbb{R}^{u \times i}$, which contains the number of interaction from users u to items i, as follows:

$$P_{u,i} = \begin{cases} 1, & \text{if } R_{u,i} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

$W \in \mathbb{R}^{d \times r}$ is a free parameter for the projection from d-dimensional feature space to the factor space. $X \in \mathbb{R}^{i \times d}$ is the feature matrix where each row corresponds to a track. Finally, $||\cdot||_C$ is the Frobenious norm weighted by the confidence matrix $C \in \mathbb{R}^{u \times i}$, which controls the credibility of the model on the given interaction data, given as follows:

$$C = 1 + \alpha R \tag{2.9}$$

where α controls credibility. As for hyper-parameters, we set $\alpha = 0.1$, $\lambda^V = 0.00001$, $\lambda^U = 0.00001$, and $\lambda^W = 0.1$, respectively. For the number of factors we choose $r = 50$ to

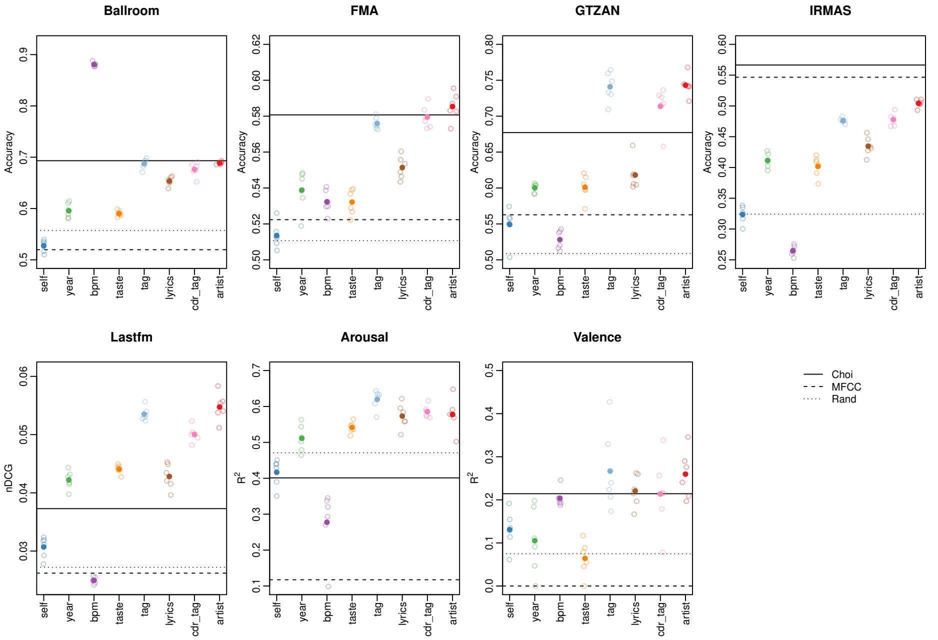


Figure 2.8: Performance of single source representations. Each point indicates the performance of a representation learned from the single source. Solid points indicate the average performance per source. The baselines are illustrated as horizontal lines.

focus only on the relative impact of the representation over the different conditions. We implemented an update rule with the Alternating Least Squares (ALS) algorithm similar to [29], and updated parameters during 15 iterations.

2.5. RESULTS AND DISCUSSION

In this section, we present results and discussion related to the proposed deep music representations. In Section 2.5.1, we will first compare the performance across the *SS-R*s, to show how different individual learning sources work for each target dataset. Then, we will present general experimental results related to the performance of the multi-source representations. In Section 2.5.2, we discuss the effect of the number of learning sources exploited in the representation learning, in terms of their general performance, reliability, and model compactness. In Section 2.5.3, we discuss effectiveness of different representations in MIR. Finally, we present some initial evidence for multifaceted semantic explainability of the proposed MTDTL in Section 2.5.5.[9]

2.5.1. Single-Source and Multi-Source Representation

Fig. 2.8 presents the performance of *SS-R* representations on each of the 7 target datasets. We can see that all sources tend to outperform the *Rand* baseline on all datasets, except for a handful cases involving sources *self* and *bpm*. Looking at the top performing sources, we find that *tag*, *cdr_tag* and *artist* perform better or on-par with the most sophisticated baseline, *Choi*, except for the IRMAS dataset. The other sources are found somewhere between these two baselines, except for datasets Lastfm and Arousal, where they perform better than *Choi* as well. Finally, the *MFCC* is generally outperformed in all cases, with the notable exception of the IRMAS dataset, where only *Choi* performs better.

Zooming in to dataset-specific observed trends, the *bpm* learning source shows a highly skewed performance across target datasets: it clearly outperforms all other learning sources in the Ballroom dataset, but it achieves the worst or second worst performance in the other datasets. As shown in [81], this confirms that the Ballroom dataset is well-separable based on BPM information alone. Indeed, representations trained on the *bpm* learning source seem to contain a latent representation close to the BPM of an input music signal. In contrast, we can see that the *bpm* representation achieves the worst results in the Arousal dataset, where both temporal dynamics and BPM are considered as important factors determining the intensity of emotion.

On the IRMAS dataset, we see that all the *SS-R*s perform worse than the *MFCC* and *Choi* baselines. Given that they both take into account low-level features, either by design or by exploiting low-level layers of the neural network, this suggests that predominant instrument sounds are harder to distinguish based solely on semantic features, which is the case of the representations studied here.

Also, we find that there is small variability for each *SS-R* run within the training setup we applied. Specifically, in 50% of cases we have within-*SS-R* variability less than 15% of the within-dataset variability. 90% of the cases are within 30% of the within-dataset variability.

We now consider how the various representations based on multiple learning sources perform, in comparison to those based on single learning sources. The boxplots in Fig. 2.9 show the distributions of performance scores for each architectural strategy and per target dataset. For comparison, the gray boxes summarize the distributions depicted in Fig. 2.8, based on the *SS-R* strategy. In general, we can see that these *SS-R* obtain the lowest scores, followed by *MS-SR@FC*, except for the IRMAS dataset. Given that these representations have the same dimensionality, these results suggest that adding a single source-specific layer on top of a heavily shared model may help improving the adaptability of the neural network models, especially when there is no prior knowledge regarding the well-matching learning sources for the target datasets. The *MS-CR* and *MSS-CR* representations obtain the best results in general, which is somewhat expected because of their larger dimensionality.

2.5.2. Effect of Number of Learning Sources and Fusion Strategy

While the plots in Fig. 2.9 suggest that *MSS-CR* and *MS-CR* are the best strategies, the high observed variability makes this statement still rather unclear. In order to gain better

[9] For the reproducibility, we release all relevant materials including code, models and extracted features at `https://github.com/eldrin/MTLMusicRepresentation-PyTorch`.

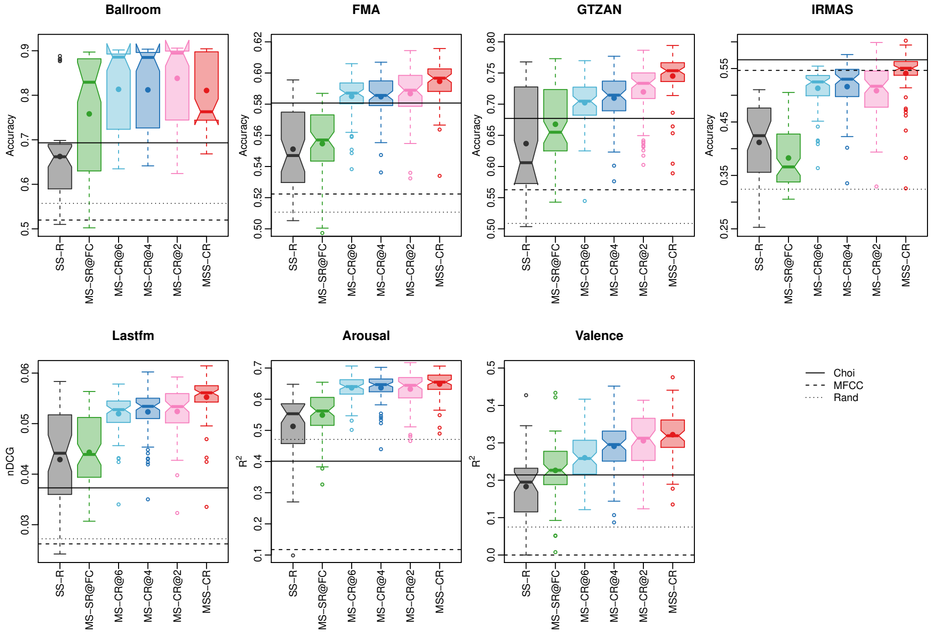


Figure 2.9: Performance by representation strategy. Solid points represent the mean per representation. The baselines are illustrated as horizontal lines.

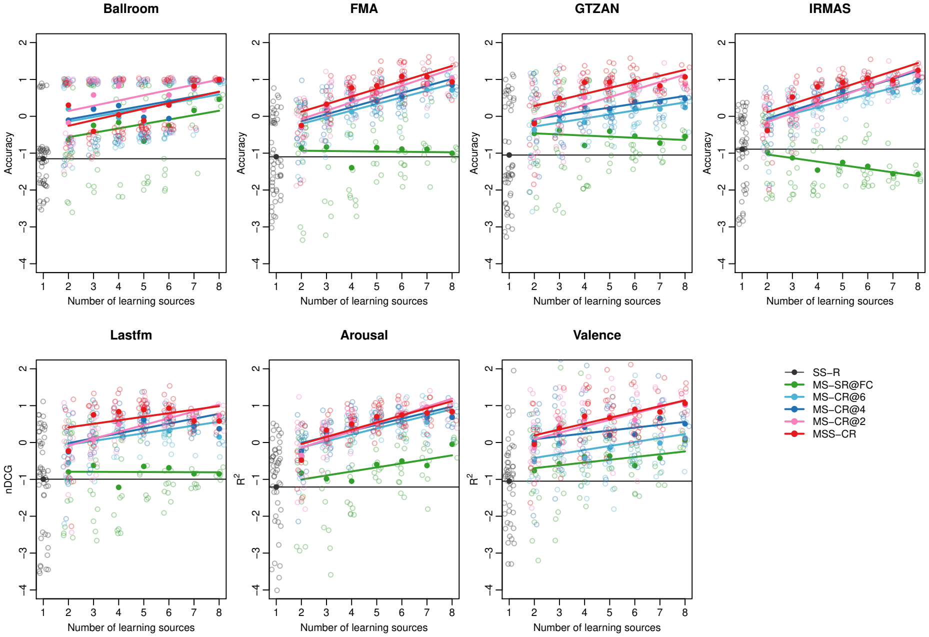


Figure 2.10: (Standardized) Performance by number of learning sources. Solid points represent the mean per architecture and number of sources. The black horizontal line marks the mean performance of the *SS-R* representations. The colored lines show linear fits.

insight of the effects of dataset, architecture strategies and number and type of learning sources, we further analyzed the results using a hierarchical or multilevel linear model on all observed scores [87]. The advantage of such a model is essentially that it accounts for the structure in our experiment, where observations nested within datasets are not independent.

By Fig. 2.9 we can anticipate a very large dataset effect because of the inherently different levels of difficulty, as well as a high level of heteroskedasticity. We therefore analyzed standardized performance scores rather than raw scores. In particular, the i-th performance score y_i is standardized with the within-dataset mean and standard deviation scores, that is, $y_i^* = (y_i - \bar{y}_{d[i]})/s_{d[i]}$, where $d[i]$ denotes the dataset of the i-th observation. This way, the dataset effect is effectively 0 and the variance is homogeneous. In addition, this will allow us to compare the relative differences across strategies and number of sources using the same scale in all datasets.

We also transformed the variable n that refers to the number of sources to n^*, which is set to $n^* = 0$ for *SS-R*s and to $n^* = n-2$ for the other strategies. This way, the intercepts of the linear model will represent the average performance of each representation strategy in its simplest case, that is, *SS-R* ($n = 1$) or non-*SS-R* with $n = 2$. We fitted a first analysis model as follows:

$$y_i^* = \beta_{0r[i]d[i]} + \beta_{1r[i]d[i]} \cdot n_i^* + e_i \qquad e_i \sim N(0, \sigma_e^2) \tag{2.10}$$

$$\beta_{0rd} = \beta_{0r} + u_{0rd} \qquad u_{0rd} \sim N(0, \sigma_{0r}^2) \tag{2.11}$$

$$\beta_{1rd} = \beta_{1r} + u_{1rd} \qquad u_{1rd} \sim N(0, \sigma_{1r}^2), \tag{2.12}$$

where $\beta_{0r[i]d[i]}$ is the intercept of the corresponding representation strategy within the corresponding dataset. Each of these coefficients is defined as the sum of a global fixed effect β_{0r} of the representation, and a random effect u_{0rd} which allows for random within-dataset variation[10]. This way, we separate the effects of interest (ie. each β_{0r}) from the dataset-specific variations (ie. each u_{0rd}). The effect of the number of sources is similarly defined as the sum of a fixed representation-specific coefficient β_{1r} and a random dataset-specific coefficient u_{1rd}. Because the slope depends on the representation, we are thus implicitly modeling the interaction between strategy and number of sources, which can be appreciated in Fig. 2.10, specially with *MS-SR@FC*.

Fig. 2.11 shows the estimated effects and bootstrap 95% confidence intervals. The left plot confirms the observations in Fig. 2.9. In particular, they confirm that *SS-R* performs significantly worse than *MS-SR@FC*, which is similarly statistically worse than the others. When carrying out pairwise comparisons, *MSS-CR* outperforms all other strategies except *MS-CR@2* ($p = 0.32$), which ourperforms all others except *MS-CR@6* ($p = 0.09$). The right plot confirms the qualitative observation from Fig. 2.10 by showing a significantly positive effect of the number of sources except for *MS-SR@FC*, where it is not statistically different from 0. The intervals suggest a very similar effect in the best representations, with average increments of about 0.16 per additional source —recall that scores are standardized.

To gain better insight into differences across representation strategies, we used a second hierarchical model where the representation strategy was modeled as an ordinal vari-

[10] We note that hierarchical models do not fit each of the individual u_{0rd} coefficients (a total of 42 in this model), but the amount of variability they produce, that is, σ_{0r}^2 (6 in total).

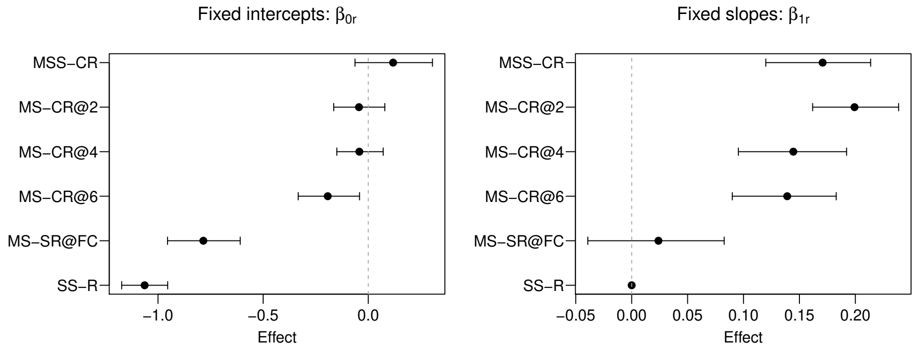


Figure 2.11: Fixed effects and bootstrap 95% confidence intervals estimated for the first analysis model. The left plot depicts the effects of the representation strategy (β_{0r} intercepts) and the right plot shows the effects of the number of sources (β_{1r} slopes).

able r^* instead of the nominal variable r used in the first model. In particular, r^* represents the size of the network, so we coded *SS-R* as 0, *MS-SR@FC* as 0.2, *MS-CR@6* as 0.4, *MS-CR@4* as 0.6, *MS-CR@2* as 0.8, and *MSS-CR* as 1 (see Fig. 2.5). In detail, this second model is as follows:

$$y_i^* = \beta_0 + \beta_{1d[i]} \cdot r_i^* + \beta_{2d[i]} \cdot n_i^* + \beta_{3d[i]} \cdot r_i^* \cdot n_i^* + e_i \qquad e_i \sim N(0, \sigma_e^2) \tag{2.13}$$

$$\beta_{1d} = \beta_{10} + u_{1d} \qquad u_{1d} \sim N(0, \sigma_1^2) \tag{2.14}$$

$$\beta_{2d} = \beta_{20} + u_{2d} \qquad u_{2d} \sim N(0, \sigma_2^2) \tag{2.15}$$

$$\beta_{3d} = \beta_{30} + u_{3d} \qquad u_{3d} \sim N(0, \sigma_3^2). \tag{2.16}$$

In contrast to the first model, there is no representation-specific fixed intercept but an overall intercept β_0. The effect of the network size is similarly modeled as the sum of an overall fixed slope β_{10} and a random dataset-specific effect u_{1d}. Likewise, this model includes the main effect of the number of sources (fixed effect β_{20}), as well as its interaction with the network size (fixed effect β_{30}). Fig. 2.12 shows the fitted coefficients, confirming the statistically positive effect of the size of the networks and, to a smaller degree but still significant, of the number of sources. The interaction term is not statistically significant, probably because of the unclear benefit of the number of sources in *MS-SR@FC*.

Overall, these analyses confirm that all multi-source strategies outperform the single-source representations, with a direct relation to the number of parameters in the network. In addition, there is a clearly positive effect of the number of sources, with a minor interaction between both factors.

Fig. 2.10 also suggests that the variability of performance scores decreases with the number of learning sources used. This implies that if there are more learning sources available, one can expect less variability across instantiations of the network. Most importantly, variability obtained for a single learning source (n = 1) is always larger than the variability with 2 or more sources. The Ballroom dataset shows much smaller variability when BPM is included in the combination. For this specific dataset, this indicates

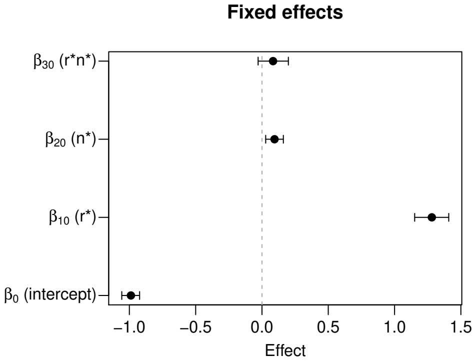


Figure 2.12: Fixed effects and bootstrap 95% confidence intervals estimated for the second analysis model, depicting the overall intercept (β_0), the slope of the network size (β_{10}), the slope of the number of sources (β_{20}), and their interaction (β_{30}).

that once *bpm* is used to learn the representation, the expected performance is stable and does not vary much, even if we keep including more sources. Section 2.5.3 provides more insight in this regard.

2.5.3. SINGLE-SOURCE VS. MULTI-SOURCE

The evidence so far tells us that, *on average*, learning from multiple sources leads to better performance than learning from a single source. However, it could be possible that the *SS-R* representation with the best learning source for the given target dataset still performs better than a multi-source alternative. In fact, in Fig. 2.10 there are many cases where the best *SS-R* representation (black circles at $n = 1$) already perform quite well compared to the more sophisticated alternatives. Fig. 2.13 presents similar scatter plots, but now explicitly differentiating between representations using the single best source (filled circles, solid lines) and not using it (empty circles, dashed lines). The results suggest that even if the strongest learning source for the specific dataset is not used, the others largely compensate for it in the multi-source representations, catching up and even surpassing the best *SS-R* representations. The exception to this rule is again *bpm* in the Ballroom dataset, where it definitely makes a difference. As the plots shows, the variability for low numbers of learning sources is larger when not using the strongest source, but as more sources are added, this variability reduces.

To further investigate this issue, for each target dataset, we also computed the variance component due to each of the learning sources, excluding *SS-R* representations [88]. A large variance due to one of the sources means that, on average and for that specific dataset, there is a large difference in performance between having that source or not. Table 2.6 shows all variance components, highlighting the per-dataset largest. Apart from *bpm* in the Ballroom dataset, there is no clear evidence that one single source is specially good in all datasets, which suggests that in general there is not a single source that one would use by default. Notably though, sources *artist*, *tag* and *self* tend to have large variance components.

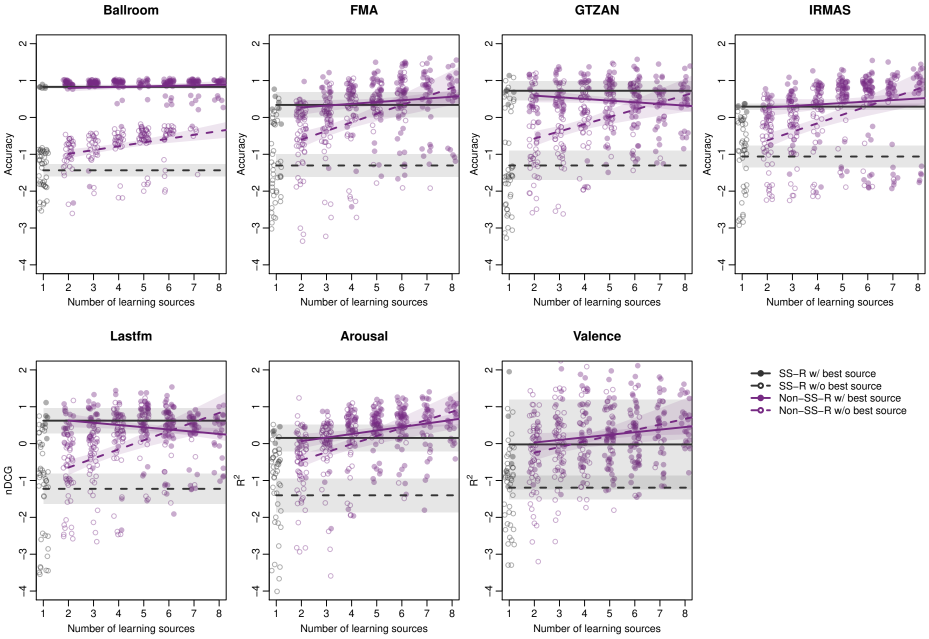


Figure 2.13: (Standardized) performance by number of learning sources. Solid points mark representations including the source performing best with *SS-R* in the dataset; empty points mark representations without it. Solid and dashed lines represent linear fits, respectively; dashed areas represent 95% confidence intervals.

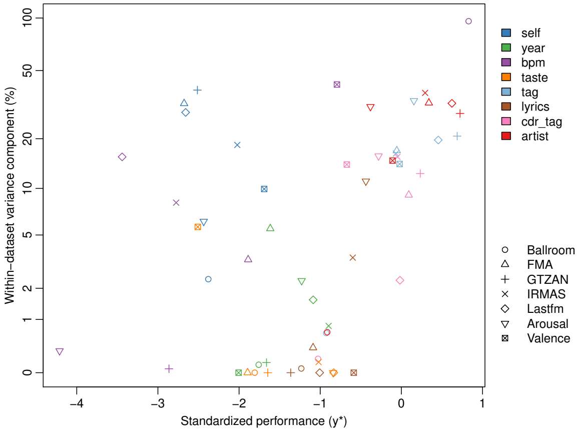


Figure 2.14: Correlation between (standardized) *SS-R* performance and variance component.

Table 2.6: Variance components (as percent of total) of the learning sources, within each of the target datasets, and for non-*SS-R* representations. Largest per dataset in bold face.

	Ballroom	FMA	GTZAN	IRMAS	Lastfm	Arousal	Valence
self	2	**32**	**39**	18	29	6	10
year	<1	6	<1	1	2	2	<1
bpm	**96**	3	<1	8	16	<1	**42**
taste	<1	<1	<1	<1	<1	<1	6
tag	1	17	21	16	20	**33**	14
lyrics	<1	<1	<1	3	<1	11	<1
cdr_tag	<1	9	12	16	2	16	14
artist	1	**32**	28	**37**	**32**	31	15

In addition, we observe that the sources with largest variance are not necessarily the sources that obtain the best results by themselves in an *SS-R* representation (see Fig. 2.8). We examined this relationship further by calculating the correlation between variance components and (standardized) performance of the corresponding *SS-R*s. The Pearson correlation is 0.38, meaning that there is a mild association. Fig. 2.14 further shows this with a scatterplot, with a clear distinction between poorly-performing sources (*year*, *taste* and *lyrics* at the bottom) and well-performing sources (*tag*, *cdr_tag* and *artist* at the right).

This result implies that even if some *SS-R* is particularly strong for a given dataset, when considering more complex fusion architectures, the presence of that one source is not necessarily required because the other sources make up for its absence. This is especially important in practical terms, because different tasks generally have different best sources, and practitioners rarely have sufficient domain knowledge to select them up front. Also, and unlike the Ballroom dataset, many real-world problems are not easily solved with a single feature. Therefore, choosing a more general representation based on multiple sources is a much simpler way to proceed, which still yields comparable or better results.

In other words, if “a single deep representation to rule them all” is pre-trained, it is advisable to base this representation on multiple learning sources. At the same time, given that *MSS-CR* representations also generally show strong performance (albeit that they will bring high dimensionality), and that they will come ‘for free’ as soon as *SS-R* networks are trained, alternatively, we could imagine an ecosystem in which the community could pre-train and release many *SS-R* networks for different individual sources in a distributed way, and practitioners can then collect these into *MSS-CR* representations, without the need for retraining.

2.5.4. COMPACTNESS

Under an MTDTL setup with branching (the *MS-CR* architectures), as more learning sources are used, not only the representation will grow larger, but so will the necessary deep network to learn it: see Fig. 2.15 for an overview of necessary model parameters for the different architectures. When using all the learning sources, *MS-CR@6*, which for a considerable part encompasses a shared network architecture and branches out relatively late, has an around 6.3 times larger network size compared to the network size needed for *SS-R*. In

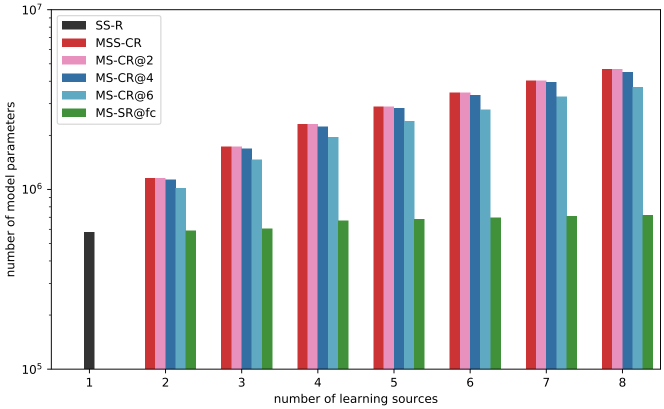


Figure 2.15: Number of network parameters by number of learning sources.

contrast, *MS-SR@FC*, which is the most heavily shared MTDTL case, uses a network that is only 1.2 times larger than the network needed for *SS-R*.

Also, while the representations resulting from the *MSS-CR* and various *MS-CR* architectures linearly depend on the chosen number of learning sources m (see Table 2.4), for *MS-SR@FC*, which has a fixed dimensionality of d independent of m, we do notice increasing performance as more learning sources are used, except *IRMAS* dataset. This implies that under MTDTL setups, the network does learn as much as possible from the multiple sources, even in case of fixed network capacity.

2.5.5. Multiple Explanatory Factors

By training representation models on multiple learning sources in the way we did, our hope is that the representation will reflect latent semantic facets that will ultimately allow for semantic explainability. In Fig. 2.16, we show a visualization that suggests this indeed may be possible. More specifically, we consider one of our *MS-CR* models trained on 5 learning sources. For each learning source-specific block of the representation, using the learning source-specific `fc-out` layers, we can predict a factor distribution z_t for each of the learning sources. Then, from the predicted z_t, one can either map this back on the original learning labels y_t, or simply consider the strongest predicted topics (which we visualized in Fig. 2.16), to relate the representation to human-understandable facets or descriptions.[12]

[11] The specific model used in the visualization is the 232th model from the experimental design we introduce in Section 2.4.3, which is performing better than 95% of other models on GTZAN target dataset.

[12] Note that, as soon as a pre-trained representation network model will be adapted to an new dataset through transfer learning, the `fc-out` layer cannot be used to obtain such explanations from the learning sources used in the representation learning, since the layers will then be fine-tuned to another dataset. However, we hypothesize it may be possible that the semantic explainability can still be preserved, if fine-tuning is jointly conducted with the original learning sources used during the pre-training time in the multi-objective strategy.

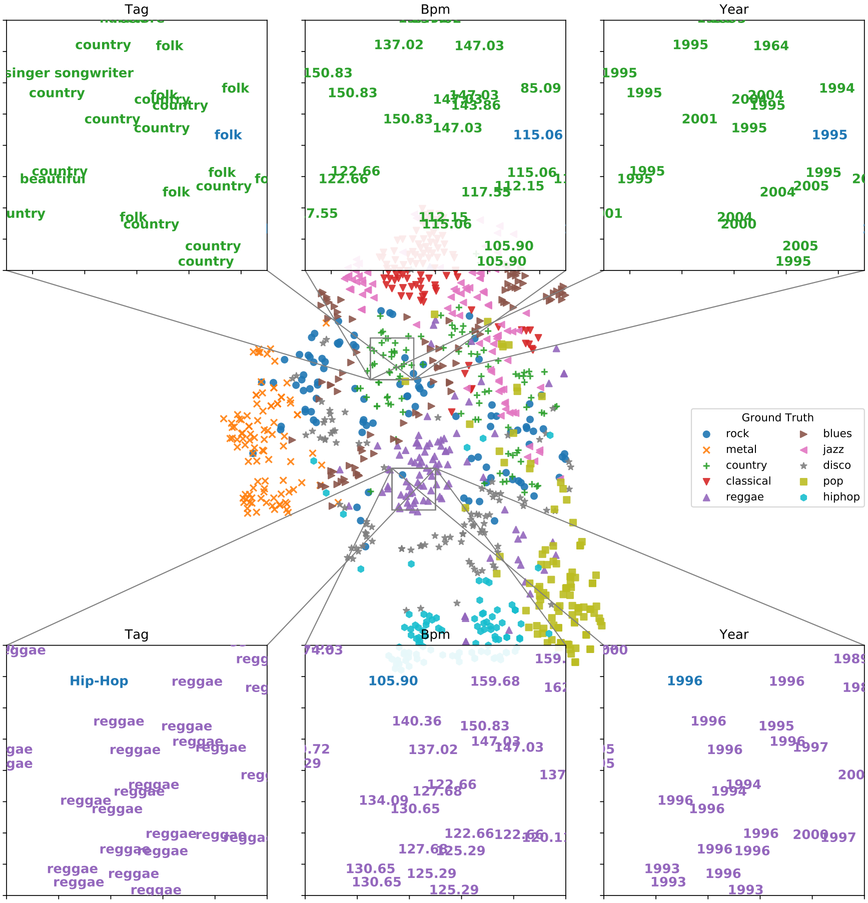


Figure 2.16: Potential semantic explainability of DTMTL music representations. Here, we provide a visualization using t-SNE [89], plotting 2-dimensional coordinates of each sample from the GTZAN dataset, as resulting from an *MS-CR* representation trained on 5 sources[11]. In the zoomed-in panes, we overlay the strongest topic model terms in z_t, for various types of learning sources.

2.6. Conclusion

In this paper, we have investigated the effect of different strategies to learn music representations with deep networks, considering multiple learning sources and different network architectures with varying degrees of shared information. Our main research questions are how the number and combination of learning sources (**RQ1**), and different configurations of the shared architecture (**RQ2**) affect effectiveness of the learned deep music representation. As a consequence, we conducted an experiment training 425 neural network models with different combinations of learning sources and architectures.

After an extensive empirical analysis, we can summarize our findings as follows:

- **RQ1** The number of learning sources positively affects the effectiveness of a learned deep music representation, although representations based on a single learning source will already be effective in specialized cases (e.g. BPM and the Ballroom dataset).

- **RQ2** In terms of architecture, the amount of shared information has a negative effect on performance: larger models with less shared information (e.g. *MS-CR@2*, *MSS-CR*) tend to outperform models where sharing is higher (e.g. *MS-CR@6*, *MS-SR@FC*), all of which outperform the base model (*SS-R*).

Our findings give various pointers to useful future work. First of all, 'generality' is difficult to define in the music domain, maybe more so than in CV or NLP, in which lower-level information atoms may be less multifaceted in nature (e.g. lower-level representations of visual objects naturally extend to many vision tasks, while an equivalent in music is harder to pinpoint). In case of clear task-specific data skews, practitioners should be pragmatic about this.

Also, we only investigated one special case of transfer learning, which might not be generalized well if one considers the adaptation of the pre-trained network for further fine-tuning with respect to their target dataset. Since there are various choices to make, which will bring substantial amount of variability, we decided to leave the aspects for further future works. We believe open-sourcing the models we trained throughout this work will be helpful for such follow-up works. Another limitation of current work is the selective set of label types in the learning sources. For instance, there are also a number of MIR related tasks that are using time-variant labels such as automatic music transcription, segmentation, beat tracking and chord estimation. We believe that such tasks should be investigated as well in the future to build a more complete overview of MTDTL problem.

Finally, in our current work, we still largely considered MTDTL as a 'black box' operation, trying to learn *how* MTDTL can be effective. However, the original reason for starting this work was not only to yield an effective general-purpose representation, but one that also would be semantically interpretable according to different semantic facets. We showed some early evidence our representation networks may be capable of picking up such facets; however, considerable future work will be needed into more in-depth analysis techniques of *what* the deep representations actually learned.

REFERENCES

[1] J. Kim, J. Urbano, C. C. S. Liem, and A. Hanjalic, *One deep music representation to rule them all? A comparative analysis of different representation learning strategies,* Neural Computing and Applications **32**, 1067 (2020).

[2] M. A. Casey, R. C. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, *Content-based music information retrieval: Current directions and future challenges,* Proceedings of the IEEE **96**, 668 (2008).

[3] R. Caruana, *Multitask learning,* Machine Learning **28**, 41 (1997).

[4] Y. Bengio, A. C. Courville, and P. Vincent, *Representation learning: A review and new perspectives,* IEEE Transactions on Pattern Analysis and Machine Intelligence **35**, 1798 (2013).

[5] W. Liu, T. Mei, Y. Zhang, C. Che, and J. Luo, *Multi-task deep visual-semantic embedding for video thumbnail selection,* in *IEEE Conference on Computer Vision and Pattern Recognition CVPR* (Boston, MA, USA, 2015) pp. 3707–3715.

[6] J. Bingel and A. Søgaard, *Identifying beneficial task relations for multi-task learning in deep neural networks,* in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2* (Association for Computational Linguistics, Valencia, Spain, 2017) pp. 164–169.

[7] S. Li, Z.-Q. Liu, and A. B. Chan, *Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network,* International Journal of Computer Vision **113**, 19 (2015).

[8] W. Zhang, R. Li, T. Zeng, Q. Sun, S. Kumar, J. Ye, and S. Ji, *Deep model based transfer and multi-task learning for biological image analysis,* in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD* (ACM, Sydney, NSW, Australia, 2015) pp. 1475–1484.

[9] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, *Facial landmark detection by deep multi-task learning,* in *Computer Vision - ECCV 13th European Conference, Proceedings, Part VI* (Springer, Zurich, Switzerland, 2014) pp. 94–108.

[10] L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit, *One model to learn them all,* CoRR **abs/1706.05137** (2017), arXiv:1706.05137 .

[11] J. R. Chang, C. Li, B. Póczos, and B. V. K. V. Kumar, *One network to solve them all - solving linear inverse problems using deep projection models,* in *IEEE International Conference on Computer Vision, ICCV* (IEEE Computer Society, Venice, Italy, 2017) pp. 5889–5898.

[12] J. Weston, S. Bengio, and P. Hamel, *Multi-tasking with joint semantic spaces for large-scale music annotation and retrieval,* Journal of New Music Research **40**, 337 (2011).

[13] Y. Aytar, C. Vondrick, and A. Torralba, *Soundnet: Learning sound representations from unlabeled video,* in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems* (Barcelona, Spain, 2016) pp. 892–900.

[14] P. Hamel and D. Eck, *Learning features from music audio with deep belief networks,* in *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR* (Utrecht, Netherlands, 2010) pp. 339–344.

[15] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, *Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription,* in *Proceedings of the 29th International Conference on Machine Learning, ICML* (Omnipress, Edinburgh, Scotland, UK, 2012).

[16] J. Schlüter and S. Böck, *Improved musical onset detection with convolutional neural networks,* in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP* (IEEE, Florence, Italy, 2014) pp. 6979–6983.

[17] K. Choi, G. Fazekas, and M. B. Sandler, *Automatic tagging using deep convolutional neural networks,* in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR* (New York City, USA, 2016) pp. 805–811.

[18] A. van den Oord, S. Dieleman, and B. Schrauwen, *Deep content-based music recommendation,* in *Advances in Neural Information Processing Systems 26 NIPS* (Lake Tahoe, NV, USA, 2013) pp. 2643–2651.

[19] P. Chandna, M. Miron, J. Janer, and E. Gómez, *Monoaural audio source separation using deep convolutional neural networks,* in *Latent Variable Analysis and Signal Separation - 13th International Conference, LVA/ICA, Proceedings* (Grenoble, France, 2017) pp. 258–266.

[20] I. Jeong and K. Lee, *Learning temporal features using a deep neural network and its application to music genre classification,* in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR* (New York City, USA, 2016) pp. 434–440.

[21] Y. Han, J. Kim, and K. Lee, *Deep convolutional neural networks for predominant instrument recognition in polyphonic music,* IEEE/ACM Transactions on Audio, Speech and Language Processing **25**, 208 (2017).

[22] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition,* in *3th International Conference on Learning Representations, ICLR* (San Diego, CA, USA, 2015).

[23] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (IEEE Computer Society, Las Vegas, NV, USA, 2016) pp. 770–778.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, *Going deeper with convolutions,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (IEEE Computer Society, Boston, MA, USA, 2015) pp. 1–9.

[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality,* in *Advances in Neural Information Processing Systems 26 NIPS* (Lake Tahoe, NV, USA, 2013) pp. 3111–3119.

[26] S. Dieleman, P. Brakel, and B. Schrauwen, *Audio-based music classification with a pretrained convolutional network,* in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR* (University of Miami, Miami, FL, USA, 2011) pp. 669–674.

[27] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, *Transfer learning for music classification and regression tasks,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR* (Suzhou, China, 2017) pp. 141–149.

[28] A. van den Oord, S. Dieleman, and B. Schrauwen, *Transfer learning by supervised pretraining for audio-based music classification,* in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR* (Taipei, Taiwan, 2014) pp. 29–34.

[29] D. Liang, M. Zhan, and D. P. W. Ellis, *Content-aware collaborative music recommendation using pre-trained neural networks,* in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR* (Málaga, Spain, 2015) pp. 295–301.

[30] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, *Cross-stitch networks for multi-task learning,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (IEEE Computer Society, Las Vegas, NV, USA, 2016) pp. 3994–4003.

[31] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, *The million song dataset,* in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR* (University of Miami, Miami, FL, USA, 2011) pp. 591–596.

[32] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, *Greedy layer-wise training of deep networks,* in *Advances in Neural Information Processing Systems 19, NIPS* (MIT Press, Vancouver, BC, Canada, 2006) pp. 153–160.

[33] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, *Extracting and composing robust features with denoising autoencoders,* in *Proceedings of the 25th International Conference on Machine Learning ICML* (ACM, Helsinki, Finland, 2008) pp. 1096–1103.

[34] P. Smolensky, *Information processing in dynamical systems: Foundations of harmony theory*, Tech. Rep. (University of Colorado, Boulder, Dept. of Computer Science, 1986).

[35] G. E. Hinton, S. Osindero, and Y.-W. Teh, *A fast learning algorithm for deep belief nets,* Neural Computation **18**, 1527 (2006), pMID: 16764513.

[36] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets,* in *Advances in Neural Information Processing Systems 27, NIPS* (Curran Associates, Inc., Montreal, QC, Canada, 2014) pp. 2672–2680.

[37] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, *Matchnet: Unifying feature and metric learning for patch-based matching,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (IEEE Computer Society, Boston, MA, USA, 2015) pp. 3279–3286.

[38] R. Arandjelovic and A. Zisserman, *Look, listen and learn,* in *IEEE International Conference on Computer Vision, ICCV* (IEEE Computer Society, Venice, Italy, 2017) pp. 609–617.

[39] Y. Huang, S. Chou, and Y. Yang, *Generating music medleys via playing music puzzle games,* in *Proceedings of the Thirty-Second Conference on Artificial Intelligence, AAAI* (AAAI Press, New Orleans, LA, USA, 2018) pp. 2281–2288.

[40] G. Salton and M. McGill, *Introduction to Modern Information Retrieval* (McGraw-Hill Book Company, 1984).

[41] P. Lamere, *Social tagging and music information retrieval,* Journal of New Music Research **37**, 101 (2008).

[42] P. Hamel, M. E. P. Davies, K. Yoshii, and M. Goto, *Transfer learning in MIR: Sharing learned latent representations for music audio classification and similarity,* in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR* (Curitiba, Brazil, 2013) pp. 9–14.

[43] E. Law, B. Settles, and T. M. Mitchell, *Learning to tag from open vocabulary labels,* in *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD, Proceedings, Part II* (Springer, Barcelona, Spain, 2010) pp. 211–226.

[44] T. Hofmann, *Probabilistic latent semantic analysis,* in *UAI: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann, Stockholm, Sweden, 1999) pp. 289–296.

[45] J. Schlüter, *Learning to pinpoint singing voice from weakly labeled examples,* in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR* (New York City, USA, 2016) pp. 44–50.

[46] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. W. Wilson, *CNN architectures for large-scale audio classification,* in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP* (IEEE, New Orleans, LA, USA, 2017) pp. 131–135.

[47] H. Lee, P. T. Pham, Y. Largman, and A. Y. Ng, *Unsupervised feature learning for audio classification using convolutional deep belief networks,* in *Advances in Neural Information Processing Systems 22, NIPS* (Curran Associates, Inc., Vancouver, BC, Canada, 2009) pp. 1096–1104.

[48] E. J. Humphrey and J. P. Bello, *Rethinking automatic chord recognition with convolutional neural networks,* in *11th International Conference on Machine Learning and Applications, ICMLA* (IEEE, Boca Raton, FL, USA, 2012) pp. 357–362.

[49] T. Nakashika, C. Garcia, and T. Takiguchi, *Local-feature-map integration using convolutional neural networks for music genre classification,* in *INTERSPEECH, 13th Annual Conference of the International Speech Communication Association* (ISCA, Portland, OR, USA, 2012) pp. 1752–1755.

[50] K. Ullrich, J. Schlüter, and T. Grill, *Boundary detection in music structure analysis using convolutional neural networks,* in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR* (Málaga, Spain, 2015) pp. 417–422.

[51] K. J. Piczak, *Environmental sound classification with convolutional neural networks,* in *25th IEEE International Workshop on Machine Learning for Signal Processing, MLSP* (IEEE, Boston, MA, USA, 2015) pp. 1–6.

[52] A. J. R. Simpson, G. Roma, and M. D. Plumbley, *Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network,* in *Latent Variable Analysis and Signal Separation - 12th International Conference, LVA/ICA, Proceedings* (Springer, Liberec, Czech Republic, 2015) pp. 429–436.

[53] H. Phan, L. Hertel, M. Maaß, and A. Mertins, *Robust audio event recognition with 1-max pooling convolutional neural networks,* in *INTERSPEECH 17th Annual Conference of the International Speech Communication Association* (ISCA, San Francisco, CA, USA, 2016) pp. 3653–3657.

[54] J. Pons, T. Lidy, and X. Serra, *Experimenting with musically motivated convolutional neural networks,* in *14th International Workshop on Content-Based Multimedia Indexing, CBMI* (IEEE, Bucharest, Romania, 2016) pp. 1–6.

[55] B. Stasiak and J. Monko, *Analysis of time-frequency representations for musical onset detection with convolutional neural network,* in *Proceedings of the Federated Conference on Computer Science and Information Systems, FedCSIS* (Gdańsk, Poland, 2016) pp. 147–152.

[56] H. Su, H. Zhang, X. Zhang, and G. Gao, *Convolutional neural network for robust pitch determination,* in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP* (IEEE, Shanghai, China, 2016) pp. 579–583.

[57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks,* Communications of the ACM **60**, 84 (2017).

[58] S. Dieleman and B. Schrauwen, *End-to-end learning for music audio,* in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP* (IEEE, Florence, Italy, 2014) pp. 6964–6968.

[59] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, *Wavenet: A generative model for raw*

audio, in *The 9th ISCA Speech Synthesis Workshop, SSW* (ISCA, Sunnyvale, CA, USA, 2016) p. 125.

[60] N. Jaitly and G. E. Hinton, *Learning a better representation of speech soundwaves using restricted boltzmann machines,* in *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP* (IEEE, Prague, Czech Republic, 2011) pp. 5884–5887.

[61] J. Lee, J. Park, K. L. Kim, and J. Nam, *Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms,* in *14th Sound and Music Computing Conference, SMC* (Espoo, Finland, 2017).

[62] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift,* in *Proceedings of the 32nd International Conference on Machine Learning, ICML* (JMLR, Inc., Lille, France, 2015) pp. 448–456.

[63] V. Nair and G. E. Hinton, *Rectified linear units improve restricted boltzmann machines,* in *Proceedings of the 27th International Conference on Machine Learning ICML* (Omnipress, Haifa, Israel, 2010) pp. 807–814.

[64] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: A simple way to prevent neural networks from overfitting,* Journal of Machine Learning Research **15**, 1929 (2014).

[65] J. Nam, J. Herrera, M. Slaney, and J. O. Smith, *Learning sparse feature representations for music annotation and retrieval,* in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR* (FEUP Edições, Porto, Portugal, 2012) pp. 565–570.

[66] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, *A comparison of audio signal preprocessing methods for deep neural networks on music tagging,* in *26th European Signal Processing Conference, EUSIPCO* (IEEE, Roma, Italy, 2018) pp. 1870–1874.

[67] M. Dörfler, T. Grill, R. Bammer, and A. Flexer, *Basic filters for convolutional neural networks applied to music: Training or design?* Neural Computing and Applications (2018), 10.1007/s00521-018-3704-x.

[68] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization,* in *3th International Conference on Learning Representations, ICLR* (San Diego, CA, USA, 2015).

[69] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, *Automatic differentiation in PyTorch,* in *NIPS-W* (2017).

[70] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python,* Journal of Machine Learning Research **12**, 2825 (2011).

[71] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, *librosa: Audio and music signal analysis in python,* in *Proceedings of the 14th Python in Science Conference SciPy*, edited by K. Huff and J. Bergstra (Austin, TX, USA, 2015) pp. 18 – 24.

[72] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, *FMA: A dataset for music analysis,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR* (Suzhou, China, 2017) pp. 316–323.

[73] G. Tzanetakis and P. R. Cook, *Musical genre classification of audio signals,* IEEE Transactions on Speech and Audio Processing **10**, 293 (2002).

[74] C. Kereliuk, B. L. Sturm, and J. Larsen, *Deep learning and music adversaries,* IEEE Transactions on Multimedia **17**, 2059 (2015).

[75] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, *An experimental comparison of audio tempo induction algorithms,* IEEE Transactions on Audio, Speech, and Language Processing **14**, 1832 (2006).

[76] U. Marchand and G. Peeters, *Scale and shift invariant time/frequency representation using auditory statistics: Application to rhythm description,* in *26th IEEE International Workshop on Machine Learning for Signal Processing, MLSP* (IEEE, Salerno, Italy, 2016) pp. 1–6.

[77] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, *A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals,* in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR* (FEUP Edições, Porto, Portugal, 2012) pp. 559–564.

[78] M. Soleymani, M. N. Caro, E. M. Schmidt, C. Sha, and Y. Yang, *1000 songs for emotional analysis of music,* in *Proceedings of the 2nd ACM International Workshop on Crowdsourcing for Multimedia CrowdMM@ACM Multimedia* (ACM, Barcelona, Spain, 2013) pp. 1–6.

[79] Ò. Celma, *Music Recommendation and Discovery - The Long Tail, Long Fail, and Long Play in the Digital Music Space* (Springer, 2010).

[80] B. L. Sturm, *The state of the art ten years after a state of the art: Future research in music information retrieval,* Journal of New Music Research **43**, 147 (2014).

[81] B. L. Sturm, *The "Horse" inside: Seeking causes behind the behaviors of music content analysis systems,* Computers in Entertainment **14**, 3:1 (2016).

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

[83] D. C. Montgomery, *Design and Analysis of Experiments,* 8th ed. (Wiley, 2012).

[84] P. Goos and B. Jones, *Optimal Design of Experiments: A Case Study Approach*, 1st ed. (Wiley, 2011).

[85] G. E. Hinton, *Connectionist learning procedures,* Artificial Intelligence **40**, 185 (1989).

[86] Y. Hu, Y. Koren, and C. Volinsky, *Collaborative filtering for implicit feedback datasets,* in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM)* (IEEE Computer Society, Pisa, Italy, 2008) pp. 263–272.

[87] A. Gelman and J. Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models* (Press, Cambridge University, 2006).

[88] S. R. Searle, G. Casella, and C. E. McCulloch, *Variance components* (Wiley, 2006).

[89] L. v. d. Maaten and G. Hinton, *Visualizing data using t-SNE,* Journal of machine learning research **9**, 2579 (2008).

3

Transfer learning of artist group factors to musical genre classification

The automated recognition of music genres from audio information is a challenging problem, as genre labels are subjective and noisy. Artist labels are less subjective and less noisy, while certain artists may relate more strongly to certain genres. At the same time, at prediction time, it is not guaranteed that artist labels are available for a given audio segment. Therefore, in this work, we propose to apply the transfer learning framework, learning artist-related information which will be used at inference time for genre classification. We consider different types of artist-related information, expressed through artist group factors, which will allow for more efficient learning and stronger robustness to potential label noise. Furthermore, we investigate how to achieve the highest validation accuracy on the given FMA dataset, by experimenting with various kinds of transfer methods, including single-task transfer, multi-task transfer and finally multi-task learning.

3.1. Introduction

Learning to Recognize Musical Genre from Audio is a challenge track of *The Web Conference 2018*. The main goal of the challenge is to predict musical genres of unknown audio segments correctly, by utilizing the FMA dataset [2] as a training set. The challenge therefore focuses on a classification task.

In machine learning, many classification tasks, such as visual object recognition, consider objective and clearly separable classes. In contrast, music genres consider subjective, human-attributed labels. These may be inter-correlated (e.g. a *rock* song may also be considered *pop*, many *classical* works are also *instrumental*) and dependent of a user's context (e.g., a *French rock* song is not *International* to a French listener). Generally, no

This chapter was published in Companion Proceedings of the The Web Conference 2018 [1].

universal genre taxonomy exists, and even the definition of 'genre' itself is problematic: what is usually understood as 'genre' in Music Information Retrieval would rather be characterized as 'style' in Musicology [3]. This makes genre classification a challenging problem. In our work, considering the given labels in the challenge, we consider a musical genre to be a category that consists of songs sharing certain aspects of musical characteristics.

Commonly, music tracks are released with explicit mentioning of titles and artists. The identity of the artist does not suffer from semantic taxonomy problems, and can thus be considered as a more objective label than the genre label. At the same time, songs from the same artist tend to share prominent musical characteristics. Considering that an artist is commonly mapped into one or multiple specific genres, but not the whole universe of possible genres, and that the other way around, sets of artists can be seen as exemplars for certain music genres, the musical characteristics that identify an artist may also be key features of certain musical genres.

Therefore, it will be beneficial to exploit artist-related information in a genre classification task. At the same time, learning a direct mapping from artist identity to genre label would not be practical. First of all, for an unknown audio segment for which a genre classification should be performed, the artist label may also not be available. Secondly, artist labels may not always be informative to a system, especially when an artist is newly introduced, so no previous history on the artist exists. Finally, an artist may have been active in multiple genres at once, but not be equally representative for all these genres. Given such constraints, we wish to employ a learning framework which only requires artist labels at training time, but not at prediction time, and that will allow for the inclusion of newly introduced artists, for whom not much extra information is available beyond their songs.

In this work, we therefore present a multi-task transfer framework for using artist labels to improve a genre classification model. Assuming that artist labels are given for each track in the training set, these labels are used as side information, allowing a model to learn the mapping between audio and artists, while capturing patterns that might as well be useful for genre prediction.

It has been shown that music representations learned from raw artist labels can effectively transfer to other music-related tasks [4]. However, learning more than thousands of artists as individual classes is not efficient for at least two reasons:

- Due to data sparsity, only a few tracks are assigned per class;
- Despite the uniqueness of each artist, it can be beneficial to group them into clusters of similar artists, avoiding learning bottlenecks caused by large numbers of classes.

To overcome these potential problems, we therefore apply a label pre-processing step, obtaining Artist Group Factors (AGF) as learning targets, rather than individual artist identities.

Finally, we train Deep Convolutional Neural Networks (DCNNs) employing different learning setups, ranging from targeting genre and various types of AGFs with individual networks, to employing a shared architecture as introduced in multiple previous Multi-Task Learning (MTL) works [5–12].

In the remainder of this paper, we first discuss an initial data exploration leading to our choice for AGFs (Section 3.2). Subsequently, we will give a detailed description of the proposed approach (Section 3.3), followed by a discussion of experimental settings (Section 3.4). Finally, we will present our results (Section 3.5), followed by a short discussion and conclusion (Section 3.6).

3.2. Initial data exploration

In the beginning of the challenge, we first explored the training data, and investigated a conventional data-driven approach using a DCNN for music genre classification, with genre labels as targets.

First of all, we had some concerns about the reliability of the genre annotations. As they were provided by users who uploaded the content, the users did not have access to a single genre taxonomy and unified annotation strategy. Thus, user-contributed annotations are expected to show more variability than annotations by experts. Furthermore, the dataset included 25,000 tracks from 5,152 unique albums. For 5,028 out of these 5,152 albums, genre annotations were made at the album level. While all tracks in an album can belong to a single genre, this is not always true. Indeed, we could discover examples of the case in which different tracks on the same album would belong to different genres, as well as multiple misannotations. Given these reliability issues, it is not guaranteed that by targeting these annotations only, generalized model performance for genre classification can be achieved.

To this end, while we will consider performance for direct (main top-)genre labels as targets (which we will denote as learning task category g in the remainder of this paper), in order to obtain more generalizable results obtained on more objective and consistent labeling data, we propose a multi-task transfer framework, introducing an Artist Group (AG) prediction task targeting AGFs.

3.3. Methodology

3.3.1. Artist Group Factors

The main idea of extracting AGFs is to cluster artists based on meaningful feature sets that allow for aggregation at (and beyond) the artist level. For instance, one can collect genre labels from songs belonging to each artist, and then construct a Bag-of-Word (BoW) artist-level feature vector. Each dimension of the vector represents a genre, with the magnitude of the vector indicating genre frequency among a song collection. Alternatively, a BoW feature vector can be constructed by counting latent 'terms' belonging to each artist, which can be obtained by a dictionary learned from song-level or frame-level features through K-means clustering [13] or the Sparse Coding [14] method.

Once artist-level BoW feature vectors are constructed, standard clustering methods such as K-Means, or more sophisticated topic modeling algorithms such as Latent Dirichlet Allocation (LDA) [15] can be applied to find a small number of latent groups of artists: the AGFs for this particular feature set. This 2-step cascading pipeline is illustrated in Figure 3.1.

In this work, we exploit four feature sets, which reflect different levels of musical and acoustical aspects of songs. From these feature sets, we obtain artist-level BoW vectors. Subsequently, LDA is applied to transform artist-level BoW vectors into dedicated AGF representations for the particular feature set. We will both consider these artist group

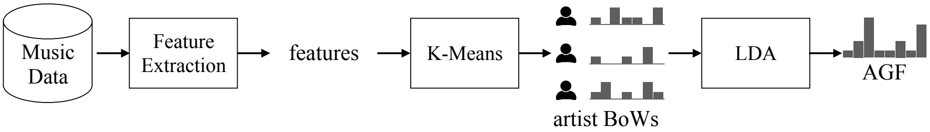


Figure 3.1: Artist group factor extraction pipeline.

prediction tasks and the main genre classification task within our learning framework: an overview summary is given in Table 3.1.

MFCCs

Mel-Frequency Cepstral Coefficients (MFCCs), which are known to be efficient low-level descriptors for timbre analysis, were used as features of the artist grouping. The coefficients are initially calculated for short-time audio frames. Considering the coefficients over all audio frames of tracks for all artists, we build an universal dictionary of features using K-Means clustering. AGFs resulting from this feature set will belong to learning task category `m`.

dMFCCs

Along with MFCCs, we also use time-deltas of MFCCs (first-order differences between subsequent frames), to consider the temporal dynamics of the timbre for the artist grouping. AGFs resulting from this feature set will be denoted by `d`.

Essentia

We use song-level feature vectors from Essentia [16], which is a music feature extraction library. It extracts descriptors ranging from low-level features, such as statistics of spectral characteristics, to high-level features, including danceability [17] or semantic features learned from the data. After filtering descriptor entries that include missing values or errors, we obtained a 4374-dimensional feature vector per track. Before training a dictionary, we apply quantile normalization: a rank-based normalization process that transforms the distribution of the given features to follow a target distribution [18], which we set to be a normal distribution in this case. AGFs resulting from this feature set will belong to learning task category `e`.

Subgenres

We also use the 150 genre labels, including sub-genres, as a pre-defined dictionary for semantic description. For these, we directly build artist-level BoW vectors by aggregating all the genre labels from tracks by an artist. AGFs resulting from this feature set will belong to learning task category `s`.

3.3.2. Network Architectures

The architecture of the proposed system can be divided into two parts, as shown in Figure 3.2. We first train multiple DCNNs, targeting the various categories of learning targets (genres or various AGFs). Subsequently, transfer takes place: a multilayer perceptron (MLP) for the final genre classification is trained, utilizing features that were derived from the previously trained DCNNs.

Table 3.1: Details of Learning Targets

id	Category	Source	Dictionary	Dimension
g	Main	Genre	N / A	16
m	AGF	MFCC	K-means	25
d		dMFCC		25
e		Essentia [16]		4374
s		Subgenre	N / A	150

Table 3.2: Network Architectures for Encoder f

Layers	Output shape
Input layer	128 × 43 × 1
Conv 5 × 5, ELU	128 × 43 × 16
MaxPooling 2 × 1	64 × 43 × 16
Conv 3 × 3, BN, ELU	64 × 43 × 32
MaxPooling 2 × 2	32 × 21 × 32
Dropout (0.1)	32 × 21 × 32
Conv 3 × 3, ELU	32 × 21 × 64
MaxPooling 2 × 2	16 × 10 × 64
Conv 3 × 3, BN, ELU	16 × 10 × 64
MaxPooling 2 × 2	8 × 5 × 64
Dropout (0.1)	8 × 5 × 64
Conv 3 × 3, ELU	8 × 5 × 128
MaxPooling 2 × 2	4 × 2 × 128
Conv 3 × 3, ELU	4 × 2 × 256
Conv 1 × 1, BN, ELU	4 × 2 × 256
GlobalAveragePooling, BN	256
Dense, BN, ELU	256
Dropout (0.5)	256
Output layer 16 or 40	16 or 40

DCNN

We adapted DCNN models to obtain transferable features for genre classification (Table 3.2). The input size of the input layer is 128×43, which is the size of a spectrogram with 128 mel bins and 43 samples (1 second of audio). After the input layer, there are seven convolutional layers followed by a max-pooling layer, except for the last two layers. The first convolutional layer has 5 × 5 kernels and the last convolutional layer has 1×1 kernels. Except for those two layers, all convolutional layers have 3×3 kernels. Outputs of the last convolutional layer are subsampled by global-average-pooling. Finally, they are connected to two dense layers for predicting AGF clusters or genres. Batch normalization [19] and dropouts [20] are sparsely used to prevent overfitting. Exponential Linear Unit (ELU) [21] is used as an activation function for the convolutional layers and Softmax is used for the output layer.

Shared Architecture

Considering that lower layers of DCNNs usually capture lower-level features such as edges from images or spectrograms, we hypothesized that sharing lower layers among the various DCNNs can be effective under the scenario where multiple learning sources are available. With this approach, one can expect that it not only ensures sufficient specialization on task-specific upper layers, but also benefits from regularization effects on lower layers[12]. Joint learning of multiple tasks with shared layers can prevent the shared layer to overfit for a specific task, instead learning underlying factors that have commonalities required across tasks [5, 22].

Throughout the experiment, we used the shared architecture that shares only the first convolutional block. It consists of the first convolutional and the max-pooling layer. For brevity, for the remainder of the paper, we use Single-Task Nets (STNs) and an Multi-Task Net (MTN) to refer to the non-shared networks and shared networks respectively.

Transfer method

The proposed system learns and predicts a genre of an input spectrogram by transferring pre-trained features from Section 3.3.2. We trained an MLP with a single hidden layer; the size of the hidden layer was 1024. ELU non-linearity was used for the hidden layer and Softmax was used for the output layer. Dropouts of 50% were applied for the input layer and a hidden layer.

Note that for both the feature learning phase and the transfer learning phase, we keep using a segment-wise learning approach. Only at the final inference step, we aggregate all the segment-level predictions, by taking the average of each segment's predicted probability for the genres.

Training

At training time, we iteratively update the model parameters with the mini-batch stochastic gradient descent method using the Adam algorithm [23]. For data augmentation, we randomly crop 1-second excerpts from the entire track included in the mini-batch. We use 64 samples per batch and set the learning rate to 0.001 across the experiments.

For comparison between methods, experiments are run with a fixed number of epochs. We set 1000 epochs for an MTN and 200 for STNs. Since we took a similar stochastic

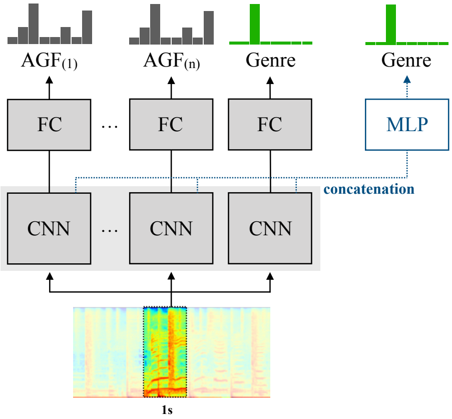


Figure 3.2: Illustration for the transfer learning scenario. Dotted lines indicate the setup for the multilayer perceptron for performing final genre classification.

update algorithm to [7] for the shared architecture, for the number of updates for task-specific layers in a shared network, the number of epochs used for training non-shared networks should be multiplied with the number of involved learning tasks. For the transfer learning phase, we also set the number of epochs to train the MLP to 50.

3.3.3. PRE-PROCESSING

We use mel spectrograms as the input representation for the neural networks. We extract 128-dimensional mel spectra for audio frames of 46ms, with 50% overlap with adjacent frames. To enhance lower-intensity levels of input mel spectrograms at higher frequencies, we take dB-scale log amplitudes of each mel spectrum.

3.3.4. IMPLEMENTATION DETAILS

The experiments were run on GPU-accelerated hardware and software environments. We used Lasagne [24], Theano [25] and Keras [26] as main experimental frameworks[1]. We used a number of different GPUs, including NVIDIA GRID-K2, NVIDIA GTX 1070, NVIDIA TITAN X.

3.4. EXPERIMENTS

To investigate the effectiveness of various types of AGFs for transfer learning, we trained all 31 possible combinations of given learning tasks, including AGFs (`m`, `d`, `e`, `s`) and main top-genre labels (`g`). For each run, to investigate the optimal feature architecture, we tested both shared networks and separate networks for each learning task. This leads to a total number of 62 cases, including all the combinations of learning tasks per network architecture.

[1] The main code for the experiment can be found in `https://github.com/eldrin/Lasagne-MultiTaskLearning`

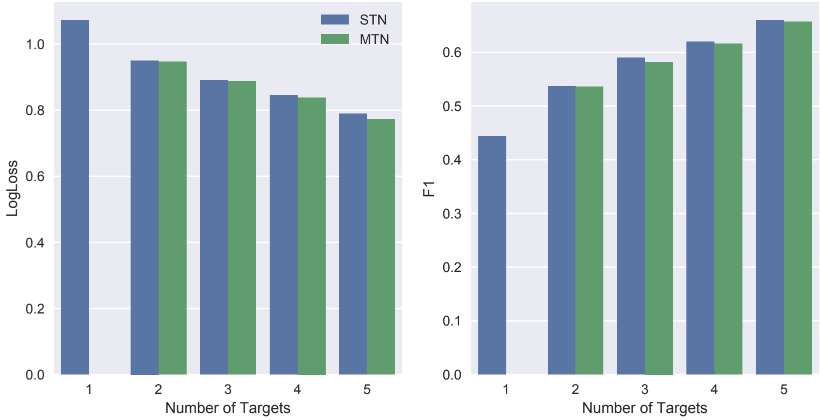


Figure 3.3: Average performance for the number of tasks involved in feature learning

Table 3.3: Comparison of the average performance with or without the main task

	LogLoss		F1	
	STN	MTN	STN	MTN
without g	1.0079	0.9618	0.4932	0.5168
with g	**0.8540**	**0.8486**	**0.6154**	**0.6155**

However, in all cases in which multiple tasks are considered, the networks have a larger number of parameters compared to the case in which a network focuses on a single task. With a subsequent experiment, we therefore tried to verify the effect of more parameters and larger networks vs. the effect of using more tasks. To this end, we train wide Single Task Networks (wSTNs), targeting only genre, but having an equal number of parameters to the MTNs/STNs targeting multiple tasks. Finally, with respect to the number of tasks involved, we compare the best performance of MTNs/STNs to the performance of wSTNs with the same number of parameters.

As for the AGFs using song-level or frame-level features, we trained K-means algorithms employing 2048 clusters. We observed that lower numbers of clusters (e.g. 1024) can cause artists with few tracks to get a zero vector as artist-level BoW representation, due to data sparsity. Throughout the experiments, we used a fixed number of latent artist groups, set to 40.

Finally, for the internal evaluation, we divided the given training dataset employing a stratified random 85/15 split.

3.5. RESULTS

3.5.1. MULTIPLE LEARNING TASKS IN STN VS. MTN

Table 3.4: The performance of various combinations of AGFs and the top-level main genre target as a feature learning task.

	STN		MTN	
	LogLoss	F1	LogLoss	F1
g	0.8891	0.5963		
m	1.1812	0.3581		
d	1.0987	0.3967	N/A	N/A
e	1.2542	0.3437		
s	0.9404	0.5218		
gs	0.8606	0.6114	0.8578	0.6190
ge	0.8811	0.5953	0.8792	0.5996
gd	0.8845	0.5898	0.8803	0.5955
gm	0.8874	0.5957	0.8813	0.6037
se	0.9124	0.5537	0.9079	0.5502
sd	0.9191	0.5601	0.9146	0.5412
sm	0.9260	0.5581	0.9283	0.5458
ed	1.0557	0.4433	1.0422	0.4399
em	1.1186	0.4244	1.1060	0.4376
dm	1.0583	0.4373	1.0704	0.4280
gse	0.8361	0.6255	0.8335	0.6277
gsd	0.8579	0.6280	0.8519	0.6150
gsm	0.8486	0.6289	0.8541	0.6153
ged	0.8528	0.6051	0.8601	0.6067
gem	0.8645	0.5988	0.8701	0.6056
gdm	0.8773	0.5985	0.8845	0.5941
sed	0.8965	0.5818	0.8867	0.5640
sem	0.9104	0.5834	0.8889	0.5668
sdm	0.9211	0.5629	0.9109	0.5572
edm	1.0359	0.4879	1.0365	0.4675
gsed	0.8211	0.6343	0.8132	0.6328
gsem	0.8264	0.6352	0.8172	0.6284
gsdm	0.8407	0.6379	0.8288	0.6170
gedm	0.8466	0.6053	0.8450	0.6152
sedm	0.8906	0.5856	0.8875	0.5870
gsedm	**0.7894**	**0.6599**	**0.7727**	**0.6571**

In general, we observe that the number of learning tasks has a positive effect on both performance metrics. As shown in Table 3.3, it also is found that cases in which the main top-genre classification are included yield better results in comparison to other combinations of tasks.

Considering STN vs. MTN, on the log loss metric, MTN shows better results, but in

Table 3.5: Comparison between wSTN (single genre classification task) and STN/MTN setups (multiple tasks) learning setups. The reported performances of STN and MTN consider the task combinations for which the best performance was obtained, given the mentioned number *N* of tasks.

	LogLoss			F1		
N	wSTN	STN	MTN	wSTN	STN	MTN
2	0.8688	0.8606	0.8578	0.6071	0.6114	0.6190
3	0.8546	0.8361	0.8335	**0.6629**	0.6289	0.6277
4	0.8278	0.8211	0.8132	0.6451	0.6352	0.6328
5	0.8290	0.7893	**0.7727**	0.6528	0.6599	0.6571

the case of the f1-measure, the opposite is shown. Generally, considering the number of learning tasks and absolute magnitude of differences, the difference observed between the two methods cannot be deemed significant; more experiments with additional datasets and multiple splits would be needed to assess whether statistically significant differences between STN vs. MTN approaches can be obtained.

For both STN and MTN, the best performance we achieved uses all the learning tasks, as shown in the last row of Table 3.4.

3.5.2. Networks for Multiple Learning Tasks vs. Large Network on a Single Task

We also compared the performance between the best STNs and MTNs for a given number of learning tasks, versus the performance of a wSTN that has equal model capability to these multi-task setups in terms of parameters and architecture, but only is trained on direct main top-genre classification. The corresponding results are shown in Table 3.5. It can be seen that MTN representations yield better performance on the log loss metric when all 5 learning tasks (all AGFs and the main top-genre) are used, although at the same time, wSTN performs better when considering the f1-measure for the case in which 2 learning tasks are used. In other cases, differences between the setups appear marginal; further experiments would be needed to assess whether STNs/MTNs will give significant performance boosts in case a larger set of tasks would be considered.

3.6. Discussion & Conclusion

In this work, we proposed including several categories of low-rank AGFs, expressing artist-level information, into the task of classifying music genre based on musical audio. Our experimental results support the hypothesis that by targeting different categories of AGFs, deep networks can learn features from musical audio that can meaningfully support genre classification. The inclusion of multiple parallel learning tasks considering different AGF categories, and the inclusion of both genre- and AGF-based tasks in a multi-task setup, also both seem beneficial, although further work will need to be done to assess whether observed effects are truly significant. For this, other datasets will have to be included for training and testing; furthermore, alternative cluster algorithms and clustering parameters should be investigated to achieve the most robust AGF-based features.

REFERENCES

[1] J. Kim, M. Won, X. Serra, and C. C. S. Liem, *Transfer learning of artist group factors to musical genre classification,* in *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, edited by P. Champin, F. L. Gandon, M. Lalmas, and P. G. Ipeirotis (ACM, 2018) pp. 1929–1934.

[2] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, *FMA: A dataset for music analysis,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017,* edited by S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull (2017) pp. 316–323.

[3] C. C. S. Liem, A. Rauber, T. Lidy, R. Lewis, C. Raphael, J. D. Reiss, T. Crawford, and A. Hanjalic, *Music information technology and professional stakeholder audiences: Mind the adoption gap,* in *Multimodal Music Processing*, Dagstuhl Follow-Ups, Vol. 3, edited by M. Müller, M. Goto, and M. Schedl (Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2012) pp. 227–246.

[4] J. Park, J. Lee, J. Park, J. Ha, and J. Nam, *Representation learning of music using artist labels,* in *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, edited by E. Gómez, X. Hu, E. Humphrey, and E. Benetos (2018) pp. 717–724.

[5] R. Caruana, *Multitask learning,* in *Learning to Learn*, edited by S. Thrun and L. Y. Pratt (Springer, 1998) pp. 95–133.

[6] Y. Bengio, A. C. Courville, and P. Vincent, *Representation learning: A review and new perspectives,* IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1798 (2013).

[7] W. Liu, T. Mei, Y. Zhang, C. Che, and J. Luo, *Multi-task deep visual-semantic embedding for video thumbnail selection,* in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (IEEE Computer Society, 2015) pp. 3707–3715.

[8] J. Bingel and A. Søgaard, *Identifying beneficial task relations for multi-task learning in deep neural networks,* in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, edited by M. Lapata, P. Blunsom, and A. Koller (Association for Computational Linguistics, 2017) pp. 164–169.

[9] S. Li, Z. Liu, and A. B. Chan, *Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network,* Int. J. Comput. Vis. **113**, 19 (2015).

[10] W. Zhang, R. Li, T. Zeng, Q. Sun, S. Kumar, J. Ye, and S. Ji, *Deep model based transfer and multi-task learning for biological image analysis,* IEEE Trans. Big Data **6**, 322 (2020).

[11] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, *Facial landmark detection by deep multi-task learning,* in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI*, Lecture Notes in Computer

Science, Vol. 8694, edited by D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars (Springer, 2014) pp. 94–108.

[12] J. Kim, J. Urbano, C. C. S. Liem, and A. Hanjalic, *One deep music representation to rule them all? A comparative analysis of different representation learning strategies,* Neural Comput. Appl. **32**, 1067 (2020).

[13] S. P. Lloyd, *Least squares quantization in PCM,* IEEE Trans. Inf. Theory **28**, 129 (1982).

[14] A. Coates and A. Y. Ng, *The importance of encoding versus training with sparse coding and vector quantization,* in *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, edited by L. Getoor and T. Scheffer (Omnipress, 2011) pp. 921–928.

[15] D. M. Blei, A. Y. Ng, and M. I. Jordan, *Latent dirichlet allocation,* J. Mach. Learn. Res. **3**, 993 (2003).

[16] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. R. Zapata, and X. Serra, *Essentia: An audio analysis library for music information retrieval,* in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, edited by A. de Souza Britto Jr., F. Gouyon, and S. Dixon (2013) pp. 493–498.

[17] P. Herrera and S. Streich, *Detrended fluctuation analysis of music signals: Danceability estimation and further semantic characterization,* in *Audio Engineering Society Convention 118* (Audio Engineering Society, 2005).

[18] D. Amaratunga and J. Cabrera, *Analysis of data from viral dna microchips,* Journal of the American Statistical Association **96**, 1161 (2001).

[19] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift,* in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, JMLR Workshop and Conference Proceedings, Vol. 37, edited by F. R. Bach and D. M. Blei (JMLR.org, 2015) pp. 448–456.

[20] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting,* J. Mach. Learn. Res. **15**, 1929 (2014).

[21] D. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (ELUs),* in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2016).

[22] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y. Wang, *Representation learning using multi-task deep neural networks for semantic classification and information retrieval,* in *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, edited by R. Mihalcea, J. Y. Chai, and A. Sarkar (The Association for Computational Linguistics, 2015) pp. 912–921.

[23] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2015).

[24] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, D. Maturana, M. Thoma, E. Battenberg, J. Kelly, J. D. Fauw, M. Heilman, diogo149, B. McFee, H. Weideman, takacsg84, peterderivaz, Jon, instagibbs, D. K. Rasul, CongLiu, Britefury, and J. Degrave, *Lasagne: First release.* (2015).

[25] Theano Development Team, *Theano: A Python framework for fast computation of mathematical expressions,* arXiv e-prints **abs/1605.02688** (2016).

[26] F. Chollet, *keras,* `https://github.com/fchollet/keras` (2015).

4

ARE NEARBY NEIGHBORS RELATIVES?: TESTING DEEP MUSIC EMBEDDINGS

Deep neural networks have frequently been used to directly learn representations useful for a given task from raw input data. In terms of overall performance metrics, machine learning solutions employing deep representations frequently have been reported to greatly outperform those using hand-crafted feature representations. At the same time, they may pick up on aspects that are predominant in the data, yet not actually meaningful or interpretable. In this paper, we therefore propose a systematic way to test the trustworthiness of deep music representations, considering musical semantics. The underlying assumption is that in case a deep representation is to be trusted, distance consistency between known related points should be maintained both in the input audio space and corresponding latent deep space. We generate known related points through semantically meaningful transformations, both considering imperceptible and graver transformations. Then, we examine within- and between-space distance consistencies, both considering audio space and latent embedded space, the latter either being a result of a conventional feature extractor or a deep encoder. We illustrate how our method, as a complement to task-specific performance, provides interpretable insight into what a network may have captured from training data signals.

4.1. INTRODUCTION

Music audio is a complex signal. Frequencies in the signal usually belong to multiple pitches, which are organized harmonically and rhythmically, and often originate from multiple acoustic sources in the presence of noise. When solving tasks in the Music Information Retrieval (MIR) field, within this noisy signal, the optimal subset of information needs to be found that leads to quantifiable and musical descriptors. Commonly,

This chapter was published in Frontiers in Applied Mathematics and Statistics **5**, 53 [1].

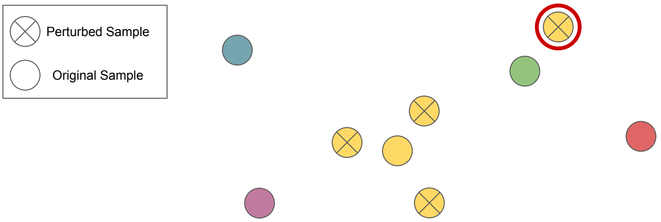


Figure 4.1: Simplified example illustrating distance assumption within a space. Circles without a cross indicate music clips. Yellow circles with crosses refer to hardly perceptible transformations of the yellow original clip. The top-right transformation, marked with a red outer circle, actually is closer to another original clip (green) than to its own original (yellow), which violates the assumption it should be closest to its original, and hence may be seen as an error-inducing transformation under a nearest-neighbor scheme.

this process is handled by pipelines exploiting a wide range of signal processing and machine learning algorithms. Beyond the use of *hand-crafted music representations*, which are informed by human domain knowledge, as an alternative, *deep music representations* have emerged, that are trained by employing deep neural networks (DNNs) and massive amounts of training data observations. Such deep representations are usually reported to outperform hand-crafted representations (e.g. [2–5]).

At the same time, the *performance* of MIR systems may be vulnerable to subtle input manipulation. The addition of small noise may lead to unexpected random behavior, regardless of whether traditional or deep models are used [6–9]. In a similar line of thought, in the broader deep learning (DL) community, increasing attention is given to adversarial examples that are barely differentiable from original samples, but greatly impact a network's performance [9, 10].

So far, the sensitivity of representations with respect to subtle input changes has mostly been tested in relation to dedicated machine learning tasks (e.g. object recognition, music genre classification), and examined by investigating whether these input changes cause performance drops. When purely considering the questions *whether relevant input signal information can automatically be encoded into a representation, and to what extent the representation can be deemed 'reliable'*, in principle, the learned representation should be general and useful to different types of tasks. Therefore, in this work, we will not focus on performance obtained by using a learned representation for certain machine learning tasks, but rather on a systematic way to verify assumptions on distance relationships between several representation spaces: the audio space and the learned space.

Inspired by [6], we will also investigate the effect of musical and acoustic transformations of audio input signals, in combination with an arbitrary encoder of the input signal, which either may be a conventional feature extractor or deep learning-based encoder. In doing this, we have the following major assumptions:

(i) if a small, humanly imperceptible transformation is introduced, the distance between the original and transformed signal should be very small, both in the audio and encoded space. This is illustrated in Figure 4.1

(ii) however, if a more grave transformation is introduced, the distance between the original and transformed signal should be larger, both in the audio and encoded space.

(iii) the degree of how these assumptions hold will differ for the tasks and the datasets on which the encoder is trained.

To examine the above assumptions, we seek to answer the following research questions:

RQ 1. Do assumption (i) and (ii) hold for conventional and deep learning-based encoders?

RQ 2. Does assumption (iii) hold for music-related tasks and corresponding datasets, especially when deep learning is applied?

By answering the above questions, ultimately we seek to test if considered music-related encoders hold a desirable consistency, such that the distances between audio space and the latent space are monotonically related.

With this work, we intend to offer directions towards a complementary evaluation method for deep machine learning pipelines, that focuses on space diagnosis rather than the troubleshooting of pipeline output. Our intention is that this will provide the researcher with additional insight into the reliability and potential semantic sensitivities of deep learned spaces.

In the remainder of this paper, we first describe our approaches including the details on the learning setup (Section 4.2) and the methodology to assess distance consistency (Section 4.3), followed by the experimental setup (Section 4.4). Further, we report the result from our experiments (Section 5.4). Afterwards we discuss the results and conclude this work (Section 4.6).

4.2. LEARNING

To diagnose a deep music representation space, such a space should first exist. For this, one needs to find a learnable deep encoder $f : \mathbb{R}^{t \times b} \to \mathbb{R}^d$ that transforms the input audio representation $x \in \mathbb{R}^{t \times b}$ to a latent vector $z \in \mathbb{R}^d$, while taking into account the desired output for a given learning task. The learning of f can be done by adjusting the parametrization Θ^f to optimize the objective function, which should be defined in accordance to a given task.

4.2.1. TASKS

In our work, we consider representations learned for four different tasks: *autoencoder* (AE), music *auto-tagging* (AT), *predominant instrument recognition* (IR), and finally singing *voice separation* (VS). By doing this, we take a broad range of problems into account that are particularly common in the MIR field. AE is a representative task for unsupervised learning using DNNs, and AT is a popular supervised learning task in the MIR field [4, 11–15]. AT is a multi-label classification problem, in which individual labels are not always mutually exclusive and often highly inter-correlated. As such, it can be seen as a more challenging problem than IR, which is a single-label classification problem. Furthermore, IR labels involve instruments, which can be seen as more objective and taxonomically stable labels than e.g. genres or moods. Finally, VS is a task that can be formulated as a regression problem, that learns a mask to segregate a certain region of interest out of a given signal mixture.

Autoencoder

The objective of an autoencoder is to find a set of encoder f and decoder g functions such that the input audio x is encoded into a fixed-length vector and reconstructed as follows:

$$\hat{x} = g(f(x)) \tag{4.1}$$

Here, the $\hat{x} = g(f(x))$ is the output of a cascading pipeline of a decoder $g : \mathbb{R}^d \rightarrow \mathbb{R}^{t \times b}$ parameterized by Θ^g, followed by an encoder f. To obtain a desired model, a reconstruction error is typically considered as its loss function:

$$J^{AE} = \sum_{i=1}^{|\mathscr{X}^{tr}|} \|x^i - \hat{x}^i\|_2 \tag{4.2}$$

where $\mathscr{X}_{tr}$ is the given set of training samples for the autoencoder task.

Music Auto-Tagging

Unlike the autoencoder, a DNN model architecture for either multi-label or multi-class classification has architectural block h to infer the posterior distribution of classes from the encoding by f:

$$\hat{y} = \sigma(h(f(x))) \tag{4.3}$$

Since we consider a single fully-connected layer as h in this study, $h : \mathbb{R}^d \rightarrow \mathbb{R}^K$ is the prediction layer parameterized by Θ_h, which transforms the deep representation z^i into the logit per class, which is finally mapped into $p(k|x^i)$ by the sigmoid function σ.

The typical approach to music auto-tagging using DNNs is to consider the problem as a multi-label classification problem, for which the objective is to minimize the binary cross-entropy of each music tag $k \in \{1, 2, ..., K\}$, which is expressed as follows:

$$J^{AT} = -\sum_{i=1}^{|\mathscr{X}^{tr}|} \sum_{k=1}^{K} y_k^i \log(\hat{y}_k^i) + (1 - y_k^i) \log(1 - \hat{y}_k^i) \tag{4.4}$$

where y_k^i is the binary label that indicates whether the tag k is related to the input audio signal x^i. Similarly, $\hat{y}_k^i$ indicates the inferred probability of x^i and tag k. The optimal functions f and h are found by adjusting Θ^f and Θ^h such that (4.4) is minimized.

Predominant Musical Instrument Recognition

The learning of the IR task can be formulated as a single-label, multi-class classification, which allows one to use a model architecture similar to the aforementioned one, except the terminal non-linearity:

$$\hat{y} = softmax(h(f(x))) \tag{4.5}$$

Here, the softmax function $softmax(o_t) = \frac{e^{o_t}}{\sum_{c=1}^{T} e^{o_c}}$, where $o \in \mathbb{R}^T$ is the output of h, substitutes the sigmoid function in (4.3) to output the categorical distribution over the class.

To maximize the classification accuracy, one of the popular loss function especially in the context of neural network learning is categorical cross-entropy, given as follows:

$$J^{IR} = -\sum_{i=1}^{|\mathscr{X}^{tr}|} \sum_{t=1}^{T} y_t^i \log(\hat{y}_t^i) \tag{4.6}$$

where $t \in \{1, 2, ..., T\}$ is a instrument class and thus, y_t^i is the binary label of instance x^i to the class t and $\hat{y}_t^i$ indicates the inferred probability of x^i and instrument t, respectively.

SINGING VOICE SEPARATION

There are multiple ways to set up an objective function for the source separation task. It can be achieved by simply applying (4.2) between the output of the network $\hat{x} = g(f(x))$ and the desired isolated signal $s \in \mathscr{R}^{t \times b}$ such that the model can infer direct isolated sound. In this case, the objective function is similar to (4.2), except that the target is substituted from the input signal x to the isolated signal s. On the other hand, as introduced in [16], one can learn a model predicting the mask that segments the target component from the mixture as follows:

$$\hat{s} = \sigma(g(f(x))) \odot x \tag{4.7}$$

where $\hat{s}$ is the estimated isolated signal and $x \in \mathscr{R}^{t \times b}$ is the representation of the original input mixture, and $\odot$ refers to the element-wise multiplication. $\sigma(g(f(x))) \in \mathscr{R}^{t \times b}$ is the mask inferred by g and f of which the elements are bounded in the range $[0, 1]$ by the sigmoid function σ, such that they can be used for the separation of the target source. As introduced in [16], we applied the skip connections.

For the optimization of the encoder parameters Θ_f and the decoder parameters Θ_g, [16] suggests to use the L1 loss as follows:

$$J^{VS} = \sum_{i=1}^{|\mathscr{X}^{tr}|} \| s^i - \hat{s}^i \|_1 \tag{4.8}$$

where s^i is the low-level representation of the isolated signal, which serves as the regression target. Note, that both input x^i and estimated target source $\hat{s}$ are magnitude spectra, so we use the original phase of input x^i to reconstruct a time-domain signal.

4.2.2. NETWORK ARCHITECTURES

The architecture of a DNN determines the overall structure of the network, which defines the details of the desired patterns to be captured by the learning process [17]. In other words, it reflects the way in which a network should *interpret* a given input data representation. In this work, we use a *VGG-like* architecture, one of the most popular and general architectures frequently employed in the MIR field.

The *VGG-like* architecture is a Convolutional Neural Network (CNN) architecture introduced by [18, 19], which employs tiny rectangular filters. Successes of VGG-like architectures have not only been reported for computer vision tasks, but also in various MIR

Table 4.1: Employed network architectures. A decoder g is constructed reversing the layers: convolution (Conv) and fully-connected (FC) layers are transposed, and pooling layers repeat the maximum input values in the pooling window.

Layers	Output shape
Input	1×128×512
Conv 3×3, BN, ReLU	16×128×512
MaxPooling 2×2	16×64×256
Conv 3×3, BN, ReLU	32×64×256
MaxPooling 2×2	32×32×128
Conv 3×3, BN, ReLU	64×16×64
MaxPooling 2×2	64×8×32
Conv 3×3, BN, ReLU	128×8×32
MaxPooling 2×2	128×4×16
Conv 3×3, BN, ReLU	256×4×16
MaxPooling 2×2	256×2×8
Conv 3×3, BN, ReLU	256×2×8
MaxPooling 2×2	256×1×4
GlobalAveragePooling	256

fields [4, 9]. The detailed architecture design used in our work can be found in the Table 4.1.

4.2.3. Architecture and Learning Details

For both architectures, we used Rectified Linear Units (ReLU) [20] for the nonlinearity, and Batch Normalization (BN) in every convolutional and fully-connected layer for fast training and regularization [21]. We use Adam [22] as optimization algorithm during training, where the learning rate is set for 0.001 across all models. We trained models with respect to their objective function, which requires different optimization strategies. Nonetheless, we regularized the other factors except the number of epochs per task, which inherently depends on the dataset and the task. The termination point of the training is set manually, where either the validation loss reaches to the plateau or starts to increase. More specifically, we stopped the training for each task at the epoch of $\{500, 200, 500, 5000\}$ for the AE, AT, IR, VS task, respectively.

4.3. Measuring Distance Consistency

In this work, among the set of potential representation spaces, we consider two specific subsets of representation spaces of interest: the audio input space and the latent embedding space. Let $\mathscr{A}$ be the space where the low-level audio representation of music excerpts belong to. $\mathscr{X} \subset \mathscr{A}$ is the set of music excerpts in the dataset and $x \in \mathscr{X}$ is each instance. Likewise, $\mathscr{L}$ is the latent space where the set of latent points $z \in \mathscr{Z} \subset \mathscr{L}$ belongs to. Therefore, an encoder $f : \mathscr{A} \rightarrow \mathscr{L}$ is trained on task-specific training data $\mathscr{X}$ and maps points from $\mathscr{X}$ to $\mathscr{Z}$ while it actually maps $\mathscr{A}$ to $\mathscr{L}$. Specifically, a fixed number of latent spaces per task $\{\mathscr{L}_{AE}, \mathscr{L}_{AT}, \mathscr{L}_{IR}, \mathscr{L}_{VS}\}$ are considered. For all relevant encoders, we will assess

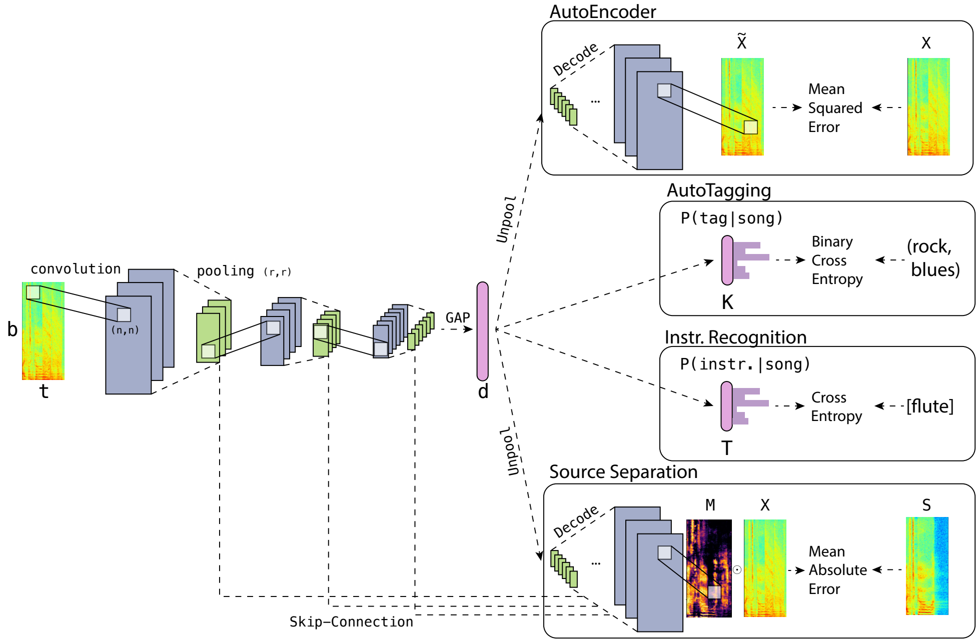


Figure 4.2: Network architecture used in this work. The left half of the model is the encoder pipeline f, whose architecture is kept the same across all the tasks of our experiments. The pink vertical bar represents the latent vector z, in which all the measures we propose are tested. The right half of the diagram refers to the four different prediction pipelines with respect to the tasks. The top block describes the decoder and the error function of the task (where, for simplicity, detailed illustrations of decoder g of f are omitted). The second and third block represent the AT and IR task, respectively. Here, the smaller pink bar represents the terminal layer for the prediction of the posterior distribution for K music tags or T musical instruments. Finally, the lowest block is describing the mask predictor g, prediction process and the way the error function is calculated. Also, this architecture includes the skip-connections from each convolution block of the encoder, which is the key characteristic of the U-Net [23].

their reliability by examining the distance consistency with respect to a set of transformations[1] $\mathscr{T} = \{\tau_l : \mathscr{A} \rightarrow \mathscr{A}, l \in [1,2,...,L]\}$ and a set of testing points $\mathscr{X}^{ts} \subset \mathscr{A}$.

In Section 4.3.1, we describe how distance consistency will be measured. Section 4.3.2 will discuss the distance measures that will be used, while Section 4.3.3 discusses what transformations will be adopted in our experiments.

4.3.1. Distance Consistency

For distance consistency, we will compute *within-space consistency* and *between-space consistency*.

Within-Space Consistency

For all audio samples $x \in \mathscr{X}^{ts}$ and transformations $\tau \in \mathscr{T}$, we obtain the transformed points $x_\tau = \tau(x)$ and $z_\tau = f(x_\tau)$ first, and then we calculate the error function δ of each transformed sample as follows:

$$\delta(p, \mathscr{P}, \tau, d) = \begin{cases} 0, & \text{if } d(p_\tau, p) < d(p_\tau, p'), \forall p' \in \mathscr{P} \setminus p \\ 1 & \text{otherwise} \end{cases} \tag{4.9}$$

Where $p \in \mathscr{P}$ can be either audio samples $x \in \mathscr{X}$ or latent points $z \in \mathscr{Z}$, according to the target space to be measured. Finally, d is a distance function between two objects.

As δ indicates how the space is unreliable at the exemplar-level, the within-space consistency can be defined as the complement of δ:

$$C^W = 1 - \mathbb{E}_{p \in \mathscr{P}}[\delta(p, \mathscr{P}, \tau, d)] \tag{4.10}$$

Between-Space Consistency

To measure consistency between the associated spaces, one can measure how they are correlated. The distances between a transformed point p_τ and its original sample p will be used as characteristic information to make comparisons between spaces. As mentioned above, we consider two specific spaces: the audio input space $\mathscr{A}$ and the embedding space $\mathscr{L}$. Consequently, we can calculate the correlation of distances for the points belonging to each subset of spaces as follows:

$$C^B_\rho = \rho(d^\tau_{\mathscr{A}}, d^\tau_{\mathscr{L}}) \tag{4.11}$$

where ρ is Spearman's rank correlation, and $d^\tau_{\mathscr{A}}$ and $d^\tau_{\mathscr{L}}$ refers to the distance array $d(x_\tau, x')$ and $d(z_\tau, z'), \forall x' \in \mathscr{X}^{ts} \setminus x$, respectively.

On the other hand, one can also simply measure the agreement between distances, which is given by:

$$C^B_{acc} = accuracy(\delta^{d,\tau}_{\mathscr{A}}, \delta^{d,\tau}_{\mathscr{L}}) \tag{4.12}$$

where $accuracy$ denotes the binary accuracy function [24], and $\delta^{d,\tau}_{\mathscr{A}}$ and $\delta^{d,\tau}_{\mathscr{L}}$ denote $\delta(x, \mathscr{X}, \tau, d)$ and $\delta(z, \mathscr{Z}, \tau, d)$, respectively.

[1] Note that, the term 'transformation' differs from the 'maps', which correspond to encoders f in our study. While It is rather close to the concept of 'input perturbation' from literature, we intentionally avoid using the term, since we also study more grave ranges of deformations which are not usually studied.

4.3.2. DISTANCE MEASURES

The main assessment of this work is based on distance comparisons between original clip fragments and their transformations, both in audio and embedded space. To our best knowledge, not many general ways are developed to calculate the distance between raw audio representations of music signals directly. Therefore, we choose to calculate the distance between audio samples using time-frequency representations as the potential proxy of perceptual distance between the music signals. More specifically, we use Mel Frequency Cepstral Coefficients (MFCCs) with 25 coefficients, dropping the first coefficient when the actual distance is calculated. Eventually, we employ two distance measures on the audio domain:

- Dynamic Time Warping (DTW) is a well-known dynamic programming method for calculating similarities between time series. For our experiments, we use the Fast-DTW implementation [25].
- Similarity Matrix Profile (SiMPle) [26] measures the similarity between two given music recordings using a similarity join [26]. We take the median of the profile array as the overall distance between two audio signals.

For deep embedding space, since any deep representation of input x is encoded as a fixed-length vector z in our models, we adopted two general distance measures for vectors: Euclidean distance and cosine distance.

4.3.3. TRANSFORMATIONS

In this subsection, we describe the details on the transformations we employed in our experiment. In all cases, we will consider a range from very small, humanly imperceptible transformations, up to transformations within the same category, that should be large enough to become humanly noticeable. While it is not trivial to set an upper bound for the transformation magnitudes, at which a transformed sample may be recognized as a 'different' song from the original, we introduce a reasonable range of magnitudes, such that we can investigate the overall robustness of our target encoders as transformations will become more grave. The selected range per each transformation is illustrated in Figure 4.3.

- Noise: As a randomized transformation, we applied both pink noise (PN) and environmental noise (EN) transformations. More specifically, for EN, we used noise recorded in a bar, as collected from `freesound`.[2] The test range of the magnitude, expressed in terms of Signal to Noise Ratio, spans from -15dB to 30dB, with denser sampling for high Signal to Noise Ratios (which are situations in which transformed signals should be very close to the original signal) [27]. This strategy also is adopted for the rest of the transformations.
- Tempo Shift: We applied a tempo shift (TS), transforming a signal to a new tempo, ranging from 30% to 150% of the original tempo. Therefore, we both slow down and speed up the signal. Close to the original tempo, we employed a step size of 2%, as a -2% and 2% tempo change has been considered as an irrelevant slowdown

[2] https://freesound.org

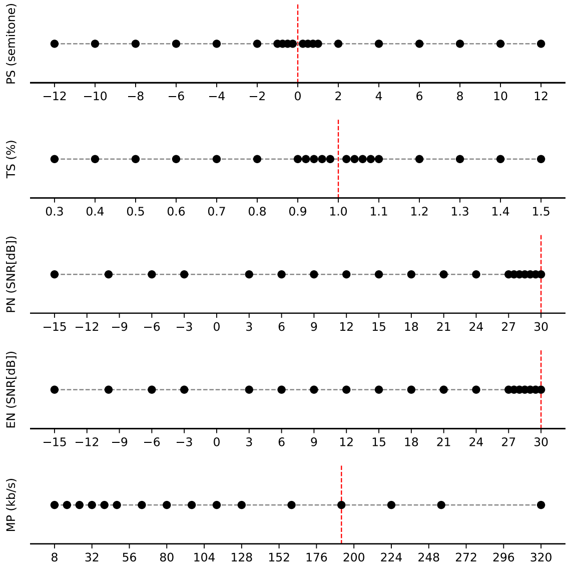


Figure 4.3: The selected range of magnitudes with respect to the transformations. Each row indicates a transformation category; each dot represents the selected magnitudes. We selected relatively more points in the range in which transformations should have small effect, except for the case of MP3 compression. Here, we tested all the possible transformations (kb/s levels) as supported by the compression software we employed. The red vertical lines indicate the position of the original sample with respect to the transformation magnitudes. For TS and PS, these consider no transformation; for PN, EN and MP, they consider the transformation magnitude that will be closest to the original sample.

or speedup in previous work [6]. We employed an implementation[3] using a phase vocoder and resampling algorithm.

- Pitch Shift: We also employed a pitch shift (PS), changing the pitch of a signal, making it lower or higher. Close to the original pitch, we consider transformation steps of ±25 cents, which is 50% smaller than the error bound considered in the MIREX challenge of multiple fundamental frequency estimation & tracking [28]. Beyond a difference of 1 semitone with respect to the original, whole tone interval steps were considered.
- Compression: For compression (MP), we simply compress the original audio sample using an MP3 encoder, taking all kb/s compression rates as provided by the *FFmpeg* software [29].

For the rest of the paper, for brevity, we use OG as the acronym of the original samples.

4.4. EXPERIMENT

4.4.1. AUDIO PRE-PROCESSING

For the input time-frequency representation to the DNNs, we use the dB-scale magnitude STFT matrix. For the calculation, the audio was resampled at 22,050 kHz. The window and overlap size are 1,024 and 256 respectively. It leads to the dimensionality of the frequency axis to be $b = 513$, only taking positive frequencies into account. The standardization over the frequency axis is applied by taking the mean and the standard deviation of all magnitude spectra in the training set.

Also, we use the short excerpts of the original input audio track with $t = 128$, which yields approximately 2 seconds per excerpt in the setup we used. Each batch of excerpts is randomly cropped from 24 randomly chosen music clips before being served to the training loop.

When applying the transformations, it turned out that some of the libraries we used did not only apply the transformation, but also changed the loudness of the transformed signal. To mitigate this, and only consider the actual transformation of interest, we applied a loudness normalization based on the EBU-R 128 loudness measure [30]. More specifically, we calculated the mean loudness of the original sample, and then ensured that transformed audio samples would have equal mean loudness to their original.

4.4.2. BASELINE

Beyond deep encoders, we also consider a conventional feature extractor: MFCCs, as also used in [11]. The MFCC extractor can also be seen as an encoder, that projects raw audio measurements into a latent embedding space, where the projection was hand-crafted by humans to be perceptually meaningful.

We first calculate the first- and second-order time derivatives of the given MFCCs and then take the mean and standard deviation over the time axis, for the original and its derivatives. Finally, we concatenate all statistics into one vector. Using the 25 coefficients excluding the first coefficient, we obtain $z^{MFCC} \in \mathbb{R}^{144}$ from all the points in $\mathcal{X}^{ts}$. For the

[3] `https://breakfastquay.com/rubberband/`

AT task, we trained a dedicated h for auto-tagging, with the same objective as Eq. 4.4, while f is substituted as z^{MFCC}.

4.4.3. DATASET

We use a subset of the Million Song Dataset (MSD) [31] both for training and testing of AT and AE task. The number of the training samples $|\mathcal{X}^{tr}|$ is 71,512. These are randomly drawn from the original subset of 102,161 samples without replacement. For the test set $\mathcal{X}^{ts}$, we used 1,000 excerpts randomly sampled from 1,000 preview clips which are not used at training time. As suggested in [4], we used the top $K = 50$ social tags based on their frequency within the dataset.

As for the IR task, we choose to use the training set of the IRMAS dataset [32], which contains 6,705 audio clips of 3-second polyphonic mixtures of music audio, from more than 2,000 songs. The pre-dominant instrument of each short excerpt is labeled. As excerpts may have been clipped from a single song multiple times, we split the dataset into training, validation and test sets at the song level, to avoid unwanted bleeding among splits.

Finally, for VS, we employed the MUSDB18 dataset [33]. This dataset is developed for musical blind source separation tasks, and has been used in public benchmarking challenges [34]. The dataset consists of 150 unique full-length songs, both with mixtures and isolated sources of selected instrument groups: *vocals*, *bass*, *drums* and *other*. Originally, the dataset is split into a training and test set; we split the training set into a training and validation set (with a 7:3 ratio), to secure validation monitoring capability.

Note that since we use different datasets with respect to the tasks, the measurements we investigate will also depend on the datasets and tasks. However, across tasks, we always use the same encoder architecture, such that comparisons between tasks can still validly be made.

4.4.4. PERFORMANCE MEASURES

As introduced in Section 4.3, we use distance consistency measures as primary evaluation criterion of our work. Next to this, we also measure the performance per employed learning task. For the AE task, the Mean Square Error (MSE) is used as a measure of reconstruction error. For the AT task, we apply a measure derived from the popular Area Under ROC Curve (AUC): more specifically, we apply AUC^C, averaging the AUC measure over clips. As for the IR task, we choose to use accuracy. Finally, as for the VS task, we choose to use the Signal to Distortion Ratio (SDR), which is one of the evaluation measures used in the original benchmarking campaign. For this, we employ the public software as released by the benchmark organizers. While beyond SDR, this software suite also can calculate 3 more evaluation measures (Image to Spatial distortion Ratio (ISR), Source to Interference Ratio (SIR), Sources to Artifacts Ratios (SAR)), in this study, we choose to only employ SDR: the other metrics consider spatial distortion, while this is irrelevant to our experimental setup, in which we only use mono sources.

4.5. Results

In the following subsections, we present the major analysis results for *task-specific performance*, *within-space consistency*, and finally, *between-space consistency*. Shared conclusions and discussions following from our observations will be presented in Section 4.6.

4.5.1. Task-Specific Performance

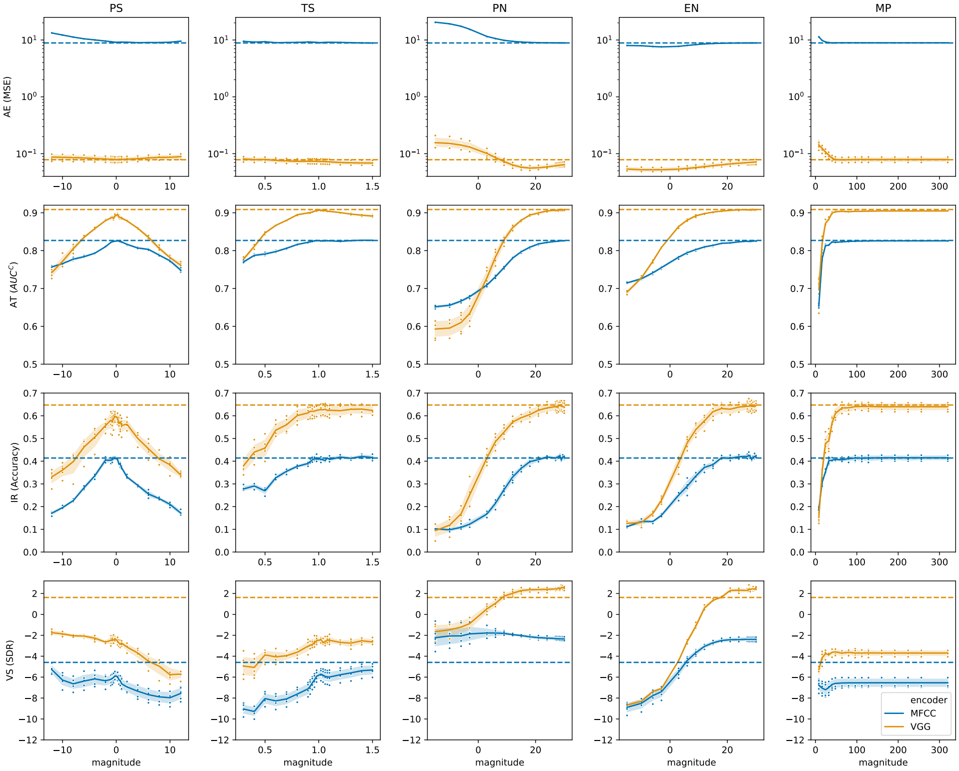


Figure 4.4: Task specific performance results. Blue and yellow curves indicate the performance of different encoders for each task, over the range of magnitude with respect to the transformations. The performance of original samples is indicated as dotted horizontal lines. For the remaining of the paper including this figure, all the confidence intervals are computed with 1,000 bootstraps at the 95% level.

To analyze task-specific performance, we ran predictions for the original samples in $\mathscr{X}^{ts}$, as well as their transformations using all $\tau \in \mathscr{T}$ with all the magnitudes we selected. The overall results, grouped by transformation, task and encoder, are illustrated in Figure 4.4. For most parts, we observe similar degradation patterns within the same transformation type. For instance, in the presence of PN and EN transformations, performance decreases in a characteristic non-linear fashion as more noise is added. The exception seems to be the AE task, which shows somewhat unique trends with a more distinct difference between encoders. In particular, when EN is introduced, performance increases with the severity of the transformation. This is likely to be caused by the fact that the environmental noise that we employed is semantically irrelevant for the other tasks, thus causing a degradation in performance. However, because the AE task just reconstructs the given input audio regardless of the semantic context, and the environmental noise that we use is likely not as complex as music or pink noise, the overall reconstruction gets better.

To better understand the effect of transformations, we fitted a Generalized Additive Model (GAM) on the data, using as predictors the main effects of the task, encoder and transformation, along with their two-factor interactions. Because the relationship between performance and transformation magnitude is very characteristic in each case, we included an additional spline term to smooth the effect of the magnitude for every combination of transformation, task and encoder. In addition, and given the clear heterogeneity of distributions across tasks, we standardized performance scores using the within-task mean and standard deviation scores. Furthermore, MSE scores in the AE task are reversed, so that higher scores imply better performance. The analysis model explains most of the variability ($R^2 = .98$).

An Analysis on Variance (ANOVA) using the marginalized effects clearly reveals that the largest effect is due to the encoders ($F(1,3522) = 12898, p < .0001$), as evidenced by Figure 4.4. Indeed, the VGG-like network has an estimated mean performance of $0.84 \pm .008$ ($mean \pm s.e.$) standardized units, while MFCCs has an estimated performance of $-0.52 \pm .009$ standardized units. The second largest effect is the interaction between transformation and task ($F(12,3522) = 466, p < .0001$), mainly because of the VS task. Comparing the VGG-like and MFCC encoders on the same task ($F(3,3522) = 210, p < .0001$), the largest performance differences appear in the AE task, with VS showing the smallest differences. It suggests that MFCCs loses a substantial amount of information required for reconstruction, while a neural network is capable of maintaining sufficient information to do a reconstruction task. The smallest performance differences in the VS task mostly relate to the performance of the VGG-like encoder, that shows substantial performance degradation in response to the transformations. Figure 4.5 shows the estimated mean performance.

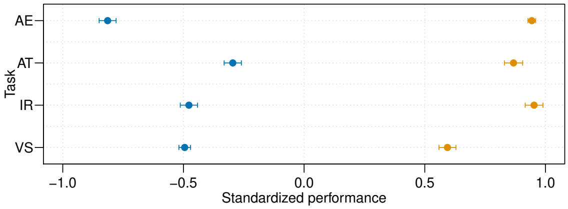


Figure 4.5: Estimated marginal mean of standardized performance by encoders and tasks, with 95% confidence intervals. Blue points and brown points indicate the performance of MFCC and VGG-like, respectively.

4.5.2. Within-Space Consistency

In terms of within-space consistency, we first examine the original audio space $\mathscr{A}$. As depicted in Figure 4.6, both the DTW and SiMPle measures show very high consistency for small transformations. As transformations have higher magnitude, as expected, the consistency decreases, but at different rates, depending on the transformation. The clear exception is the TS transformation, where both measures, and in particular DTW, are highly robust to the magnitude of the shift. This result implies that the explicit consideration of both measures on the temporal dynamics can be beneficial.

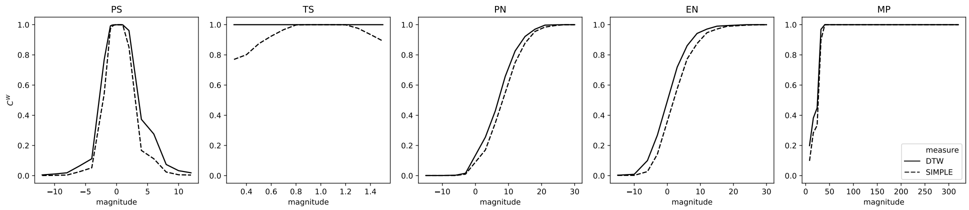


Figure 4.6: Within-space consistency by transformation on the audio space $\mathscr{A}$. Each curve indicates the within-space consistency C^W.

With respect to the within-consistency of the latent space, Figure 4.7 and 4.8 depicts the results for both the Euclidean and cosine distance measures. In general, the trends are similar to those found in Figure 4.6. For analysis, we fitted a similar GAM model, including the main effect of the transformation and task, their interaction, and a smoother for the magnitude of each transformation within each task. When modeling consistency with respect to Euclidean distance, this analysis model achieved $R^2 = .98$. An ANOVA analysis shows very similar effects due to transformation ($F(4,1793) = 1087, p < .0001$) and due to tasks ($F(4,1793) = 1066, p < .0001$), with a smaller effect of the interaction. In particular, the model confirms the observation from the plots that the MFCC encoder has significantly higher consistency ($0.741 \pm .014$) than the others. For the VGG-like cases, AT shows the highest consistency ($0.671 \pm .007$), followed by IR ($0.539 \pm .008$), VS ($0.331 \pm .007$) and lastly by AE ($0.17 \pm .006$). As Figure 4.8 shows, all these differences are statistically significant.

A similar model to analyze consistency with respect to the cosine distance yielded very similar results ($R^2 = 0.981$). However, the effect of the task ($F(4,1794) = 1263, p < .0001$) was larger than the effect of the transformation ($F(4,1794) = 913, p < .0001$), indicating that the cosine distance is slightly more robust to transformations than the Euclidean distance.

To investigate observed effects more intuitively, we visualize in Figure 4.9 the original dataset samples and their smallest transformations, which should be hardly perceptible to imperceptible to human ears [6, 9, 28][4] in a 2-dimensional space, using t-SNE [35]. In MFCC space, (Figure 4.9), the distributions of colored points, corresponding to each of the transformation categories, are virtually identical to those of the original points. This matches our assumption that very subtle transformations, that humans will not easily recognize, should stay very close to the original points. Therefore, if the hidden latent embedded space had high consistency with respect to the audio space, the distribution of colored points should be virtually identical to the distribution of original points. However, this is certainly not the case for neural networks, especially for tasks such as AE and VS (see Figure 4.9). For instance, in the AE task every transformation visibly causes clusters that do

[4] The smallest transformations are ±25 cents in PS, ±2% in TS, 30dB in PN and EN, and 192 kb/s in MP.

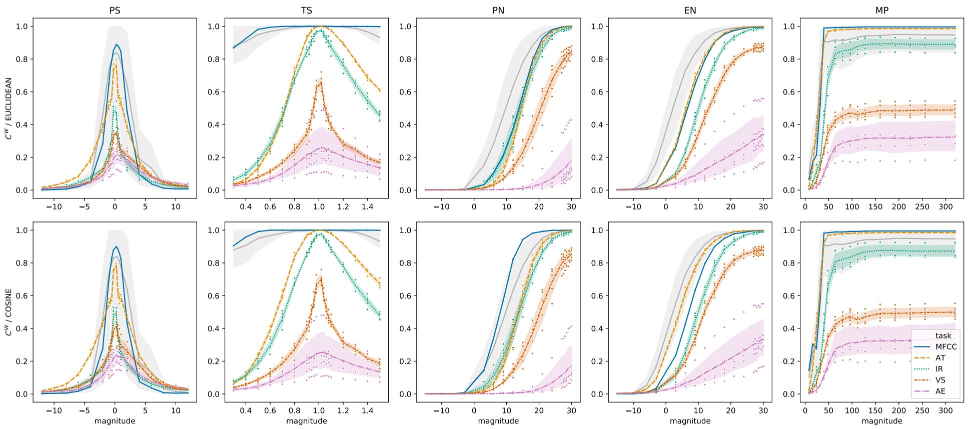


Figure 4.7: Within-space consistency by transformation on the latent space $\mathscr{L}$. Each curve indicates the within-space consistency C^W by task and transformation. The gray curves indicate C^W on $\mathscr{A}$, taken as a weak upper bound for the consistency in the latent space. Confidence intervals are drawn at the 95% level. Points indicate individual observations from different trials.

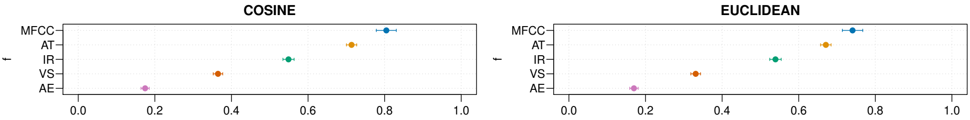


Figure 4.8: Estimated marginal mean within-space consistency C^W in the latent domain. Confidence interval are at 95% level.

not cover the full space. This suggests that the model may recognize transformations as important *features*, characterizing a subset of the overall problem space.

4.5.3. Between-Space Consistency

Next, we discuss between-space consistency according to C^B_{acc} and C^B_{ρ}, as discussed in Section 4.3.1. As in the previous section, we first provide a visualization of the relationship between transformations and consistency, and then employ the same GAM model to analyze individual effects. The analysis will be presented for all pairs of distance measures and between-space consistency measures, which results in 4 models for C^B_{acc} and another 4 models for C^B_{ρ}. As in the within-space consistency analysis, we set the MFCC and other VGG-like networks from different learning tasks as independent ‘encoder’ f to a latent embedded space.

Accuracy: C^B_{acc}

The between-space consistency, according to the C^B_{acc} criterion, is plotted in the upper plots of Figure 4.10. Comparing this plot to the within-space consistency plots for $\mathscr{A}$ (Figure 4.6) and $\mathscr{L}$ (Figure 4.8), one trend is striking: when within-space consistency in $\mathscr{A}$ and $\mathscr{L}$ becomes substantially low, the between-space consistency C^B_{acc} becomes high.

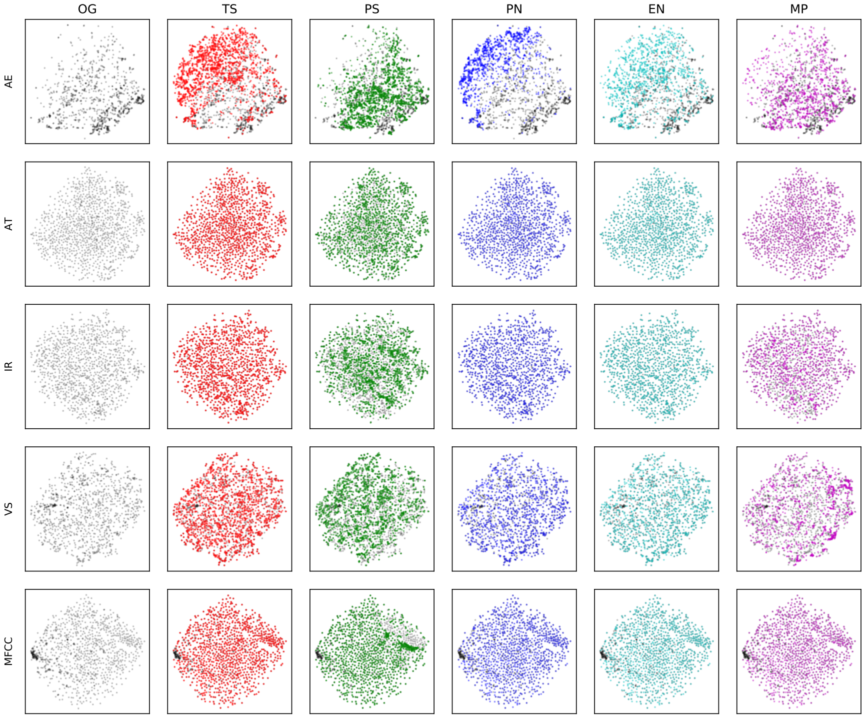


Figure 4.9: Scatter plot of encoded representations and their transformations for baseline MFCC and f encoders with respect to the tasks we investigated. For all panes, black points indicate original audio samples in the encoded space, and the colored, overlaid points indicate the embeddings of transformations according to the indicated category.

This can be interpreted: when grave transformations are applied, the within-space consistencies in both $\mathscr{A}$ and $\mathscr{L}$ space will converge to 0, and comparing the two spaces, this behavior is consistent.

A first model to analyze the between-space consistency with respect to the SiMPle and cosine measures ($R^2 = .96$), reveals that the largest effect is that of the task/encoder $F(4,1772) = 440, p < .0001$), followed by the effect of the transformation ($F(4,1772) = 285, p < .0001$). The left plot of the first row in Figure 4.11 confirms that the estimated consistency of the MFCC encoder (0.796±.015) is significantly higher than that of the VGG-like alternatives, which range between 0.731 and 0.273. In fact, the relative order is the same as observed in the within-space case: MFCC is followed by AT, IR, VS, and finally AE.

We separately analyzed the data with respect to the other three combinations of measures, and found very similar results. The largest effect is due to the task/encoder, followed by the transformation; the effect of the interaction is considerably smaller. As the first rows of Figure 4.11 shows, the same results are observed in all four cases, with statistically significant differences among tasks.

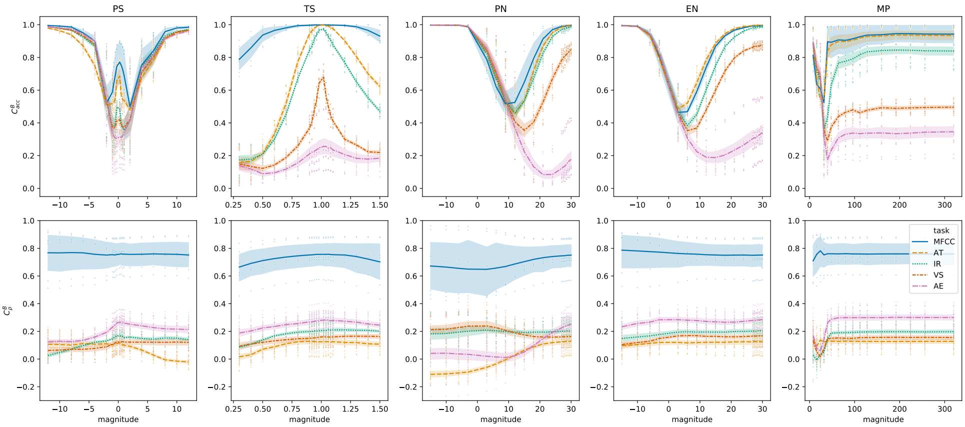


Figure 4.10: C^B_{acc} (top) and C^B_ρ (bottom) between-space consistency by transformation and magnitude. Each curve indicates the between-space consistency C^B with respect to the task. Confidence intervals are drawn at the 95% level. Points indicate individual observations from different trials.

Correlation: C^B_ρ

The bottom plots in Figure 4.10 show the results for between-space consistency measured with C^B_ρ. It can be clearly seen that MFCC preserves the consistency between spaces much better than VGG-like encoders, and in general, all encoders are quite robust to the magnitude of the perturbations.

Analyzing data again using a GAM model confirms these observations. For instance, when analyzing consistency with respect to the DTW and Euclidean measures ($R^2 = 0.96$), the largest effect is by far that of the task/encoder ($F(4,1877) = 6549, p < .0001$), with the transformation and interaction effect being two orders of magnitude smaller. This is because of the clear superiority of MFCC, with an estimated consistency of $0.881 \pm .004$, fol-

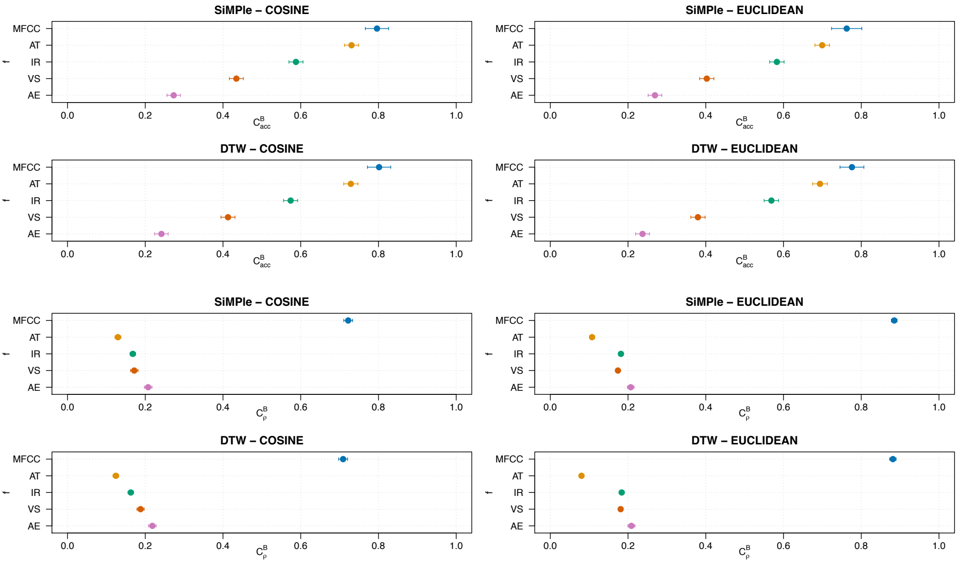


Figure 4.11: Estimated marginal means for between-space consistency by encoder f. The first and second rows are for C^B_{acc} and the third and fourth rows are for C^B_ρ. Confidence intervals are at the 95% level.

lowed by AE (0.209 ± .005), IR (0.184 ± .003), VS (0.181 ± .002) and finally AT (0.08 ± .003) (see right plot of the fourth row in 4.11).

As before, we separately analyzed the data with respect to the other three combinations of measures, and found very similar results. As first two rows of Figure 4.11 shows, the same qualitative observations can be made in all four cases, with statistically significant differences among tasks. Noticeably, the superiority of MFCC is even clearer when employing the Euclidean distance. Finally, another visible difference is that the relative order of VGG-like networks is reversed with respect to C^B_{acc}, with AE being the most consistent, followed by VS, IR, and finally AT.

4.5.4. SENSITIVITY TO IMPERCEPTIBLE TRANSFORMATIONS

TASK-SPECIFIC PERFORMANCE

In this subsection, we focus more on the special cases of transformations with a magnitude small enough to hardly be perceivable by humans [6, 9, 28] As the first row of Figure 4.12 shows, performance is degraded even with such small transformations, confirming the findings from [6]. In particular, the VS task shows more variability among transformations compared to other tasks. Between transformations, the PS cases show relatively higher degradation.

WITHIN-SPACE CONSISTENCY

The second row of Figure 4.12 illustrates the within-space consistency on the $\mathscr{L}$ space when considering these smallest transformations. As before, there is no substantial difference between the distance metrics. In general, the MFCC, AT, and IR encoder/tasks are

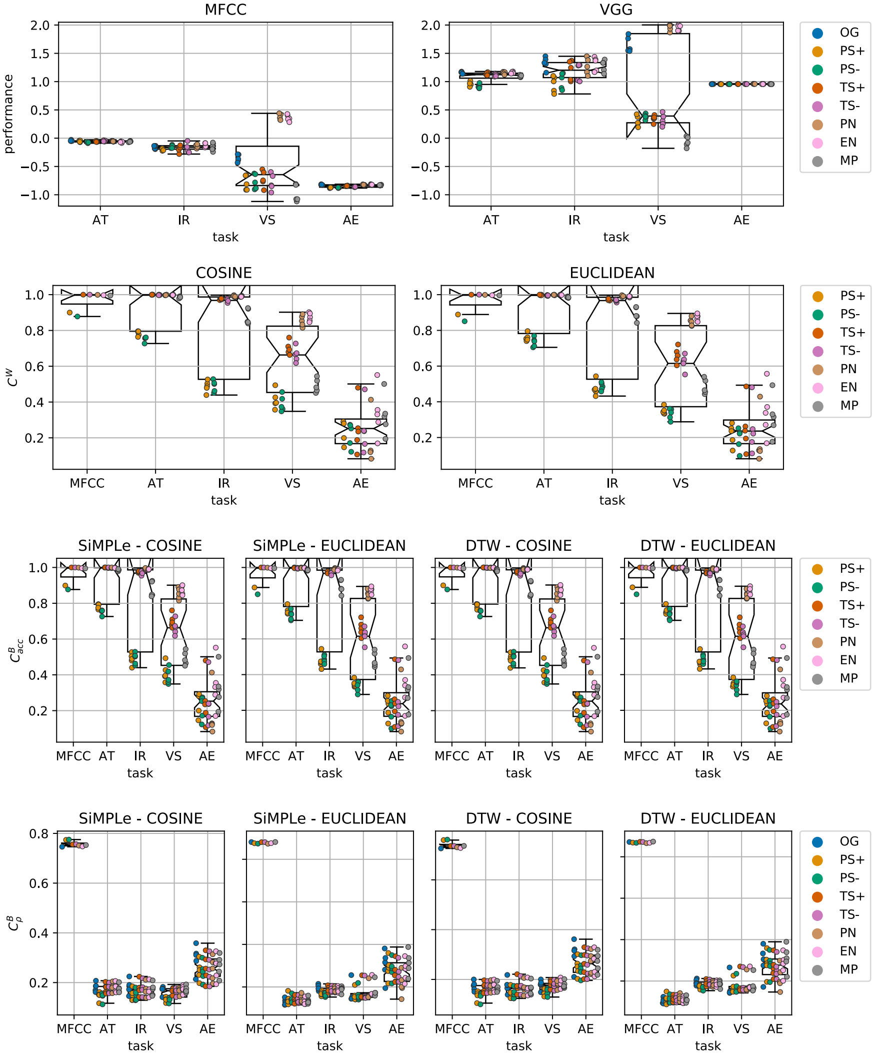


Figure 4.12: Performance, within-space consistency, and between-space consistency distribution on the minimum transformations. The points are individual observations with respect to the transformation types. For PS and TS, we distinguish in the direction of the transformation (+: pitch/tempo up, -: pitch/tempo down). The first row indicates the task-specific performance, and the second row depicts the within-space consistency C^W, and finally, the third and fourth rows show the between-space consistency C^B_{acc} and C^B_ρ, respectively. The performance is standardized per task, and the sign of AE performance is flipped, similarly to our analysis models.

relatively robust on these small transformations, with their median consistencies close to 1. However, encoders trained on the VS and AE tasks show undesirably high sensitivity to these small transformations. In this case, the effect of the PS transformations is even more clear, causing considerable variance for most of the tasks. The exception is AE, which is more uniformly spread in the first place.

Between-Space Consistency

Finally, the between-space consistencies on the minimum transformations are depicted in the last two rows of Figure 4.12. First, we see no significant differences between pairs of distance measures. When focusing on C^B_{acc}, the plots highly resemble those from 4.5.4, which can be expected, because the within-space consistency on $\mathscr{A}$ is approximately 1 for all these transformations, as illustrated in Figure 4.6. On the other hand, when focusing on C^B_ρ, The last row of Figure 4.12 shows that even such small transformations already result in large inconsistencies between spaces when employing neural network representations.

4.6. Discussion and Conclusion

4.6.1. Effect of the Encoder

For most of our experiments, the largest differences are found between encoders. As is well-known, the VGG-like deep neural network shows *significantly better task-specific performance* in comparison to the MFCC encoder. However, when considering distance consistency, MFCC is shown to be *the most consistent encoder* for all cases, with neural network approaches performing substantially worse in this respect. This suggests that, in case a task requires robustness to potential musical/acoustical deviations in the audio input space, it may be more preferable to employ MFCCs than neural network encoders.

4.6.2. Effect of the Learning Task

Considering the neural networks, our results show that the choice of learning task is the most important factor affecting consistency. For instance, a VGG-like network trained on the AE task seems to preserve the relative distances among samples (high C^B_ρ), but individual transformed samples will fall closer to originals that were not the actual original the transformation was applied to (low C^B_{acc}). On the other hand, a task like AT yields high consistency in the neighborhood of corresponding original samples (high C^B_{acc}), but does not preserve the general structure of the audio space (low C^B_ρ). This means that a network trained on a low-level task like AE is more consistent than a network trained on a high-level task like AT, because the resulting latent space is less morphed and it more closely resembles the original audio space. In fact, in our results we see that the semantic high-levelness of the task (AT > IR > VS > AE) is positively correlated with $C^B_a cc$, while negatively correlated with C^B_ρ.

To further confirm this observation, we also computed the between-space consistency C^B_ρ only on the set of original samples. The results, in Figure 4.13, are very similar to those in the last two rows of Figure 4.11 and 4.12. This suggests that in general, the global distance structure of an embedded latent space with respect to the original samples generalizes over the vicinity of those originals, at least for the transformations that we employed.

Considering that AE is an unsupervised learning task, and its objective is merely to

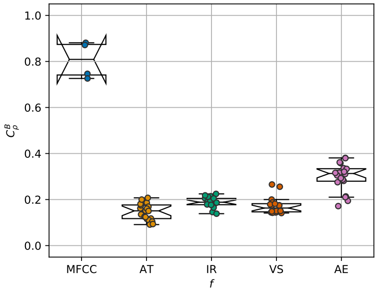


Figure 4.13: C_ρ^B on the original samples, including all the possible distance pairs between audio and latent domain.

embed an original data point into a low-dimensional latent space by minimizing the reconstruction error, the odds are lower that data points will cluster according to more semantic criteria, as implicitly encoded in supervised learning tasks. For instance, in contrast, the VS task should morph the latent space such, that input clips with similar degrees of 'vocalness' should fall close together, as indeed is shown in Figure 4.14. As the task becomes more complex and high-level, such as with AT, this clustering effect will become more multi-faceted and complex, potentially morphing the latent space with respect to the semantic space that is used as the source of supervision.

4.6.3. Effect of the Transformation

Across almost all experimental results, significant differences between transformation categories are observed. On the one hand, this supports the findings from [6, 9], which show the vulnerability of MIR systems to small audio transformations. On the other hand, this also implies that different types of transformations have different effects on the latent space, as depicted in Figure 4.7.

4.6.4. Are Nearby Neighbors Relatives?

As depicted in Figure 4.7, substantial inconsistencies emerge in $\mathscr{L}$ when compared to $\mathscr{A}$. Clearly, these inconsistencies are not desirable, especially when the transformations we applied are not supposed to have noticeable effects. However, as our consistency investigations showed, the MFCC baseline encoder behaves surprisingly well in terms of consistency, evidencing that hand-crafted features should not always be considered as inferior to deep representations.

While in a conventional audio feature extraction pipeline, important salient data patterns may not be captured due to accidental human omission, our experimental results indicate that DNN representations may be unexpectedly unreliable. In the deep music embedding space,'known relatives' in the audio space may suddenly become faraway pairs. That a representation has certain unexpected inconsistencies should be carefully studied and taken into account, specially given the increasing interest in applying transfer learning using DNN representations, not only in the MIR field. For example, if a system requires to use degraded audio inputs for a pre-trained DNN (which e.g. may be done in

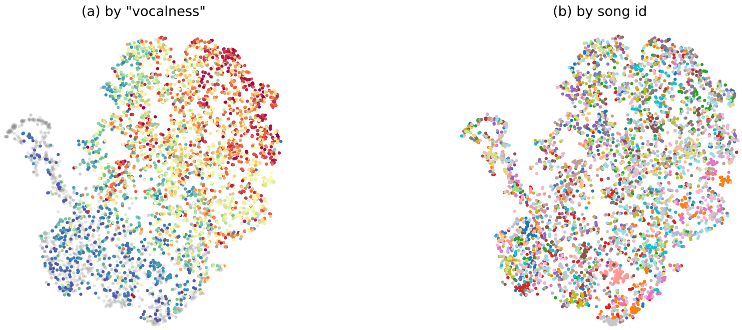


Figure 4.14: 2-dimensional scatter plot using t-SNE. Each point represents 2-second audio mixture signal chunks that are encoded by a VS-specialized encoder. In the left plot, the color map of points is based on the loudness of the isolated vocal signal for a given mixture signal. The red color indicates higher loudness, and the blue color indicates smaller loudness. On the right plot, the same chunks are colored by the song each chunk belongs to. The samples are randomly sampled from the MUSDB18 dataset.

music identification tasks), while humans may barely recognize the differences between the inputs and their original form, it does not guarantee that this transformed input may be embedded at a similar position to its original version in a latent space.

4.6.5. Towards Reliable Deep Music Embeddings

In this work, we proposed to use several distance consistency-based criteria, in order to assess whether representations in various spaces can be deemed as consistent. We see this as a complementary means of diagnosis beyond task-related performance criteria, when aiming to learn more general and robust deep representations. More specifically, we investigated whether deep latent spaces are consistent in terms of distance structure, when smaller and larger transformations on raw audio are introduced (*RQ 1*). Next to this, we investigated how various types of learning tasks used to train deep encoders impact the consistencies (*RQ 2*).

Consequentially, we conducted an experiment employing 4 MIR tasks, and considering deep encoders versus a conventional hand-crafted MFCC encoder, to measure the consistency for different scenarios. Our findings can be summarized as follows:

RQ 1. Compared to the MFCC baseline, all DNN encoders indicate lower consistency, both in terms of within-space consistency and between-space consistency, especially when transformations grow from imperceptibly small to larger, more perceptible ones.

RQ 2. Considering learning tasks, the high-levelness of a task is correlated with the consistency of resulting encoder. For instance, an AT-specialized encoder, which needs to deal with semantically high-level task, yields the highest within-space consistency, but the lowest between-space consistency. On the other hand, an AE-specialized encoder, which deals with a semantically low-level task, shows opposite trends.

To realize a fully robust testing framework, there still are a number of aspects to be investigated. First of all, more in-depth study is required considering different magnitudes in the transformations, and their possible comparability. While we applied different magnitudes for each transformations, we decided not to comparatively consider the magnitude ranges in the analysis at this moment. This was done, as we do not have any exact means to compare the perceptual effect of different magnitudes, which will be crucial to regularize between transformations.

Furthermore, similar analysis techniques can be applied to more diverse settings of DNNs, including different architectures, different levels of regularizations, and so on. Also, as suggested in [9, 10], the same measurement and analysis techniques can be used for *adversarial examples* generated from the DNN itself, as another important means of studying a DNN's reliability.

Moreover, and based on the observations from our study, it may be possible to develop countermeasures for maintaining high consistency of a model, while yielding high task-specific performance. For instance, unsupervised de-noising such as [36, 37] might be one of the potential solutions. In particular, it can be used when the noise is drawn from the known, relatively simple distribution, such as white noise. However, we also observed some encoders are substantially affected by a very small amount of the noise, which implies even artifacts produced from the de-noising algorithm can cause another unexpected inconsistency. Also, it might not guarantee more musical and structured cases such as tempo or pitch shifts.

For those cases, it can be effective if, during learning, a network is directly supervised to treat transformations in similar ways as their original versions in the latent space. This can be implemented as an auxiliary objective to the main objective of the learning procedure, or introducing directly the transformed examples as the data augmentation.

We believe that our work can be a step forward towards a practical framework for more interpretable deep learning models, in the sense that we suggest a less task-dependent measure for evaluating a deep representation, that still is based on known semantic relationships in the original item space.[5]

REFERENCES

[1] J. Kim, J. Urbano, C. C. S. Liem, and A. Hanjalic, *Are nearby neighbors relatives? testing deep music embeddings,* Frontiers in Applied Mathematics and Statistics **5**, 53 (2019).

[2] A. van den Oord, S. Dieleman, and B. Schrauwen, *Deep content-based music recommendation,* in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2015. December 5-8, 2013, Lake Tahoe, Nevada, United States*, edited by C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger (2013) pp. 2643–2651.

[5] The Python code that is used for this experiment can be found in `https://github.com/eldrin/are-nearby-neighbors-relatives`

[3] E. J. Humphrey, J. P. Bello, and Y. LeCun, *Moving beyond feature design: Deep architectures and automatic feature learning in music informatics,* in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012,* edited by F. Gouyon, P. Herrera, L. G. Martins, and M. Müller (FEUP Edições, 2012) pp. 403–408.

[4] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, *Convolutional recurrent neural networks for music classification,* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (IEEE, 2017) pp. 2392–2396.

[5] P. Chandna, M. Miron, J. Janer, and E. Gómez, *Monoaural audio source separation using deep convolutional neural networks,* in *Latent Variable Analysis and Signal Separation - 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings*, Lecture Notes in Computer Science, Vol. 10169, edited by P. Tichavský, M. Babaie-Zadeh, O. J. J. Michel, and N. Thirion-Moreau (2017) pp. 258–266.

[6] B. L. Sturm, *A simple method to determine if a music information retrieval system is a "horse",* IEEE Trans. Multim. **16**, 1636 (2014).

[7] F. Rodríguez-Algarra, B. L. Sturm, and H. Maruri-Aguilar, *Analysing scattering-based music content analysis systems: Where's the music?* in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, edited by M. I. Mandel, J. Devaney, D. Turnbull, and G. Tzanetakis (2016) pp. 344–350.

[8] B. L. Sturm, *The "horse" inside: Seeking causes behind the behaviors of music content analysis systems,* Comput. Entertain. **14**, 3:1 (2016).

[9] C. Kereliuk, B. L. Sturm, and J. Larsen, *Deep learning and music adversaries,* IEEE Trans. Multim. **17**, 2059 (2015).

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings,* edited by Y. Bengio and Y. LeCun (2015).

[11] K. Choi, G. Fazekas, M. B. Sandler, and K. Cho, *Transfer learning for music classification and regression tasks,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017,* edited by S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull (2017) pp. 141–149.

[12] J. Lee, T. Kim, J. Park, and J. Nam, *Raw waveform-based audio classification using sample-level CNN architectures,* CoRR **abs/1712.00866** (2017), arXiv:1712.00866 .

[13] J. Lee, J. Park, K. L. Kim, and J. Nam, *Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms,* in *14th Sound and Music Computing Conference, SMC* (2017).

[14] S. Dieleman and B. Schrauwen, *End-to-end learning for music audio,* in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014* (IEEE, 2014) pp. 6964–6968.

[15] J. Lee, J. Park, K. L. Kim, and J. Nam, *SampleCNN: End-to-end deep convolutional neural networks using very small filters for music classification,* Applied Sciences **8** (2018), 10.3390/app8010150.

[16] A. Jansson, E. J. Humphrey, N. Montecchio, R. M. Bittner, A. Kumar, and T. Weyde, *Singing voice separation with deep U-Net convolutional networks,* in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, edited by S. J. Cunningham, Z. Duan, X. Hu, and D. Turnbull (2017) pp. 745–751.

[17] I. J. Goodfellow, Y. Bengio, and A. C. Courville, *Deep Learning,* Adaptive computation and machine learning (MIT Press, 2016).

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks,* Commun. ACM **60**, 84 (2017).

[19] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2015).

[20] V. Nair and G. E. Hinton, *Rectified linear units improve restricted boltzmann machines,* in *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, edited by J. Fürnkranz and T. Joachims (Omnipress, 2010) pp. 807–814.

[21] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift,* in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, JMLR Workshop and Conference Proceedings, Vol. 37, edited by F. R. Bach and D. M. Blei (JMLR.org, 2015) pp. 448–456.

[22] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, edited by Y. Bengio and Y. LeCun (2015).

[23] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional networks for biomedical image segmentation,* in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, Lecture Notes in Computer Science, Vol. 9351, edited by N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi (Springer, 2015) pp. 234–241.

[24] C. E. Metz, *Basic principles of ROC analysis.* Seminars in nuclear medicine **8 4**, 283 (1978).

[25] S. Salvador and P. Chan, *FastDTW: Toward accurate dynamic time warping in linear time and space,* in *3rd International Workshop on Mining Temporal and Sequential Data (TDM-04)* (Citeseer, 2004).

[26] D. F. Silva, C. M. Yeh, G. E. A. P. A. Batista, and E. J. Keogh, *SiMPle: Assessing music similarity using subsequences joins,* in *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016*, edited by M. I. Mandel, J. Devaney, D. Turnbull, and G. Tzanetakis (2016) pp. 23–29.

[27] A. Kurakin, I. J. Goodfellow, and S. Bengio, *Adversarial examples in the physical world,* in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings* (OpenReview.net, 2017).

[28] J. Salamon and J. Urbano, *Current challenges in the evaluation of predominant melody extraction algorithms,* in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, edited by F. Gouyon, P. Herrera, L. G. Martins, and M. Müller (FEUP Edições, 2012) pp. 289–294.

[29] S. Tomar, *Converting video formats with FFmpeg,* Linux Journal **2006**, 10 (2006).

[30] EBU, *Loudness normalisation and permitted maximum level of audio signals,* (2010).

[31] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, *The million song dataset,* in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, edited by A. Klapuri and C. Leider (University of Miami, 2011) pp. 591–596.

[32] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, *A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals,* in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, edited by F. Gouyon, P. Herrera, L. G. Martins, and M. Müller (FEUP Edições, 2012) pp. 559–564.

[33] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, *The MUSDB18 corpus for music separation,* (2017).

[34] F. Stöter, A. Liutkus, and N. Ito, *The 2018 signal separation evaluation campaign,* in *Latent Variable Analysis and Signal Separation - 14th International Conference, LVA/ICA 2018, Guildford, UK, July 2-5, 2018, Proceedings*, Lecture Notes in Computer Science, Vol. 10891, edited by Y. Deville, S. Gannot, R. Mason, M. D. Plumbley, and D. Ward (Springer, 2018) pp. 293–305.

[35] L. van der Maaten and G. E. Hinton, *Visualizing data using t-SNE,* Journal of machine learning research **9**, 2579 (2008).

[36] M. Nazeer, N. Bibi, A. Wahab, Z. Mahmood, T. Akram, S. R. Naqvi, H. S. Oh, and D. Kim, *Image de-noising with subband replacement and fusion process using bayes estimators,* Comput. Electr. Eng. **70**, 413 (2018).

[37] M. Nazeer, N. Bibi, A. Jahangir, and Z. Mahmood, *Image denoising with norm weighted fusion estimators,* Pattern Anal. Appl. **21**, 1013 (2018).

5

EVALUATION FRAMEWORK FOR MODEL-AGNOSTIC EXPLAINERS: A CASE STUDY ON RECOMMENDER SYSTEMS

Due to the increasing employment of complex, black-box machine learning methods, explainability of these models has become an important challenge in modern machine learning. One way to tackle this is to apply model-agnostic explanation methods, which interpret the black-box models in a post-hoc manner. While various model-agnostic methods have been introduced, solid evaluation frameworks do not exist for them yet. In this paper, we address this gap, proposing a framework that allows for comparative evaluation of model-agnostic explanation methods, employing a standardized perspective on explanation effectiveness, and explicitly taking into account that the choice of target systems and datasets will affect performance in a hierarchical, multi-level fashion. We demonstrate an application of our framework to the recommender systems domain, considering multiple recommenders, explainers and datasets. Our results indicate that linear recommenders, explainers with sparsity constraints and user features have positive effects on explanation effectiveness.

5.1. INTRODUCTION

In supervised machine learning, formerly popular models such as linear regression or decision trees were interpretable: the associative rules they learned between dependent and independent variables could easily be understood and explained by humans. These days, in many cases, such models are deemed too simple to capture the complex nature of real-world data. Instead, more powerful, but much more complex models have become the standard. While in terms of performance, these models are superior, concerns have arisen that their decisions are black-box, and cannot easily be comprehended by users with limited resources, nor by machine learning practitioners themselves.

As a consequence, where the need for explainability has been acknowledged for long in complex decision-making application domains, such as recommender systems [1], recently, multiple works have emerged addressing the explainability of black-box machine learning models [2–4]. In this, post-hoc, **model-agnostic** explanation methods can be applied, which seek to fit an interpretable model to any black-box target model. While the interpretable model should be as close as possible to the black-box model (**fidelity**), at the same time, it will be constrained to have low **complexity**, such that the logic explaining the black-box target model can still be humanly understood. In optimizing the fit of the interpretable model, a fidelity-complexity trade-off is often observed [4].

While the need for model-agnostic explainability of complex machine learning models is increasing, it is an open problem to evaluate how good an explanation is. While the common agreement is that model-agnostic explanations should be 'as close as possible' to a target model, effectiveness has been formalized in various ways, e.g. by measuring to what extent an explanation successfully retrieves relevant attributes [4, 5], or surrogate models are coherent with respect to the target black-box model [6, 7].

Furthermore, no solid frameworks exist yet to assess how multiple alternative explanation techniques compare against one another. For example, where in the broader machine learning world, the Local Interpretable Model-agnostic Explanation (LIME) technique [4] has gained considerable attention, several years earlier, saliency maps had been proposed in the computer vision domain [8], which also can be seen as interpretable model-agnostic explanations.

Next to this, while the explanation methods are applied to potentially complex machine learning models, in their turn, these machine learning models often capture complex underlying problems, causing multi-level variability. The performance of an explanation will be closely tied to the property of the target system. However, if the target system is the outcome of a machine learning procedure, it has been instantiated in a data-driven manner, thus depending on the data distribution and objective function that were used during training and validation.

Ultimately, the goal of an explanation is human understandability. However, as recently was shown, humans may not be solid judges of an explanation's effectiveness, and can e.g. be biased by social factors when having to select a preferred explanation method [9]. Given all these considerations, a clear need is emerging for systematic, objective and comparative evaluation methods for assessing the effectiveness of model-agnostic explanation techniques.

In this work, we address this need, proposing an evaluation framework for model-agnostic explanation methods. Our framework explicitly addresses possible variability induced by multiple systems and multiple datasets; as such, to the best of our knowledge, it provides the richest perspective so far on what makes for robust explanation method performance.

Beyond introducing the framework, we will illustrate how it can be applied, considering it in the context of Recommender Systems (RS). As argued before, RS pose complex decision-making challenges. Operating on large collections of items, they should automatically surface useful items for users; however, users may need explanations on why

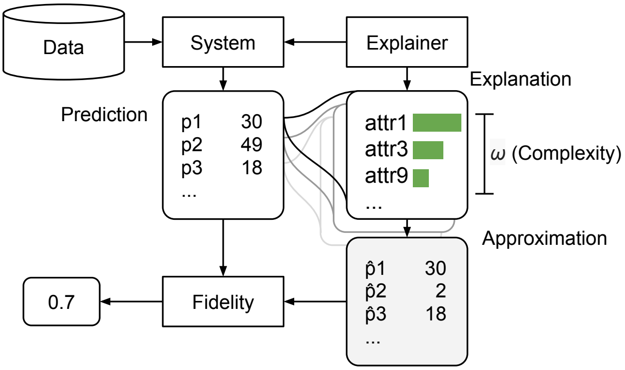


Figure 5.1: Overview illustrating the point estimation of the explainability with a single system, explainer, dataset and use case.

certain items got surfaced [1]. With many recent successful RS having high recommendation accuracy, but lower interpretability due to the complexity of the underlying machine learning models [10–13], they form a natural real-world application domain for our framework. To account for multiple sources of variability, and to be able to do statistically robust assessments, our experiments will consider multiple types of RS, for which multiple explaination methods are assessed, considering multiple datasets to which these RS can be applied. As a consequence, our results will yield useful insights into how explainability for RS can be understood and improved.

5.2. FRAMEWORK

A good explanation model should maximize the fidelity with respect to its black-box target model, while minimizing complexity. Following [4], a score for the total explainability $\mathscr{E}$ can be generally described as follows:

$$\mathscr{E}(G; f, \mathscr{X}) = \mathbb{E}_{x \in \mathscr{X}}[\mathscr{L}(f, g)] - \Omega(g) \tag{5.1}$$

where f is the target black-box model instantiated from the model class F. g is the instantiation of an explanation derived from the explainer class G. The complexity of the explanation model g is represented by Ω, such that an optimal solution holds minimal complexity as well as the maximum fidelity. Practically, for the fidelity $\mathscr{L}$, any function can be chosen that quantifies how closely $f(x)$ and $g(x)$ match, for any input observation $x \in \mathscr{X}$, where $\mathscr{X}$ is the dataset used for the black-box model.

Considering this definition, when seeking to find an estimate for $\mathscr{E}$ in case of a single instantiation g of a model-agnostic model class G, a setup can be envisioned as illustrated in Figure 5.1. Here, a target arbitrary black-box system f is instantiated from a dataset $\mathscr{X}$, generating prediction outputs p in response to each x of interest. Explanation is given by g with a certain complexity ω.

Specifically, in our current framework, we will assume that each input observation x can be described by a limited vocabulary of user-understandable discrete attributes such as tags. The explainer will need to generate weights for up to ω attributes; the complexity of the explanation can thus be quantified by considering the size of this attribute set. From the attribute-specific explanation, an approximation of the output scores $\hat{p}$ will be made.

In our approach, we do not actually seek an estimate for single instantiation of g; instead, we seek to compare explainability for a explainer class G Fidelity $\mathscr{L}$ of a G depends on multiple factors: the complexity ω we allow them, and the target black-box model f, which in turn is typically instantiated by a dataset $\mathscr{X}$ which is domain-dependent. We could test the fidelity of the G for a single target f, but that doesn't tell us anything beyond that one target model. If we want to estimate the *expected* fidelity of an explainer class G, we need a sample of such black box models f on a sample of datasets $\mathscr{X}$, to derive the explainers g of a certain complexity ω. This leads to the experimental design, for instance, such as the crossed-design; Every explainer class G with every model class F with every dataset $\mathscr{X}$s, which minimizes confounding.

To analyze such data, hierarchical linear models [14] will be suitable, as they will allow us to model fixed effects for the variables of interest (e.g. explainers and complexity), as well as random effects for the variables that moderate the fidelity of the explainers but should be considered as arbitrary, random instances from a larger population of interest determined by the application domain (e.g. class of target black-box models and datasets). Such models may be easily extended as well to incorporate other factors of interest specific to the domain of application, such as the mainstreamness of a user in recommendation, the query type in information retrieval, the genre of a music track in a music description setting, the type of picture in image classification, etc. By incorporating such domain-specific variables, we are able to analyze fidelity while controlling for factors that can potentially produce confounding in our results and make us attribute general performance to the explainers when in reality that performance depends on variables other than just its complexity and the target model. A sample instance of such hierarchical linear model can be as follows:

$$
\begin{aligned}
y_i = \beta_{0d[i]e[i]s[i]} + \beta_{1e[i]}\omega_i + \beta_{2d[i]s[i]}e_i + \beta_{3d[i]s[i]}m_i + \epsilon_i \\
\beta_{0des} &= \beta_0 + v_{0ds} + v_{0des} \\
\beta_{2ds} &= \beta_2 + v_{2ds} \\
\beta_{3ds} &= v_{3ds}
\end{aligned}
\tag{5.2}
$$

where y_i denotes the observed fidelity for the i-th observation of the experiment, e_i is the overall effect of the class of explainers employed in that observation, ω_i is the effect of complexity on the explainer, and m_i is a sample domain specific variable to control for. The intercept of the analysis model is defined as a fixed global intercept β_0 that models the expected fidelity of an arbitrary explainer for an arbitrary target model, from which we allow random deviations due to the specific target model (i.e. $v_{0ds} \sim \mathcal{N}(0,\sigma_0^2)$), and due to its interaction with the explainer (i.e. $v_{0des} \sim \mathcal{N}(0,\sigma_0^{'2})$).

The effect of the class of explainers is modeled with coefficient β_{2ds}, which consists in the overall fixed effect of the explainer (i.e. β_2), and a random deviation once again specific of the target model (i.e. $v_{2ds} \sim \mathcal{N}(0,\sigma_2^2)$). The effect of the explainer complexity is modeled with coefficient β_{1e}, which represents a different slope for each class of explainers, thus accounting for their interaction effect (i.e. the effect of complexity is more or less pronounced across explainers). Finally, the domain specific effect of m_i is modeled as a random effect specific of the target model (i.e. $v_{3ds} \sim \mathcal{N}(0, \sigma_3^2)$). Throughout the model, note that random effects sub-scripted with ds denote a different effect for each combination of dataset and class of target models, reflecting that a particular target model is instantiated from a training dataset and does not exists in its own.

By comparing the β_2 coefficients, we can make inferences about the general performance of classes of explainers, as well as of the effect of their complexity through the β_{1e} coefficients. Such comparisons are sound because they account for the random but irrelevant variations due to target models, datasets and domain-specific factors. Nonetheless, specific cases may still be analyzed when looking at the v coefficients as well (e.g. a target model being specially easy to explain, that is, a high v_{0ds} value).

Subsequently, the data should be collected to fit the model and infer the effect of each variable of interest. Significant effects with respect to the explainers can be examined. For instance, post-hoc methods such as the estimated marginal means (EMM) allow us to estimate main effects controlling for other variables. For the specific model suggested by (5.2), effects of the explainers, the complexity and the interaction of those two factors can be examined by EMM. Additionally, the random effect v_{3ds} can be examined to assess the potential effect of the domain-specific feature if the effect shows the substantial variability.

Another benefit of the suggested framework is its generality. For instance, under binary classification such as the spam detection, the decision is given by the binary indicator $p \in \{0, 1\}$. Once the explanation is given with the certain complexity, approximation can be made based on the explanation. It may be a score or the probability per each of the decision. The fidelity then can be computed by the area under curve of the reciever operating characteristic (AUC-ROC) against the original binary decision. On the other hand, in a multi-class classification scheme, posterior distribution of the classes can be explained by the explainer with respect to the input variables. Once the approximation of the target distribution is made based on the explanation, fidelity can be computed by the KL-divergence. Or, one can also adopt the framework where the output is given as the ranked list of elements such as the retrieval or the recommendation. Once the ranked list is reconstructed from the explanation, the fidelity can be measured by the rank correlation.

5.3. Use case: Recommender Systems

In the remainder of this paper, we apply our proposed evaluation framework to the RS domain, considering various typical recommender models, a range of explainers, and multiple recommender datasets.

5.3.1. Recommenders

Explanations are ultimately intended for humans, who should be able to understand them with limited resources. As discussed in Section 5.2 , this can e.g. be done by employing a discrete vocabulary of humanly understandable attributes for each data point of interest. In RS scenarios, tags associated with items to be recommended are a natural choice for this: they form a discrete and limited vocabulary, while still describing the items in a rich way. In terms of digital representation, each RS item can be represented as an N-hot vector, encoding what tags are relevant to the particular item.

While in this work, we seek to evaluate the performance of explainers, we have to be aware that their performance will be intimately tied to the performance of the black-box models they operated on, which in the current case will be recommender systems. Thus, it will not suffice to consider explainers on a single RS. Instead, we choose to study a number of representative recommenders, whose characteristics are distinctive from each other, as discussed below. For simplicity, in the remainder of this text, we will use “model” and “recommender system” interchangeably.

Item Neighborhood (iNN)

Neighborhood-based methods have been employed in RS for a long time, and can be considered as linear models, which are explainable by design [15, 16]. For instance, for a given set of interaction records between a set of users $\mathscr{U}$ and a set of items $\mathscr{I}$, item-based neighborhood models can be expressed as [16]:

$$\mathbf{P} = \mathbf{R}\mathbf{S}^{\mathscr{I}} \tag{5.3}$$

where $\mathbf{P} \in \mathbb{R}^{|\mathscr{U}|\times|\mathscr{I}|}$ is the score matrix to determine which item should be recommended to the user. $\mathbf{R} \in \mathbb{R}^{|\mathscr{U}|\times|\mathscr{I}|}$ is the sparse interaction matrix between users and items, where the scores either can be explicit user feedback of users (e.g. ratings) or implicit feedback (e.g. interaction counts). Finally, the $\mathbf{S}^{\mathscr{I}} \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{I}|}$ denotes the similarity matrix between items. Ultimately, the recommendation for user u is given by the list of items that is sorted by the scores $\mathbf{P}_u$ in descending order.

A common approach to construct the similarity matrix is using $\mathbf{R}$, applying a distance function between two slices of vectors indicating which users interacted with the target item. In this study, however, we employ the information of tags associated to the item to build a linear tag-aware recommender. From a given set of tags $\mathscr{T}$ associated with the items, one can then construct $\mathbf{S}$ as follows:

$$\mathbf{S}^{\mathscr{I}}_{i,j} = \frac{\mathbf{T}^{\mathscr{I}}_i \cdot \mathbf{T}^{\mathscr{I}\top}_j}{||\mathbf{T}^{\mathscr{I}}_i|| \cdot ||\mathbf{T}^{\mathscr{I}}_j||} \tag{5.4}$$

where $\mathbf{T}^{\mathscr{I}} \in \mathbb{R}^{|\mathscr{I}|\times|\mathscr{T}|}$ is the sparse annotation matrix of tags to the items. Each element indicates the degree of relevance of tags to each item. And $\mathbf{T}^{\mathscr{I}}_i$ indicates the vector corresponding to item i. By exploiting the high sparsity, alternatively, one can also compute the similarity efficiently through the low-rank approximation of $\mathbf{T}^{\mathscr{I}}$ as follows:

$$\mathbf{S}^{\mathscr{I}} \approx \hat{\mathbf{U}}\hat{\mathbf{U}}^\top \tag{5.5}$$

$$\mathbf{T}^{\mathscr{I}} = \mathbf{U}\Sigma\mathbf{V}^\top \tag{5.6}$$

where $\mathbf{U} \in \mathbb{R}^{|\mathscr{I}|\times h}$, $\mathbf{V} \in \mathbb{R}^{|\mathscr{T}|\times h}$ are the orthonormal singular vectors that are truncated at the dimensionality of h, and the $\Sigma \in \mathbb{R}^{h\times h}$ is a square matrix whose diagonal entries are the singular values of $\mathbf{T}$. $\hat{\mathbf{U}}$ denotes the truncated singular vectors $\mathbf{U}$ that is normalized by the $L2$ norm of each row. By substituting $\mathbf{S}^{\mathscr{I}}$ in (5.3) by (5.5), (5.3) can be reformulated as the form of an alternative matrix factorization as follows:

$$\mathbf{P} \approx \mathbf{Q}\hat{\mathbf{U}}^\top \tag{5.7}$$

where $\mathbf{Q} = \mathbf{R}\hat{\mathbf{U}} \in \mathbb{R}^{|\mathcal{U}|\times h}$ can be seen as the user factors while $\hat{\mathbf{U}}$ can be treated as the item factors.

Given its linear relationships, another benefit of the tag-aware neighborhood model is that it can operate on newly introduced items. For instance, one can compute the score of a new item l with tags $\mathbf{T}_l$ as follows:

$$\mathbf{P}_{u,l} \approx \mathbf{Q}_u \hat{\mathbf{U}}_l \tag{5.8}$$

where $\hat{\mathbf{U}}_l = \mathbf{T}_l \mathbf{V}\Sigma \in \mathbb{R}^h$ is the inferred item factor from the tag input $\mathbf{T}_l$, and $\mathbf{P}_{u,l}$ is the estimated preference score for user u to the new item l.

USER NEIGHBORHOOD (uNN)

In a similar manner to the item-based neighborhood model, we can also build a user-based neighborhood model:

$$\mathbf{P} = \mathbf{S}^{\mathcal{U}} \mathbf{R} \tag{5.9}$$

where $\mathbf{S}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ is the user similarity matrix, which can be derived similarly to (5.4) by substituting the item-tag matrix $\mathbf{T}$ with a user-tag profile $\mathbf{T}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{T}|}$. In this study, due to the lack of an explicit user-tag profile, we derive it by using the interaction matrix $\mathbf{R}$ and the item-tag matrix $\mathbf{T}$:

$$\mathbf{T}^{\mathcal{U}} = \mathbf{R}\mathbf{T}^{\mathcal{I}} \tag{5.10}$$

Following the same procedure of (5.5) and (5.6), ultimately we can obtain another form of matrix factorization that exploits the user-tag profile:

$$\mathbf{P} \approx \hat{\mathbf{U}}^{\mathcal{U}} \mathbf{Y} \tag{5.11}$$

where $\hat{\mathbf{U}}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times h}$ are the truncated left singular vectors decomposed from $\mathbf{T}^{\mathcal{U}}$ as (5.6) and used as the user factors, and $\mathbf{Y} = \mathbf{U}^{\mathcal{U}\top}\mathbf{R} \in \mathbb{R}^{h\times|\mathcal{I}|}$ are the item factors.

FACTORIZATION MACHINE (FM)

Introduced in [17], Factorization Machines are not only a generalization of the matrix factorization, but also one of the flexible models that allow one to employ item or user attributes to the model. In this study, a second-order factorization machine is used. For the objective function we employ the negative sampling loss [18]. Further, item tags are considered as a third set of entities, along with the users and items.

NEURAL COLLABORATIVE FILTERING (NCF)

Neural network models can exploit complex and non-linear relationships between entities, and thus have become an increasingly popular choice when seeking to optimize for accuracy. In this work, we employ a model similar to the Neural Collaborative Filtering approach proposed in [13]. To incorporate the tag information, we model the item factors of the model as a linear combination of the raw item factors $\mathbf{z}_i \in \mathbb{R}^h$ and the tag factors $\mathbf{z}_t \in \mathbb{R}^h$ that are associated with the item i as follows:

$$\mathbf{z}_i^{\mathcal{T}} = \mathbf{z}_i + \frac{1}{|\mathcal{T}_i|} \sum_{t\in\mathcal{T}_i} \mathbf{z}_t \tag{5.12}$$

where $\mathcal{T}_i$ denotes the set of tags associated with the item i. Finally, we choose to use the same objective function as the FM.

Measurement

To assess how the recommenders perform, we will measure their *accuracy* and *diversity*, as they represent two important aspects of good recommendations. In particular, we compute the Normalized Discounted Cumulative Gain (nDCG) as the main measure for recommendation accuracy. As for diversity, we compute the entropy based diversity [19]:

$$eDiv_p = -\sum_{j=1}^{p} q_j \log(q_j) \tag{5.13}$$

where q_j denotes the number of users who have the item j, normalized by the number of total top-p items delivered to the subset of users. Since the range of the measure is determined by p and the number of subset of users, ultimately one can have the normalized diversity measure:

$$nDiv_p = eDiv_p / \log(p|\mathscr{U}'|) \tag{5.14}$$

where $\mathscr{U}' \subseteq \mathscr{U}$ may the subset or the entire set of users within the dataset.

Experimental Setup

For each non-baseline recommender, we instantiated 3 models whose embedding dimensionality h is $\{32, 64, 128\}$, respectively. For learning the *FM* and *NCF* models, we use 10 negative samples to minimize the objective function and Adam solver [20] is in particular used. To further accelerate the learning, we pre-trained the user factors and the tag factors using Weighted Regularized Matrix Factorization [21] (WRMF), and reuse them for *FM* and *NCF* training. Finally, we set the hidden layer dimensionalities of the *NCF* models to 1024.

5.3.2. Explainers

In our work, we will consider a set of model-agnostic explainers, as explained below. For each of the explainers, we consider explanations to be represented as a vector $\theta \in \mathbb{R}^{|\mathscr{T}|}$, indicating the positive or negative influence of a corresponding item attribute $\mathbf{t} \in \mathscr{T}$ for the instance of the prediction. This assumes that the user u and the item i are fixed, such that we can only observe the relative influence of attribute inputs. Consequentially, an explainer also is able to infer the approximated score for the ranking items as $f(u, i, \mathbf{t}) \sim g(\mathbf{t}, \theta) = \theta\mathbf{t}^{\intercal}$. Specifically, we set Ω such that only top-ω attributes have non-zero coefficients following [4]. In case we cannot force this constraint while the explanation vector θ is derived, we simply keep the top-ω elements, and mask rest of the elements to zero after the fully dense explanation θ is obtained.

Random

Serving as a baseline, explanations from this method are formed by values drawn from the normal distribution:

$$\theta_t^r \sim \mathscr{N}(0, 1) \tag{5.15}$$

PARTIAL DEPENDENCE PLOT

The partial dependence plot (PDP) is useful for illustrating the marginal effect of one or two features against the output of the model [22, 23]. It is a global explanation method, where dependencies from the feature dimension to the output are identical for predictions from different samples. Under the context of this study, such globality holds at the user level. Formally, it is defined as follows:

$$\theta_t^{pdp} = \mathbb{E}_{t'}[f(u,i,t,t')] \tag{5.16}$$

where f is the recommender to be explained, and input u, i, t correspond to the target user, item, and tag, respectively. The score is the expectation over all the output of f given its inputs, plus the additional tags t' that are drawn from the powerset $P(\mathscr{T} \setminus t)$, which is the set of all the subsets of tag $\mathscr{T}$, excluding the target tag t. By the marginalization, θ_t^{pdp} only depends on the given user, item, and the target tag t. However, the size of powerset $|P(\mathscr{T} \setminus t)|$ exponentially increases with the number of unique t. An efficient estimation of function θ_t^{pdp} can be obtained through a Monte-Carlo approach, using the samples from the dataset:

$$\theta_t^{pdp} \approx \frac{1}{|T_u|} \sum_{t' \in T_u} f(u,i,t,t') \tag{5.17}$$

where $\mathscr{I}_u$ is the set of items from the histories of user u, T_i is the N-hot tag vector of item i. $f_{u,i}$ refers to the recommender, operating for a specific user u and item i. Note that the explanation is obtained by the marginalization over the items that are interacted with each u. It eventually makes the explanation as the user-level explanation.

SALIENCY

If the target function is differentiable, one can approximate the linear dependency between the input variables and the score using the partial derivative [8, 24]:

$$\theta_t^s = \frac{\partial f(u,i,t)}{\partial t} \tag{5.18}$$

It estimates the input-output relationship between the target tag t and the output score f by computing the first-order Taylor expansion [8], yielding local explanations.

LIME

LIME [4] is a model-agnostic approach that employs an interpretable surrogate model to explain a complex target function. Specifically, the local surrogate model g learns the local behaviors of a given model $f(x)$ from the near neighbors of a given sample x, by perturbating the variables of inputs within a certain proximity boundary π_x, while not only maximizing the fidelity $\mathscr{L}$ but also minimizing the complexity Ω. In the context of our study, this can be defined as follows:

$$g = arg\,min_{g \in \mathscr{G}} \mathscr{L}(f_{u,i}, t, \pi_t) + \Omega(g) \tag{5.19}$$

where $\mathscr{G}$ is the set of interpretable models, $\mathscr{L}$ is the loss function measuring the fidelity of the model g to the target model f, and Ω measures the complexity of the explainer g. Any scalar function measures the 'faithfulness' of g to f can be used for $\mathscr{L}$, such as weighted mean squared error:

$$\mathscr{L}(f_{u,i}, \mathbf{t}, \pi_t) = \frac{1}{M} \sum_{\mathbf{t}' \in \mathcal{Z}_i} \pi_{\mathbf{t}}(\mathbf{t}')(f(u,i,\mathbf{t}) - g(\mathbf{t}'))^2 \tag{5.20}$$

where $\mathbf{t}, \mathbf{t}' \in \mathbb{R}^{|\mathscr{T}|}$ denote the N-hot vector representations of the tags that are associated with the item i and their random perturbation, respectively. $\mathcal{Z}_i$ indicates the set of M perturbed samples $\mathbf{t}'$. $\pi_{\mathbf{t}}(\mathbf{t}')$ is the kernel that indicates the degree of proximity between the perturbed sample and the original sample, which weighs the loss for each $\mathbf{t}'$. Ultimately, if g is chosen to be a linear model, such that $g(\mathbf{t}) = \theta^\intercal \mathbf{t}$, one can use the learned vector $\theta \in \mathbb{R}^{|\mathscr{T}|}$ as an explanation indicating the impact of each tag on f.

Measurement

For ranking prediction problems, fidelity should measure the similarity between the list of items produced by the original recommender and the list of items approximated by the explainer. One intuitive way to measure this similarity is by computing a rank correlation coefficient such as Kendall's τ or Yilmaz' τ_{ap} [25]. We chose the latter because it is top-heavy, that is, differences towards the top of the list are penalized more than differences towards the bottom. In particular, we use the coefficient $\tau_{ap,a}$ [26] because it handles tied items:

$$\mathscr{L}_\tau = \tau_{ap,a}(\mathbf{f}, \mathbf{g}) \tag{5.21}$$

where $\mathbf{f}, \mathbf{g} \in \mathbb{R}^n$ are the vectors of the function values for n items from the recommender f with a specific user u and the explainer g that targets f, respectively. The need to handle ties is due to the sparsity of the explanation vector. Another advantage of $\tau_{ap,a}$ is that it is asymmetric, whereby, in our case, $\mathbf{f}$ is considered as a reference ranking and $\mathbf{g}$ as an approximation. As such, $\tau_{ap,a}$ can be interpreted as a measure of the accuracy of $\mathbf{g}$ with respect to $\mathbf{f}$ [26], which is precisely what we look for.

5.3.3. Datasets

Each recommender model will be instantiated in a data-driven way. Thus, recommenders may show different behaviors for different types of consumption datasets. In our current experiment, we therefore consider multiple datasets, including both public and proprietary recommendation data, as summarized in Table 5.1 and further discussed below. All these datasets contain the interaction count or ratings of each user for the items they interacted with, along with tag annotations for each item. Considering the diverse noisy characteristics for each dataset, we employed filtering for both user-item interactions and the tags, tailored to the characteristics of each particular dataset. Details about the nature of this filtering process are presented in the Reproducibility section.

For our analysis, to investigate the underlying potential association between user characteristics and explainability, we will also consider a user-related feature: the mainstreaminess [27] of each user. More specifically, the mainstreaminess measures the distance between the global popularity of the songs and the individual preference distribution over the songs [27].[1]

[1] As outlined in our Reproducibility section, further user features were initially investigated, but all of these

Table 5.1: Detailed properties of selected dataset.

	#Users	#Items	#Tags	#Nonzero	Density
Goodbooks	53k	10k	34k	6m	1.12%
LFM1b	107k	466k	1.3k	56.5m	0.11%
MSD	500k	90k	8.6k	28m	0.06%
Internal	332k	134k	27k	30m	0.07%

GOODBOOKS

Goodbooks-10k [28] is a book recommendation dataset. It includes the 10,000 most popular books and 5-point ratings for those books from 53k users, and also includes a rich set of user-generated tags related to the books. We framed the ratings as implicit feedback from users, to align this data with the further datasets considered.

MILLION SONG DATASET

The Million Song Dataset (MSD) [29] provides both the music listening history of users and the tags per songs. The listening history has been collected until 2011 in the former *Echonest* service, where the listening counts of users were accumulated over time. The tag data is collected by cross-matching the songs with information from the *LastFM* platform, in which a large number of users voluntarily annotated songs with free-form tags, thus introducing substantial noise in the dataset. Hence, we sub-sampled the dataset to about to the half of the original scale, by filtering less active items and users.

LFM1B

LFM1b is another music listening dataset, collected from the *LastFM* platform [30]. Unlike the MSD, the listening interactions are provided as individual events, which are accumulated to one billion records. However, to employ the dataset under the same experimental setup, we marginalize the individual listening records such that the format of the dataset becomes identical to the MSD.

Another difference between the MSD and LFM1b is the source of tag annotation, which is derived from acknowledged knowledge bases, such as *freebase*[2] [31], thus having less vocabulary noise than the MSD. At the same time, the dataset provides tags at the artist level, rather than the song level. For our work, we therefore consider each song of a given artist to be annotated with all tags applying to the artist.

INTERNAL

Finally, we also consider proprietary music listening behavior data, which we denote as *Internal* for the rest of the work. This data is collected from an active on-line music streaming service, containing the entire set of user-item transactions within 8 days during April 2019. While this is a relatively short collection period compared to other datasets, the descriptive statistics show that in terms of size, the dataset compares to the other datasets,

turned out heavily cross-correlated. Given space constraints, we therefore chose to only report on mainstreaminess in this paper.

[2] `https://developers.google.com/freebase`

indicating a potential discrepancy between real-world data volumes and traditional research datasets. As an extra interesting aspect, unlike the other datasets, which were more Western-focused, the *Internal* dataset considers a streaming service strongly tailored to an Asian domestic market, thus being likely to show different consumption trends particular to this market. Tags in the music service are offered at the level of playlists, and made by certified curators, who mostly perform the annotations in their domestic, non-Western language. To move from playlist-level tags to song-level tags, for each song, we take the union of music tags that were associated with the playlists containing this song. Consequentially, songs that are not included in any playlists are excluded from our study.

5.4. Results

5.4.1. Recommendation

We first measure the recommendation performance as explained in 5.3.1, and compare it to several baselines.[3] This gives us a general overview of the model performance on this use case, allowing us to detect irregular or outlier model behavior that would affect the study of explainability. The baselines we compare to are:

- *WRMF* (Weighted Regularized Matrix Factorization) [21], developed for implicit feedback such as the music listening count. We tested the algorithm with the same latent size h as in the aforementioned models.
- *Most Popular* method recommends the most frequently interacted items within the dataset.
- *Random* method recommends items at random from the set of items not yet consummed by the target user.

As Figure 5.2 indicates, nDCG scores vary widely across datasets and recommendation models. The best results are observed on the *Goodbooks* dataset (0.3173 ± 0.13), followed by the *Internal* dataset (0.2126±0.10), *MSD* (0.1248±0.07) and *LFM1b* (0.1133±0.06) dataset. On the other hand, diversity performance follows the opposite trend. Within models, *WRMF* is indicated as the most accurate model in most of the cases. However, it is not as diverse as the others, except the *Most Popular* algorithm and the *uNN*. *NCF* follows *WRMF* in terms of accuracy, while showing slight improvement on diversity. *FM*, which follows *NCF* in accuracy by small margin, shows also a small improvement in diversity. Unlike to the non-linear competitors, neighborhood-based models appear to be substantially less accurate. In particular, we see that *iNN* is the least accurate model besides the *Random* recommendation, while it achieves one of the highest diversity scores.

5.4.2. Explainability

We now consider how the various model-agnostic explanation methods perform on multiple datasets and recommender systems. Figure 5.3 and Figure 5.4 illustrate the explainability performance as a function of complexity and user mainstreaminess, respectively.

[3] For consistency, we uniformly sampled 1000 users from each dataset and used only those for the purposes of analysis. This is because it was infeasible for us to compute fidelity scores for all users of all datasets and all recommenders due to the high computational cost of the explainer implementations [4].

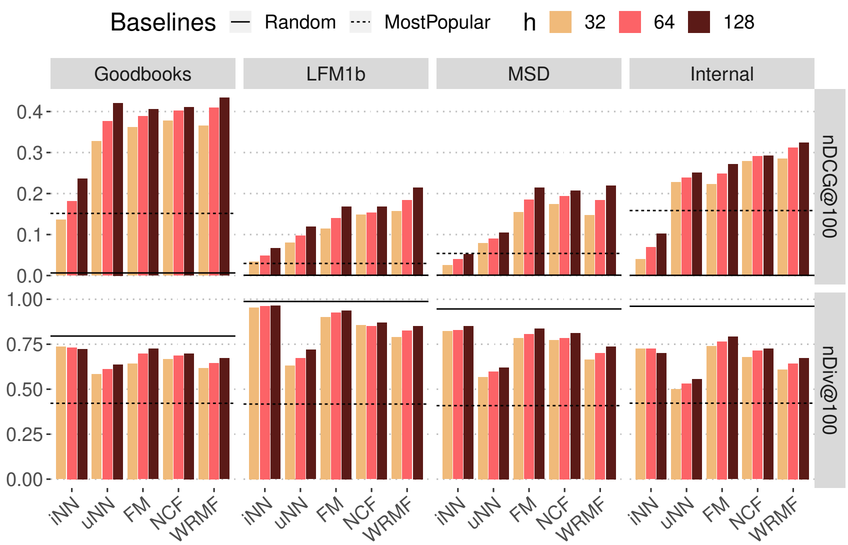


Figure 5.2: Performance of the recommenders.

Each point indicates the $\tau_{ap,a}$ score achieved by the explainer for one user. The lines show a linear fit for each explainer. Such visualization effectively illustrates the overall distributions, which can reveal the potential interactions among the various factors contemplated in our framework.

In general, *LIME* shows the best fidelity, followed by the *Saliency* and *PDP* explainers. As expected, the *Random* explainer achieves a correlation of zero. Among recommender systems, *iNN* and *uNN* achieve better overall fidelity than *FM* and *NCF*. It shows that the linearity of the neighborhood based models generally makes them more easily interpretable.

For all conditions, it is shown that ω has a positive effect on explainability. The exception is of course the *Random* explainer, which is not affected by ω. On the other hand, Figure 5.4 shows that mainstreaminess has a less consistent trend compared to ω. Specifically, we see a larger effect for the non-linear recommender systems.

Employing the analysis model from Equation (5.2), we considered the explainers e and the number of tags used for the explanation ω as the main fixed effects, including the interaction between them. As we observe sub-linearity from Figure 5.3, we will consider the natural logarithm of ω. We set the interaction of the dataset d and the RS s as the group for the random effect, such that the model can explain the heterogeneity within groups. Finally, we consider the mainstreaminess m to have the random effect. We compute model fitness by the estimated R^2, using the method introduced in [32]. From this, it is shown that the fixed effects explain the data reasonably well ($R^2 = 0.6023$), while the entire model including both fixed and random effects explains the most of the variance of the data ($R^2 = 0.9363$).

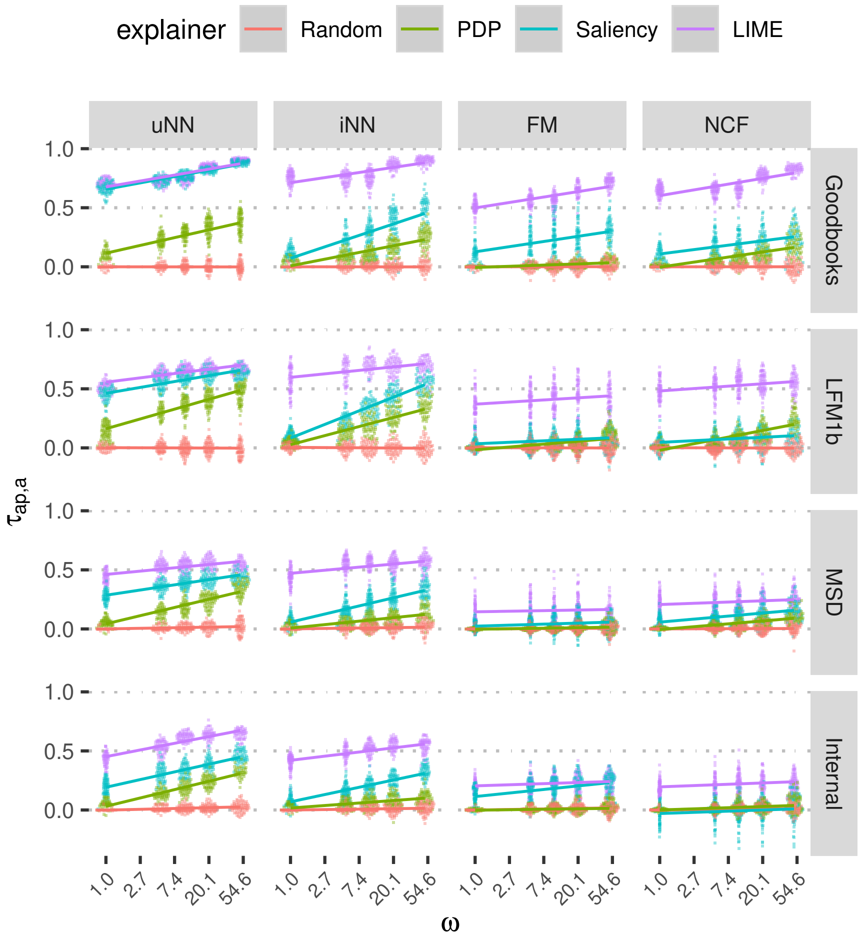


Figure 5.3: Overview of the explainability measurement. Each point represents the $\tau_{ap,a}$ measurement of each user. Each row of panes is dedicated for each dataset, while columns are assigned per recommenders. Colors are mapped for each explainer. The distinction between different model size h is omitted, not to make the visualization uncluttered. The horizontal axis is plotted in the logarithmic scale.

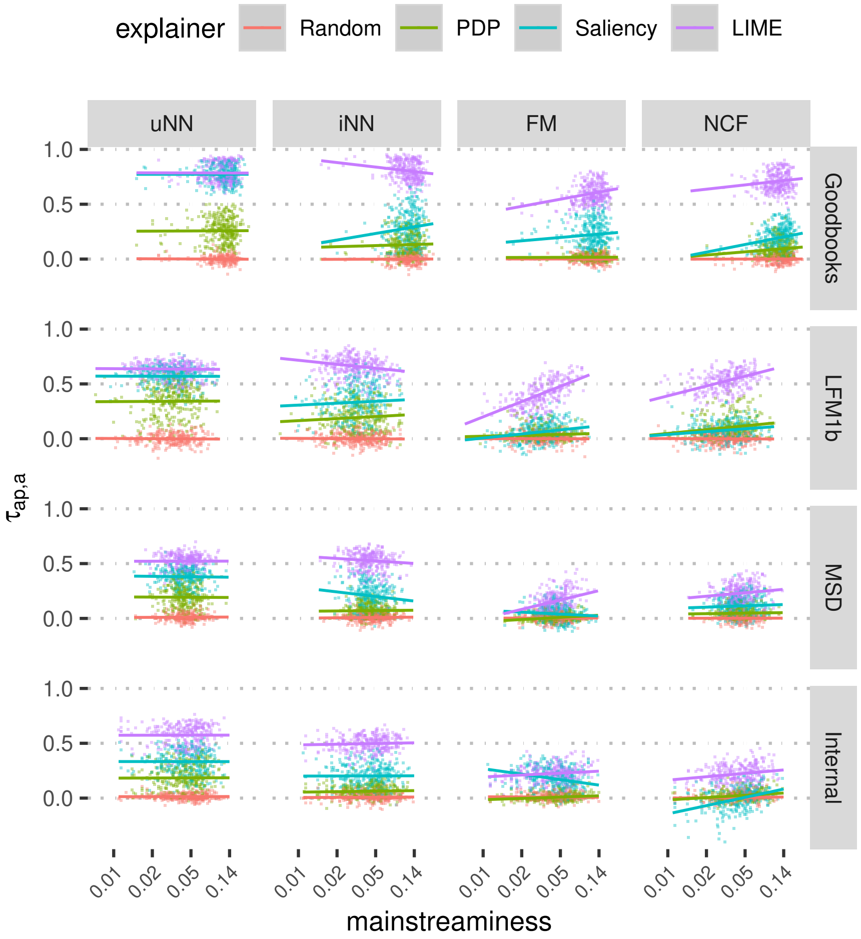


Figure 5.4: Overview of the explainability measurement over the mainstreaminess. The horizontal axis is plotted in the logarithmic scale.

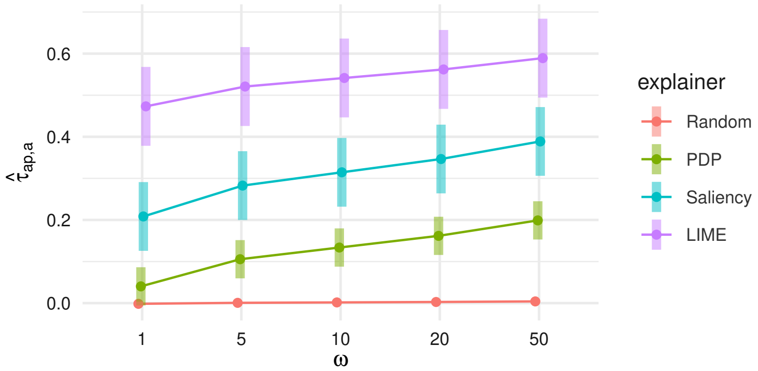


Figure 5.5: The estimated marginal mean of the predicted $\hat{\tau}_{ap,a}$ with respect to ω and the explainers. The error bars in the figure indicates the 95% confidence intervals.

Figure 5.5 illustrates the fixed effects for the ω and the explainers from the analysis model. It indicates that as expected, the number of the tags has an overall positive effect on the explainability. At the same time, the large confidence interval suggests that other factors, such as the choice of the recommender or explainer, further affect the explainability. Considering the explainers, again, *LIME* clearly stands out in comparison to other explanation model candidates, with *Saliency* consistently following up. However, considering the confidence interval obtained for $\omega = 1$, *PDP* cannot be claimed to significantly outperform the *Random* explainer.

Inspecting the variance components in the fitted model, we find that mainstreaminess shows a substantially larger variance component (0.2569) than other factors, such as the employment of *LIME* (0.0439), which is the second largest variance component among all random factors. This is followed by other factors such as *Saliency* explainer (0.0434), *PDP* explainer (0.0101), and the intercept (0.0006).

5.4.3. Discussion

Non-Linearity

Our results suggest that there is notable difference between the neighborhood recommenders and the other models. As expected, while these linear recommenders are inferior to their non-linear competitors in terms of accuracy (as also shown in Figure 5.2), our evaluation results confirm that they arguably are more interpretable. Thus, there indeed is a trade-off between explainability and accuracy, and systems may need to optimize for both criteria.

Sparsity

As our results show, having a sparsity constraint during the inference of the explanation (which is a key feature of *LIME*) has substantial effect on the explainability. It will naturally force its explanation vector θ to have a small ω during the inference, where all other methods initially yield a dense explanation vector that has to be truncated post-hoc.

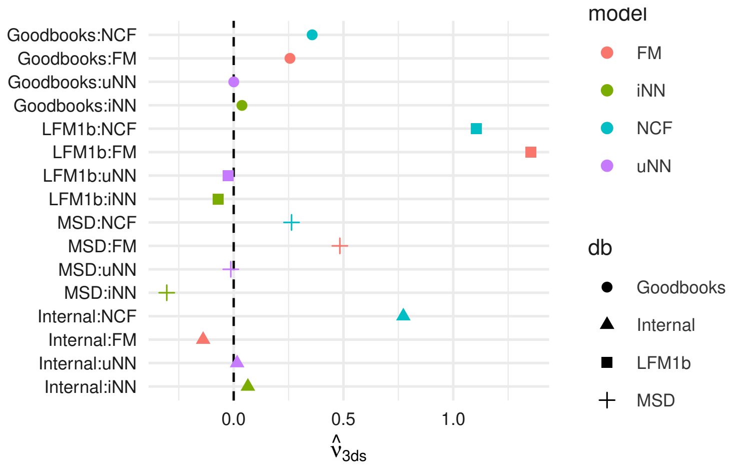


Figure 5.6: Estimated random effect $\hat{v}_{3ds}$ for the the mainstreaminess variable. Labels on the vertical axis indicates the dataset-recommender combinations of interest.[4]

USER MODELING

As discussed before, applying our analysis model, we find that the mainstreaminess feature explains a much larger amount of variation than the other random factors considered. Thus, as a further exploratory study, we examined the estimated individual random effect of mainstreaminess.

Figure 5.6 illustrates this estimated random effect. Overall, user mainstreaminess has a positive effect when *FM* or *NCF* recommenders are used, except for the case in whiche *FM* is used on the LFM1b dataset. Another general trend is that the effects are closer to 0 or negative where linear neighborhood models are used. This indicates that the more complex RS models pick up mainstreaminess information more strongly than linear models.

5.5. CONCLUSION & FUTURE WORK

In this paper, we proposed an evaluation framework for assessing various model-agnostic explainers, and applied the framework to the context of recommender systems. As the results of our experiments show, our framework can reveal underlying differences between explainers, despite the heteroskedasticity introduced by underlying factors such as the type of recommender or the dataset.

Our results indicate that the local surrogate model with forced sparsity [4] leads to significantly better explainability.As expected, we find that complex non-linear recommender models generally will be more accurate, but are more difficult to explain. Furthermore, we find that user factors such as the mainstreaminess [27] may affect the explainability.

In future work, several open challenges can still be addressed. First of all, the explicit analysis model we chose to use in Equation 5.2 is an instantiation of our framework in

[4] For instance, "MSD:NCF" refers the the case where a *NCF* recommender is learned from the MSD.

itself, and can still be further refined and improved to account for further potential factors and interactions.

Furthermore, in existing work on model-agnostic explanation methods, explanations were based on input features directly. Under the RS scenario, suitable input features can have many different forms (e.g. item identity, signal properties, discrete attributes associated to the items). In the current work, we considered discrete attributes, but our approach can be generalized to also consider continuous variables as attributes.

As one limitation of our current approach, we assumed that each possible attribute value (i.e. each possible tag) can be considered to have an equal amount of information and an equal level of complexity. This may not be realistic in practice, and further research is needed to assess whether the explainability measurement will be biased in undesired ways by this assumption.

Ultimately, a human user will always have the final word on the best explanation [9]. In our current work, we deliberately chose not to focus on user-facing studies yet; this will require further careful consideration in how explanations will be presented, both in terms of content and in terms of user experience. Instead, with the current work, we aimed to offer a tool to perform offline sub-component evaluations as an intermediate step towards final user studies. As our framework can help to systematically and objectively inform what explanations are sufficiently effective, such explanations can be prioritized in the further work necessary to conduct user studies and assess end-user satisfaction.

REFERENCES

[1] N. Tintarev and J. Masthoff, *A survey of explanations in recommender systems,* in *Proceedings of the 23rd International Conference on Data Engineering Workshops, ICDE 2007, 15-20 April 2007, Istanbul, Turkey* (IEEE Computer Society, 2007) pp. 801–810.

[2] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, *A survey of methods for explaining black box models,* ACM Comput. Surv. **51**, 93:1 (2019).

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, *Model-agnostic interpretability of machine learning,* CoRR **abs/1606.05386** (2016), arXiv:1606.05386 .

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, *"why should I trust you?": Explaining the predictions of any classifier,* in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, edited by B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi (ACM, 2016) pp. 1135–1144.

[5] M. Verma and D. Ganguly, *LIRME: locally interpretable ranking model explanation,* in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019* (2019) pp. 1281–1284.

[6] J. Singh and A. Anand, *Posthoc interpretability of learning to rank models using secondary training data,* CoRR **abs/1806.11330** (2018), arXiv:1806.11330 .

[7] G. Peake and J. Wang, *Explanation mining: Post hoc interpretability of latent factor models for recommendation systems,* in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, edited by Y. Guo and F. Farooq (ACM, 2018) pp. 2060–2069.

[8] K. Simonyan, A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps,* in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings,* edited by Y. Bengio and Y. LeCun (2014).

[9] H. Kaur, H. Nori, S. Jenkins, R. Caruana, H. M. Wallach, and J. W. Vaughan, *Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning,* in *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*, edited by R. Bernhaupt, F. F. Mueller, D. Verweij, J. Andres, J. McGrenere, A. Cockburn, I. Avellino, A. Goguey, P. Bjøn, S. Zhao, B. P. Samson, and R. Kocielnik (ACM, 2020) pp. 1–14.

[10] H. Wang, N. Wang, and D. Yeung, *Collaborative deep learning for recommender systems,* in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, edited by L. Cao, C. Zhang, T. Joachims, G. I. Webb, D. D. Margineantu, and G. Williams (ACM, 2015) pp. 1235–1244.

[11] H. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, *Wide & deep learning for recommender systems,* in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016, Boston, MA, USA, September 15, 2016*, edited by A. Karatzoglou, B. Hidasi, D. Tikk, O. S. Shalom, H. Roitman, B. Shapira, and L. Rokach (ACM, 2016) pp. 7–10.

[12] P. Covington, J. Adams, and E. Sargin, *Deep neural networks for youtube recommendations,* in *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, edited by S. Sen, W. Geyer, J. Freyne, and P. Castells (ACM, 2016) pp. 191–198.

[13] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, *Neural collaborative filtering,* in *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, edited by R. Barrett, R. Cummings, E. Agichtein, and E. Gabrilovich (ACM, 2017) pp. 173–182.

[14] A. Gelman and J. Hill, *Data analysis using regression and multilevel/hierarchical models* (Cambridge university press, 2006).

[15] K. Verstrepen and B. Goethals, *Unifying nearest neighbors collaborative filtering,* in *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, edited by A. Kobsa, M. X. Zhou, M. Ester, and Y. Koren (ACM, 2014) pp. 177–184.

[16] S. Sedhain, A. K. Menon, S. Sanner, and D. Braziunas, *On the effectiveness of linear models for one-class collaborative filtering,* in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, edited by D. Schuurmans and M. P. Wellman (AAAI Press, 2016) pp. 229–235.

[17] S. Rendle, *Factorization machines,* in *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, edited by G. I. Webb, B. Liu, C. Zhang, D. Gunopulos, and X. Wu (IEEE Computer Society, 2010) pp. 995–1000.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality,* in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* (2013) pp. 3111–3119.

[19] G. Adomavicius and Y. Kwon, *Improving aggregate recommendation diversity using ranking-based techniques,* IEEE Trans. Knowl. Data Eng. **24**, 896 (2012).

[20] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization,* in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015).

[21] Y. Hu, Y. Koren, and C. Volinsky, *Collaborative filtering for implicit feedback datasets,* in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy* (IEEE Computer Society, 2008) pp. 263–272.

[22] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics (Springer New York Inc., New York, NY, USA, 2001).

[23] J. H. Friedman, *Greedy function approximation: a gradient boosting machine,* Annals of statistics , 1189 (2001).

[24] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K. Müller, *How to explain individual classification decisions,* J. Mach. Learn. Res. **11**, 1803 (2010).

[25] E. Yilmaz, J. A. Aslam, and S. Robertson, *A new rank correlation coefficient for information retrieval,* in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008* (2008) pp. 587–594.

[26] J. Urbano and M. Marrero, *The treatment of ties in AP correlation,* in *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR 2017, Amsterdam, The Netherlands, October 1-4, 2017*, edited by J. Kamps, E. Kanoulas, M. de Rijke, H. Fang, and E. Yilmaz (ACM, 2017) pp. 321–324.

[27] M. Schedl and C. Bauer, *An analysis of global and regional mainstreaminess for personalized music recommender systems,* J. Mobile Multimedia **14**, 95 (2018).

[28] Z. Zajac, *Goodbooks-10k: a new dataset for book recommendations,* FastML (2017).

[29] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, *The million song dataset,* in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, edited by A. Klapuri and C. Leider (University of Miami, 2011) pp. 591–596.

[30] M. Schedl, *The LFM-1b dataset for music retrieval and recommendation,* in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ICMR 2016, New York, New York, USA, June 6-9, 2016*, edited by J. R. Kender, J. R. Smith, J. Luo, S. Boll, and W. H. Hsu (ACM, 2016) pp. 103–110.

[31] M. Schedl and B. Ferwerda, *Large-scale analysis of group-specific music genre taste from collaborative tags,* in *19th IEEE International Symposium on Multimedia, ISM 2017, Taichung, Taiwan, December 11-13, 2017* (IEEE Computer Society, 2017) pp. 479–482.

[32] S. Nakagawa and H. Schielzeth, *A general and simple method for obtaining R^2 from generalized linear mixed-effects models,* Methods in Ecology and Evolution **4**, 133 (2013).

6

Conclusion

Throughout the main body of the thesis, we delivered a series of studies addressing ways to incorporate principles of trustworthy ML into ML practice. We conducted these studies in the music domain, but state confidently that the main insights acquired are generally applicable. In this concluding chapter, we reflect on the conducted research, distil the main recommendations for the community of ML researchers and practitioners, and propose the way forward.

6.1. ON RELIABLE USE OF PRE-TRAINED NEURAL NETWORKS

One of the emerging practices in modern ML is to rely on adopting the neural networks pre-trained on a (source) task, and apply them for learning a new (target) task. The underlying assumption is that one could obtain a well-performing ML system with only a handful of training data points at hand, if the pre-trained network is already well trained on a relevant source task, that used a representative dataset at scale. The main question related to the effectiveness of this transfer learning is to what extent the pre-training on the initial task is "useful" to a new task. When answering this question, however, one needs to know what "usefulness" means in a given context. Is it related to the overall accuracy gain brought by the pre-trained network, or should the robustness to noise be the primary concern over the accuracy gain? We referred to these two specific qualities as the *transferability* and *robustness*, respectively, and conducted studies that empirically assess those two qualities of the pre-trained networks on music data and under diverse task conditions.

The investigation in Chapter 2 shows that using a neural network pre-trained on a single source task is not likely to lead to high general transferability to different target tasks. In other words, we found no single source task that secures transferability to all target tasks in our experiments. We derived this conclusion based on the experiments involving network pre-training on a range of source and target tasks, and by varying the learning capacity of the network. Our experiments indicate that considering multiple source tasks helps achieve more reliable transfer learning in a general case. Therefore, if the resources allow multi-task pre-training, we advise pre-training the network using a variety of source tasks. As an alternative option, we suggest deploying multiple pre-trained networks as an ensemble, where each network is pre-trained on a different source task. Further, in Chapter 3, we reassure the effectiveness of the conclusion. We employ the best transfer learning strategy found in Chapter 3 to the independent music classification challenge and successfully achieve the best accuracy among the participants.

The investigation in Chapter 4 points out the need to carefully check the robustness of pre-trained networks to input perturbation, before deploying them for transfer learning. This is important, as even a hardly perceptible change in network input can lead to substantially irregular behavior at the output. We devised a way to measure this robustness, without the need for the full original data of the source task, and before deploying the network to learn for a target task. In essence, our method measures to what extent the internal representation of an input data point drifts away from the original position, after introducing a perturbation. Robustness would imply that the position remains close enough if the perturbation is barely perceptible. As the assessment procedure is independent of either source or target task, it only requires a few input data points and the network itself.

To our surprise, we observed that all of the considered networks violate the expectation to be robust. Even a minimum perturbation could substantially deform the relational structure between data points. We also observed that the choice of the source task can make a substantial difference in making the network robust. Semantically high-level source tasks (i.e., music auto-tagging) lead to relatively more robustness to input perturbations than lower-level tasks (i.e., auto-encoder). These insights lead to a number of recommendations. First, if small perturbations in input data are expected, networks pre-trained on semantically high-level tasks should be considered more suitable for transfer learning than the networks pre-trained on lower-level tasks. If larger perturbations are expected, a conventional feature encoder would be the preferred choice over transfer learning. Second, if it is feasible for the practitioners to pre-train the network from a source task on their own, we recommend using the experimental elements we suggested in various ways to ensure robust pre-training. For instance, the robustness criterion can be used as regularization during the training, or perturbations can be used to augment the (source) dataset. Finally, our observation that transferability and robustness of pre-trained networks should not be taken for granted, also points out the need for a more structured and comprehensive approach to transfer learning. We believe the community should become better aware of various properties of pre-trained models up front. It would be beneficial to work together to build a centralized repository of pre-trained models, in which characteristics such as robustness or general transferability are measured and registered. This would enable identification of the right networks for a given transfer learning task. We believe that our proposed testing method aligns well with such an approach, and could be used in automated testing frameworks, that can report on the robustness of a network to perturbations.

6.2. On Making the ML Systems better explainable

High learning capacity and expressiveness of modern ML models, and in particular deep neural networks, are crucial factors that differentiate them from the conventional ML models. These factors, however, go together with the fact that modern ML models are exceptionally complex, making their learning and decision-making processes hardly explainable. This may hamper wide adoption of such models as the key elements of our socio-technical infrastructure, because explainability is the key to reliability and accountability of ML-automated decision-making processes.

While the issue of explainability is gaining attention in the research community, the effort to actually improve the explainability of complex ML systems is still in its infancy. In addition to addressing reliability issues in transfer learning, the studies introduced throughout Chapter 2 and Chapter 4 were also intended to provide a contribution in this respect. As discussed, multi-task learning can be useful for partially understanding the internals of the black-box models. Furthermore, provided that one can design and conduct the pre-training stage, multiple tasks can provide multiple explanatory factors for specific parts of the internal layers. However, in many practical transfer learning scenarios, control over the design of the pre-training stage is rare. This implies that post-hoc testing would be necessary to understand, for instance, the sensitivity to irregular behavior at the input.

A more promising approach would be a more explicit one, deploying post-hoc explanation methods, which approximate a black-box model with an interpretable explainer [1]. While many model-agnostic explainers have already been suggested and widely deployed, an evaluation framework enabling the users to choose the best explainer based on objective criteria would make this approach more effective [2]. In Chapter 5, we propose such an evaluation framework, which enables one to measure the quality of the explanation provided by different post-hoc explanation methods when applied to a black-box system. The quality is evaluated by analyzing the interplay between two relevant measurements: fidelity and complexity. Fidelity refers to the extent to which the interpretable explainer model can approximate the target black-box model in terms of its input-output relations. Complexity, in contrast, measures how complicated (e.g., "long") the resulting explaination is. Intuitively, a more complicated explanation could be expected for more complicated systems. Since, however, the complexity of the explanation may increase the cognitive load at users, we suggest searching for the explainer with sufficient fidelity, but minimum possible complexity.

The proposed framework is also flexible enough to allow inclusion of domain-specific factors potentially associated with the explainability. For example, in the recommender systems context used as the testbed in Chapter 5, we rely on "user-mainstreamness" as a domain-specific factor. The underlying rationale is that mainstream users would likely consume less diverse, popular items, making it easier to explain the underlying recommendation mechanism. The population of niche users, by contrast, presumably are interested in niche, long-tail items, the recommendation of which could be based on a less explainable rationale.

6.3. THE WAY FORWARD

Machine learning solutions are increasingly embedded in a wide range of software solutions, empowering automated decision-making. The decisions made through ML-based algorithms are increasingly affecting the way we live. it therefore is imperative to develop powerful evaluation frameworks for ML models, that can assess their decision-making abilities beyond accuracy alone, in order to increase their trustworthiness.

While the research conducted for this thesis targeted music data and application domains, we emphasize the "transferability" of insights from our work to other domains and data types. For example, the use of multiple source tasks to train a network to be deployed for transfer learning can help overcome typical problems of incompatibility between source and target tasks. Since such incompatibility is likely to appear in any domain, the suggestions from Chapter 2 can be followed in other cases and domains too. The same can be said for the proposed testing framework to assess the robustness of a pre-trained network with respect to input perturbations described in Chapter 4, as well as the method proposed in Chapter 5 to assess post-hoc model explainers. All three contributions are data- and application-agnostic, and therefore applicable to any given domain with minimum adaptation.

We therefore invite researchers and practitioners from different domains to apply the insights from this thesis, to report to which extent our frameworks and methods will indeed shed light on trustworthiness issues in broader ML application domains, and to use obtained insights to help resolving these issues. We argue that ML system trustworthiness should receive more focused attention, both as an integral part of the research process, as well as in quality assurance processes for ML system development. Here, widespread community effort will be critical for success.

For example, as indicated in Chapter 2 and 4, assessing and opting for the best set of pre-trained networks for the transfer learning scenario only is possible, if a diverse range of networks is already available. It is, however, also important to structure this widespread effort, so it is maximally effective. This requires an adjustment of the focus of the publications reporting such networks. For example, the networks are often published in a research paper that typically describes the network properties and the network's learning setup. While this can be used as a "manual" for the users of those networks, it may be insufficiently informative for the readers interested in applying such networks for transfer learning. For the latter purpose, for instance, one would also expect more explicit listing of the specific data points that were used for the training, which is essential to avoid overfitting during the transfer. In other words, we plead for more elaborate standardization in assessing and reporting about a neural network model, in ways that are sufficiently informative for future considerations. Unified reporting schemes, such as the model card [3], may be a solution to this, providing examples on how to generate and document a network's description and its assessment from diverse angles (including trustworthiness principles), along with the data, task, and possible confounding factors.

Furthermore, regarding trustworthiness assessment as part of quality assurance in ML model development, we are aware of the Software Engineering community's emerging interest in Testing for Machine Learning [4]. We believe that interesting connections can be made between our current contributions and the testing techniques researched in this field, and that future work combining the fields' insights will lead to further refinement of systematic ways to improve ML trustworthiness in practice.

In addition to applying the insights from this thesis to other domains, it also is of vital importance to keep a close eye on how trustworthiness will be operationalized. As briefly discussed in the introduction, trustworthy ML principles and guidelines are not necessarily sharply or consistently defined. Therefore, we expect that in many practical scenarios, the formulation, interpretation and operationalization of these principles for use in technical frameworks will need several iterations with relevant stakeholders, in order to reach practically relevant and workable common ground.

For instance, the research reported in this thesis builds on a specific definition of robustness, which focuses on perturbations induced by semantically relevant transformations of input data. However, the robustness can also be investigated with respect to semantically irrelevant transformations that could hamper the expected model's performance [5]. Another aspect of robustness that requires attention is related to malicious data transformations to attack a ML system in an adversarial learning context. Addressing such diverse perspectives, and incorporating them into ML practice, is critical for broad and effective ML trustworthiness.

Finally, as already hinted when referring to stakeholders: trustworthiness cannot be defined or established without considering human designers and users of ML systems. Therefore, the human factor is explicitly important to ML trustworthiness too. The perception and interpretation of trustworthiness by the human designers and users of ML systems needs conscious focus, and possible revisions throughout the development process. Next to the technical frameworks and methods proposed in this thesis, this requires a sufficient breadth and depth of user studies, which will help to understand how to fine-tune and deploy these frameworks in practice, and to improve human-in-the-loop feedback mechanisms. Finally, this calls for an inter-disciplinary approach to developing ML solutions, also including insights from human-computer interaction. [6].

REFERENCES

[1] C. Molnar, *Interpretable Machine Learning* (2019).

[2] H. Kaur, H. Nori, S. Jenkins, R. Caruana, H. M. Wallach, and J. W. Vaughan, *Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning,* in *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020* (2020) pp. 1–14.

[3] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru, *Model cards for model reporting,* in *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, Atlanta, GA, USA, January 29-31, 2019* (2019) pp. 220–229.

[4] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, *Machine learning testing: Survey, landscapes and horizons,* CoRR **abs/1906.10742** (2019), arXiv:1906.10742 .

[5] B. L. Sturm, *A simple method to determine if a music information retrieval system is a "horse",* IEEE Trans. Multimedia **16**, 1636 (2014).

[6] H. Cramer and J. Kim, *Confronting the tensions where UX meets AI,* Interactions **26**, 69 (2019).

ACKNOWLEDGEMENTS

Looking back, I realize that the academic training I received has been quite a challenging journey. It was full of obstacles, and I must have dealt with them to go forward. It would have been impossible if I did not have your helping hands, guidance, and patience with my surroundings. I want to say appreciation here for those who made my incredible journey possible.

Cynthia, my journey could have not been started without you, so I want to thank you first. As my direct supervisor, you have been showing me how to science and what academics should be and can be. Beyond influencing and guiding me as a baby academic, you always show incredible amounts of patience to pick up things when I am slow or when I am away for extracurricular activities such as the internship. I remember all the long nights of writing, chasing deadlines. We also had many interesting discussions about academia, machine learning, and science over countless cups of coffee. I could have the experience of helping the ISMIR organization, which is one of the most memorable events I have ever had. I appreciate all the things I have learned from you.

Alan, you were also the one who allowed me to start my Ph.D. and guided me to finish. Thank you for all discussions and advice we have had for our works. Your insights always helped me how to effectively / efficiently communicate the research to others. I also appreciate the constructive and long editorial work you have helped with my thesis.

I want to say thanks to my mother and JC, who passed through a difficult time in the middle of my Ph.D. Being a far way, I was not the greatest member while the rest were facing a hard time. I appreciate all your understanding and care you spared for me, without which I could not achieve my degree. Concerning this challenging time of our family, I also appreciate all the MMC group members for all the kind words and care you showed to my family and me.

Julián, I thank you for what I learned from the works we have done, which are eventually all of my main chapters. The experience helped me a lot to be confident to conduct research. There remain endless things to learn, but you also showed me how to keep learning. Stjepan, it would not be possible for me to experience totally another field of study without you. Although it sometimes was difficult, it was an excellent opportunity to experience a new problem space.

I also learned a lot from my home team, Multimedia Computing Group. I have consistently earned good insights and feedback from different reading groups and discussion groups, including our weekly meeting. Elvin, Huijuan, Martha, Odette, Pablo. It might not be sufficient time that we interacted within the group. Still, it was always true that I learned directly/indirectly a lot from the incredibly diverse range of research topics of yours. I appreciate it.

Andrew, Sandy, the “Cynthia-team,” I spare this part for you to say thank you for all the discussions, both research projects and cooking projects we have done. It has always been amazing that we exchange diverse perspectives and knowledge about science and research, building up the spirit of team-science among us.

My office mates! Alessio, Babak, Ernestasia, Karthik, Raynor, Soude, Jaeyoung, Xiuxiu, Manel, Roger, Harlley, Alberto, Omar, Bence, Li. We had great moments, working/hanging out/discussing/drinking tea or (sometimes bad, sometimes good) coffee together. I regret we could not have it more. It also was one of the most important ingredients of my Ph.D. Also, Saskia, from starting to the end, you were always the best person I could rely on as far as the administrative issues concern, and I can tell you are the best without any doubt. Thank you for all your support.

Minz, it has been more than half a decade that we work together in a way or another. I want to thank all the professional/personal interactions we have had, all of them not just for fun but also helpful for my Ph.D. (one of the motivations for actually considering mine was your passion and eagerness to go for the Ph.D. position). I hope you also finish your Ph.D. soon with exceptional deliverables. Petros, my first master student and a good friend, dungeon master. I must say thank you as well, having me in your circle, hanging around. I believe you will do a great job in the end, so keep pushing a bit further to the end. Also, Norman, Jochem, Tung. You might think you are the one who picks up things from our discussion, but I also always get to know more stuff throughout it as well. Thank you for what you taught me. I hope all of you succeed in your future paths.

Also, I would like to thank to Anelale Nájera and Phoebe Strafford, who shared beautiful photographs to world through Unsplash so that I can make the cover of this thesis.

For the final remark, I would like to leave a couple of words to my father. Tragically, I cannot share with you how my journey of Ph.D. ends. But I know you would be proud of me, actually, no matter how it ends. It went marvelously well. It was not bad. So please rest in peace.

I deeply apologize for whom I happened to forget to mention. I am grateful for whichever interaction I had with you during my Ph.D., which was a mere inter-connection of the colorful dots that laid out thanks to all of you.

Curriculum Vitæ

Jaehun Kim

18-12-1986	Born in Seoul, Republic of Korea

Education

2006–2013	Undergraduate in English Literature & Linguistics Seoul National University Republic of Korea
2010–2013	Undergraduate in Information & Culture Technology Seoul National University Republic of Korea
2013–2015	Master of Science in Digital Contents and Information Studies Seoul National University Republic of Korea
2016–2021	PhD. in Computer Science Delft University of Technology The Netherlands

Experience

2014	Research Intern SK Planet Republic of Korea
2016	Research Scientist Spoqa Republic of Korea
2019	Research Intern Kakao Republic of Korea

LIST OF PUBLICATIONS

9. Ahn, H., **Kim, J.**, Kim, K., & Oh, S. (2020). Generative Autoregressive Networks for 3D Dancing Move Synthesis from Music. IEEE Robotics and Automation Letters.

8. **Kim, J.**, Urbano, J., Liem, C. C. S. & Hanjalic, A. (2020). One deep music representation to rule them all? A comparative analysis of different representation learning strategies. Neural Comput & Applic 32, 1067–1093.

7. **Kim, J.**, Demetriou, A. M., Manolios, S., & Liem, C. C. S. (2019). Beyond Explicit Reports: Comparing Data-Driven Approaches to Studying Underlying Dimensions of Music Preference. In Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization (pp. 285-293).

6. **Kim, J.**, Urbano, J., Liem, C. C. S., & Hanjalic, A. (2019). Are Nearby Neighbors Relatives?: Are Nearby Neighbors Relatives?: Testing Deep Music Embeddings. Frontiers in Applied Mathematics and Statistics, 5, 53.

5. **Kim, J.**, Picek, S., Heuser, A., Bhasin, S., & Hanjalic, A. (2019). Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems, 148-179.

4. Picek, S., Samiotis, I. P., **Kim, J.**, Heuser, A., Bhasin, S., & Legay, A. (2018). On the performance of convolutional neural networks for side-channel analysis. In International Conference on Security, Privacy, and Applied Cryptography Engineering (pp. 157-176). Springer, Cham.

3. **Kim, J.**, Won, M., Liem, C. C. S., & Hanjalic, A. (2018). Towards seed-free music playlist generation: Enhancing collaborative filtering with playlist title information. In Proceedings of the ACM Recommender Systems Challenge 2018 (pp. 1-6).

2. **Kim, J.**, Won, M., Serra, X., & Liem, C. C. S. (2018). Transfer Learning of Artist Group Factors to Musical Genre Classification. In Companion Proceedings of the The Web Conference 2018 (WWW '18). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1929–1934.

1. Kim, C. W., **Kim, J.**, Kim, K., & Won, M. (2017). Single and Multi-Column Neural Networks for Content-based Music Genre Recognition. In MediaEval.