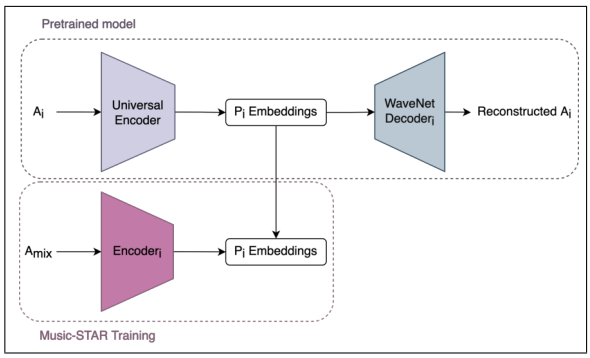

Figure 2. The embedding-supervised encoder is trained to generate the same embeddings as the universal encoder for every instrument *i* present in an audio mix (A_{mix}).

in every audio segment while training. This way, they ensure that the encoder does not memorize the content and drive it to capture meaningful features. In other words, the network is trained as a denoising autoencoder that learns to recover the original input by applying teacher forcing [22] while training the decoders. The teacher forcing technique is carried out by feeding the original input audio segment to the decoder instead of the generated samples.

3.2 Separation-based Translation Pipeline

The previous method does not apply to use cases where the instrument tracks are not available separately and where the system must take the audio mixture as input. One way to tackle this is to adopt a pipeline of audio source separation and single-instrument translation. The source separation module isolates each of the tracks so that a single-instrument translation model can transform each of them into a target instrument.

To perform source separation, we adopt Demucs [23], a state-of-the-art music source separation model that operates in the waveform domain, taking an audio mixture and outputting isolated audio tracks. These stems are then fed into the universal network described in Section 3.1 for the translation phase.

3.3 Embedding-supervised Method

Training source separation models to separate every instrument in a mixture is a demanding task, and thus we favor solutions that do not depend on it. We have investigated a potential solution that we refer to as the embedding-supervised method, which performs a semi-separation task through the encoding process. In this case, the embedding-supervised encoder learns to capture the pitch-related features of one of the two instruments in the mixture. In order to achieve this, we need a pre-trained encoder that can provide the desirable pitch embeddings for a single instrument and assist the embedding-supervised encoder in distinguishing between the instruments throughout the training step. Recall that the universal encoder in our baseline model learns to extract pitch-related information from the input. Thus the output of such an encoder can provide the embeddings we seek. Based on this, we can train the embedding-supervised encoder to mimic the output of the universal encoder when given a mixture as the input. (Fig. 2)

For instance, if we have the mixture as:

$$A_{mix} = A_{strings} + A_{piano}$$

we feed the piano track A_{piano}:

$$A_{piano} := T_{piano} + P_{piano}$$

into the universal encoder, and its output will represent piano track pitch information (P_{piano}). We then input the mixture A_{mix} to the embedding-supervised encoder and train it to output the same code, i.e., the embeddings for P_{piano}. Consequently, we have a piano-specific encoder to extract the piano pitch content from any mixture. Note that for such training, existence of the stems in the dataset is necessary. We can isolate the information corresponding to one instrument via this approach without applying actual source separation. We can train one encoder per instrument with the help of the universal encoder. The model strives to minimize the loss function below:

$$L(E^i_{es}(mix), E_u(x^i)) \tag{1}$$

where E_{es} is the embedding-supervised encoder, mix is the audio segment from the input mixture, E_u is the universal encoder, x^i is the fragment of the instrument track i that we want to extract from the mixture, and L is L1 loss.

During inference, a pre-trained WaveNet decoder can also be engaged to translate the code to an arbitrary target instrument.

3.4 Music-STAR

Our ultimate goal is to realize the idea of multi-instrument translation where we do not need to engage source separation modules or additional encoders, and in general remove the restrictions of the aforementioned techniques. To this end, we introduce Music-STAR, designed explicitly for audio-based multi-instrument translation as described below.

As mentioned in Section 3.1, the universal network is trained using a random local pitch modulation that distorts the input to ensure that the encoder does not memorize the input content. Since the encoder learns to extract pitch-related information, we expect the code to embody data from the distorted segments. However, the decoder is trained using the teacher forcing technique, where the original audio segment is fed into the decoder alongside the embeddings produced by the encoder. The benefits of this approach are two-fold. First, the decoder learns the timbre of the original segment that is not present in the latent space. Second, for the decoder to reconstruct the original audio with the right pitch, it forces the encoder to represent the original segment in the latent space by removing the distortion. We will employ a similar idea below.

To begin, we represent our input audio as:

$$A_{input} = A_{strings} + A_{piano} := \\ T_{strings} + P_{strings} + T_{piano} + P_{piano}$$

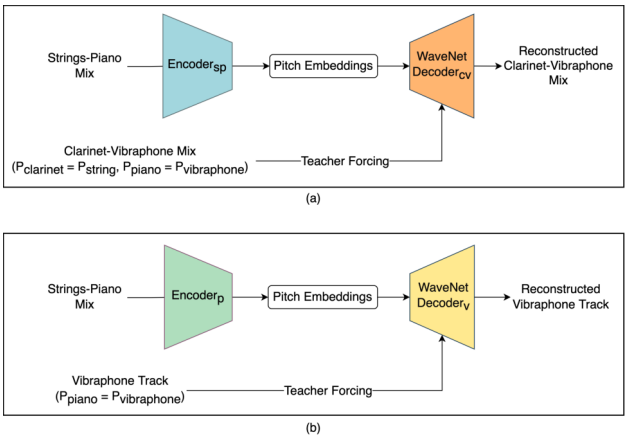

Figure 3. Training Music-STAR using (a) the mixture-supervised method to generated the target mixture, (b) the stem-supervised method to generate a single target stem. The modules subscripts *sp*, *cv*, *p*, and *v* corresponds to string-piano, clarinet-vibraphone, piano, and vibraphone, respectively.

The output we are looking for is:

$$A_{output} = A_{clarinet} + A_{vibraphone} := \\ T_{clarinet} + P_{clarinet} + T_{vibraphone} + P_{vibraphone}$$

where $P_{strings} = P_{clarinet}$ and $P_{piano} = P_{vibraphone}$.

To obtain this output, we should make sure that:

1. The encoder includes only the representations of P_{piano} and $P_{strings}$ in the embeddings, removing the information related to the other components of A_{input} (T_{piano} and $T_{strings}$), which has been proven possible when removing the distortion in [7].

2. The decoder generates the output signal by applying the target timbres that it has learned from the audio segments used for teacher forcing.

Accordingly, we conclude that the autoencoder will be able to extract the pitch-related features of the input mixture and translate it into target timbres if we train the model by:

1. Using the strings-piano mixture as the encoder's input, and

2. Using the clarinet-vibraphone counterpart of the input to apply teacher forcing to the decoder.

When the decoder is learning to translate the encoder's output to generate the clarinet-vibraphone segment, it inevitably guides the encoder to hand in helpful information for the task, which should be strings and piano pitch representations (see Fig. 3(a)). Since we train the model using the source and target audio mixtures, we name this approach the *mixture-supervised* method. The loss function for the mixture-supervised method is formulated as

$$\sum_{j} L(D(E(mix), t_j), t_j) \qquad (2)$$

where L is the cross-entropy loss, D is the decoder, E represents the encoder, mix is the input mixture segment, and t_j is the jth sample from the target mixture used for teacher forcing.

In our experiments, we also account for a variation of the mixture-supervised method where the target mixture (used in teacher forcing) is replaced by only one of its instrument tracks. We call the resulting method *stem-supervised*, in which we employ two autoencoders, each for translating one of the instruments in the mixture, and combine their outputs in the end to obtain the final mix (see Fig. 3(b)).

4. EXPERIMENTS

4.1 Single-instrument Translation Pipeline

We employ the universal network [7] to perform single-instrument translation. The network is originally trained on six classical music domains from the MusicNet dataset that includes mono audio segments with a sample rate of 16 kHz, which are later quantized by 8-bit mu-law encoding. The input files are randomly selected, and then a 0.75-second audio chunk is randomly segmented out of the file for every training input. A duration between 0.25 to 0.5 seconds of that segment is then selected for applying distortion by modulating the pitch.

In order to use this architecture as a baseline, we need to make sure that the decoders can generate the timbres included in the StarNet dataset. Therefore, we use the pre-trained universal encoder and finetune the decoders on the reduced StarNet dataset. Note that the data in reduced StarNet has the same properties as the data used for training the original model in terms of sample rate, bit depth, and the number of channels. Since the universal encoder has been trained using substantial computational resources on six domains and captures domain-agnostic features, it is powerful enough to successfully encode inputs from other domains. We finetuned the decoders with a batch size of 16, a learning rate of $1\mathrm{e}{-3}$, and a decay factor of 0.98 for 100 epochs.

4.2 Separation-based Translation Pipeline

We adopt Demucs [23] as the music source separation model, which is trained originally on MuseDB [24]. It is vital for our experiment that the network separates the two instruments present in the mixtures. Therefore, we train Demucs on preprocessed StarNet from scratch. We use the original learning rate of $3\mathrm{e}{-4}$ and Adam optimizer with a batch size of 32 for 250 epochs. The resulting model can take an arbitrary length of an input mixture and successfully separate the strings track from the piano or the clarinet track from the vibraphone.

The separated audio tracks are then post-processed to comply with the data configuration in reduced StarNet and fed into the finetuned models from Section 4.1 that correspond to their target instruments.

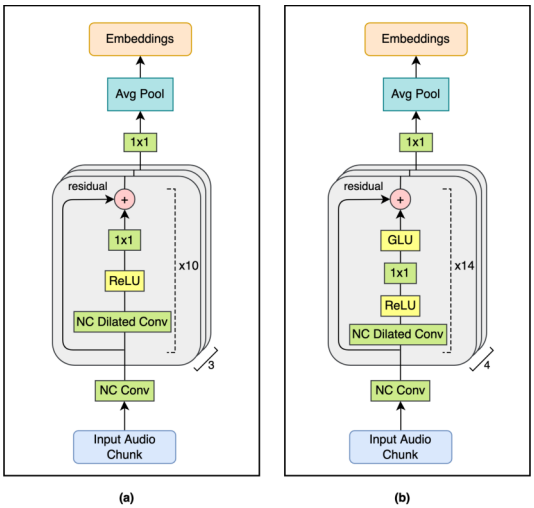

Figure 4. (a) The architecture of the universal encoder. (b) The architecture of the encoders used in the embedding-supervised and Music-STAR models.

4.3 Embedding-supervised Method

An autoencoder architecture builds the backbone of the model. This method focuses on training the embedding-supervised encoder with the help of the universal encoder. The architecture of the universal encoder is depicted in Fig. 4(a). Since the embedding-supervised encoder should learn to extract the same information as the universal encoder from a mixture rather than a stem, one would expect that a more complex architecture would be required for it. This was indeed confirmed by our pilot experiments.

The architecture we use as the embedding-supervised encoder is shown in Fig. 4(b). Similar to the universal encoder, it starts with a non-causal non-dilated convolution. The encoder consists of four blocks of 14-layer residual networks composed of non-causal dilated convolutions with a kernel size of 3, followed by ReLU nonlinearity and a 1x1 convolution. A GLU activation function follows the 1x1 convolution, and the output is summed with the input to form the residual connection. The result of this connection is then fed into the next layer of the encoder. The dilation increase factor of 2 and the 128 channels are identical to the universal encoder.

Unlike the baseline model that trains a universal encoder for all the domains in the training set, we need to train a single encoder for each instrument to extract content information from a mixture. The embedding-supervised encoders are trained on 0.75-second audio chunks randomly segmented out of the mixture files from reduced StarNet, and the same segments are extracted from the instrument tracks to be fed into the universal encoder. We adopt Adam optimization and exponential learning decay with a learning rate of $3e{-}4$, a decay factor of 0.98, and a batch size of 16 for a total of 100 epochs.

The finetuned WaveNet decoders described in Section 4.1 are then attached to the embedding-supervised encoders during inference to translate the code into the target instruments.

4.4 Music-STAR

We employ the WaveNet autoencoder architecture in the mixture-supervised method as well. The encoder we use for this method has the same architecture as the one used in the embedding-supervised method (Fig. 4(b)). The decoder's architecture is identical to the WaveNet decoders described in [7].

We train the models on the reduced version of StarNet. We pick random one-second audio segments of the input mixture and extract the same segment in the target mixture to apply teacher forcing to the decoder. Training is done using the Adam optimizer and exponential learning decay, where a learning rate of $3e{-}4$ and a decay rate of 0.99 are applied. The model is trained for a total of 100 epochs with a batch size of 16.

We use the same training configuration as above for the stem-supervised setting, except that two autoencoders are involved, each for transforming the source mixture into one of the target instruments.

5. EVALUATION

We conduct subjective and objective evaluations of the obtained audio and compare the re-instrumentation methods on three criteria:

1. Content preservation: how much of the pitch content of the input is retained in the output.
2. Style fit: How well the output presents the target timbres.
3. Audio quality: How clean and distortion-free the generated audio is.

5.1 Subjective Evaluation

The subjective evaluation provides a qualitative assessment of our models based on how users perceive the generated outputs. We carry out the subjective evaluation by distributing a survey among 30 participants, some of whom have a musical background. Each survey contains two different pieces, one from each domain (strings-piano and clarinet-vibraphone) selected out of 10 pieces in total. The target mixture is provided for each piece to demonstrate the gold standard, followed by corresponding translation outputs resulting from the five methods. The order of outputs is different for the two pieces, and the participants have no prior knowledge of the order.

Every piece in the survey is evaluated through three questions asking the participants to rank the five outputs based on:

1. How well they preserve the target musical content, which accounts for content preservation.
2. How well they present the instruments' tone colors (timbre) of the target piece, corresponding to style fit.
3. How clean and distortion-free they are, presenting the audio quality.

Method	Subjective			Objective	
	Content	Style	Quality	Content (Jaccard)	Style (Cosine)
Single-instrument	166	150	153	0.371	0.483
Separation-based	165	147	159	0.392	0.474
Embedding-supervised	161	157	149	0.350	0.472
Stem-supervised	154	190	183	0.323	**0.699**
Mixture-supervised	**254**	**256**	**256**	**0.426**	**0.698**

Table 1. Evaluation results on the three criteria of content preservation, style fit, and audio quality. The results from the subjective evaluation show the scores for each model according to the rankings provided by the surveys. Objective evaluation reports Jaccard similarity and cosine similarity for the first two criteria, respectively.

After collecting the surveys, we concluded the rankings for each question by scoring the models. For each criterion, the methods receive a score of 5 if one of their generated outputs is ranked first, a score of 4 if ranked second, a score of 3 if ranked third, a score of 2 if ranked fourth, and a score of 1 if ranked last. The aggregate scores are shown in Table 1. More detailed analysis is available online [1] .

5.2 Objective Evaluation

We aim to provide a quantitative quality assessment on content preservation and style fit using techniques employed in previous studies.

5.2.1 Content Preservation

Following the work by Cífka et al. [3], we assess the models on content preservation by calculating the Jaccard similarity between the pitch contours of the outputs and their corresponding gold standards. Since we are addressing multi-instrument music in our study, we extract the pitch contours using the multi-pitch Melodia algorithm [25] which provides the existing pitch frequencies in Hz. We round the frequency values to the nearest semitone. Then we express the similarity of the pitch sets of each time step in terms of the Jaccard index. Higher Jaccard similarity between the output and the gold standard signifies better content preservation.

5.2.2 Style Fit

We evaluate the style fit factor using the deep metric triplet network offered by Lee et al. [26]. The network consists of a backbone model, which provides embeddings for three inputs, and a triplet model that outputs a similarity score between those embeddings. The three inputs are called *the anchor*, *the positive*, and *the negative*. The network is trained to generate the embeddings in a way that the positive is closer to the anchor than the negative. During inference, a similarity score between the anchor and the other two will be reported.

Following Cífka et al. [3], we use MFCCs as the input features as they provide timbre-related information. We train the triplet network using the mixtures in the preprocessed StarNet dataset. For instance, the MFCCs of eight-second clarinet-vibraphone audio segments are used as both the anchors and the positive inputs in the training phase. At the same time, their counterparts from the strings-piano domain are regarded as negative inputs.

During inference, we presented the translation outputs from the five methods as the anchors, their corresponding gold standard as the positive, and the performance from the other domain as the negative. We report the average cosine similarity between the outputs of each translation method and their corresponding gold standards. Higher cosine similarity denotes more likeness to the target timbre and a better style fit.

5.3 Discussion

All evaluation results are presented in Table 1. The scores reported by the subjective evaluation are the total sum of the models' scores on both Clarinet-Vibraphone $\leftrightarrow$ Strings-Piano translations. Objective evaluation reports the average performance of models in translating the two domains. Both objective and subjective evaluations conclude that mixture-supervised Music-STAR is the predominant model in performing multi-instrument music translation. The mixture-supervised method outperforms the other methods in all the assessments except for objective style fit, where it reaches an almost equal cosine similarity with the stem-supervised method. An advantage that it has over the other methods is that the presence of the stems is not necessary for training the model. Mixture-supervised Music-STAR is, to the best of our knowledge, the only model capable of performing timbre translation on multiple instruments in a mixed signal.

The embedding-supervised method is, on average, the worst-performing model. The encoder used in this model is trained based on the universal encoder's outputs to extract the pitch information of one instrument out of a mixture. An imperfect ground truth places the model at the disadvantage of even more unsatisfactory performance. Also, the performance of single-instrument and separation-based translation models on different criteria are very similar.

6. CONCLUSION

This paper introduced Music-STAR, the first audio-based multi-instrument music translation system. Music-STAR tackles multi-instrument translation without applying explicit source separation to the input mixtures. We also introduce the StarNet dataset that includes two-instrument pieces performed in two domains alongside their stems. We explored a variety of possible solutions based on the WaveNet autoencoder, and finally reached a successful mixture-supervised method capable of performing simultaneous source separation and pitch-timbre disentanglement for two instruments.

Future work will target increasing the number of instrument tracks in the mixtures, and adding to the variety of instrument combinations. We also plan to develop a more ornate dataset in terms of the details on articulations and dynamics.

[1] `https://mahshidaln.github.io/Music-STAR`

7. REFERENCES

[1] S. Dai, Z. Zhang, and G. G. Xia, "Music style transfer: A position paper," *arXiv preprint arXiv:1803.06841*, 2018.

[2] A. Bitton, P. Esling, and A. Chemla-Romeu-Santos, "Modulated variational auto-encoders for many-to-many musical timbre transfer," *arXiv preprint arXiv:1810.00222*, 2018.

[3] O. Cífka, A. Ozerov, U. Şimşekli, and G. Richard, "Self-supervised VQ-VAE for one-shot music style transfer," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 96–100.

[4] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "TimbreTron: A WaveNet (CycleGAN (CQT (audio))) pipeline for musical timbre transfer," *arXiv preprint arXiv:1811.09620*, 2018.

[5] D. K. Jain, A. Kumar, L. Cai, S. Singhal, and V. Kumar, "ATT: Attention-based timbre transfer," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–6.

[6] C.-Y. Lu, M.-X. Xue, C.-C. Chang, C.-R. Lee, and L. Su, "Play as you like: Timbre-enhanced multi-modal music style transfer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1061–1068.

[7] N. Mor, L. Wolf, A. Polyak, and Y. Taigman, "A universal music translation network," *arXiv preprint arXiv:1805.07848*, 2018.

[8] W. T. Lu and L. Su, "Transferring the style of homophonic music using recurrent neural networks and autoregressive model," in *ISMIR*, 2018, pp. 740–746.

[9] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer," *arXiv preprint arXiv:1809.07600*, 2018.

[10] O. Cífka, U. Şimşekli, and G. Richard, "Groove2Groove: One-shot music style transfer with supervision from synthetic data," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2638–2650, 2020.

[11] J. Wang, C. Jin, W. Zhao, S. Liu, and X. Lv, "An unsupervised methodology for musical style translation," in *2019 15th International Conference on Computational Intelligence and Security (CIS)*. IEEE, 2019, pp. 216–220.

[12] Y.-N. Hung, I. Chiang, Y.-A. Chen, and Y.-H. Yang, "Musical composition style transfer via disentangled timbre representations," *arXiv preprint arXiv:1905.13567*, 2019.

[13] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[14] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard GAN," *arXiv preprint arXiv:1807.00734*, 2018.

[15] Y. Pang, J. Lin, T. Qin, and Z. Chen, "Image-to-Image translation: Methods and applications," *arXiv preprint arXiv:2101.08629*, 2021.

[16] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," *arXiv preprint arXiv:2001.04643*, 2020.

[17] F. Ganis, E. F. Knudesn, S. V. Lyster, R. Otterbein, D. Südholt, and C. Erkut, "Real-time timbre transfer and sound synthesis using DDSP," *arXiv preprint arXiv:2103.07220*, 2021.

[18] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with WaveNet autoencoders," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1068–1077.

[19] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[20] J. Thickstun, Z. Harchaoui, and S. M. Kakade, "Learning features of music from scratch," in *International Conference on Learning Representations (ICLR)*, 2017.

[21] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[22] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[23] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," *arXiv preprint arXiv:1911.13254*, 2019.

[24] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," Dec. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.1117372

[25] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[26] J. Lee, N. J. Bryan, J. Salamon, Z. Jin, and J. Nam, "Disentangled multidimensional metric learning for music similarity," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6–10.

LEARNING UNSUPERVISED HIERARCHIES OF AUDIO CONCEPTS

Darius Afchar[1,2] **Romain Hennequin**[1] **Vincent Guigue**[2]
[1]Deezer Research, France [2]MLIA, ISIR - Sorbonne Université - CNRS, France
research@deezer.com

ABSTRACT

Music signals are difficult to interpret from their low-level features, perhaps even more than images: e.g. highlighting part of a spectrogram or an image is often insufficient to convey high-level ideas that are genuinely relevant to humans. In computer vision, concept learning was therein proposed to adjust explanations to the right abstraction level (e.g. detect clinical concepts from radiographs). These methods have yet to be used for MIR.

In this paper, we adapt concept learning to the realm of music, with its particularities. For instance, music concepts are typically non-independent and of mixed nature (e.g. genre, instruments, mood), unlike previous work that assumed disentangled concepts. We propose a method to learn numerous music concepts from audio and then automatically hierarchise them to expose their mutual relationships. We conduct experiments on datasets of playlists from a music streaming service, serving as a few annotated examples for diverse concepts. Evaluations show that the mined hierarchies are aligned with both ground-truth hierarchies of concepts – when available – and with proxy sources of concept similarity in the general case.

1. INTRODUCTION

Music signals are challenging to interpret [1]. For instance, inspecting a spectrogram by parts – *e.g.* think of attention maps [2] – does not convey much meaning with respect to the high-level descriptions that are useful to humans – *e.g.* the mood of a song. Fortunately, a solution was proposed in a field that shares similar problems: computer vision. As pixels of images are too low-level to be understandable, *concept learning* was introduced to rationalise the abstractions steps involved in a feed-forward model (*e.g.* detection of colours, shapes, patterns, ...), which in turn can be used to describe the content of an image directly, and to align models with human reasonings better. Indeed, this field has had successful applications in explainability (XAI) – dissecting a model's predictions into human-grounded concepts [3–5], interactive learning [6] or knowledge transfer between tasks [7]. However, these methods have yet to be used for Music Information Retrieval (MIR).

Music has particularities that are not captured by the current formulation of concept learning. For instance, let us consider genres as common musical concepts we would like to detect from spectrograms. Genres are typically blended as they influence or result from other genres (*e.g. Punk Rock*). By contrast, most literature on concept learning has considered sets of few identified and disentangled concepts, which does not apply to genres. Moreover, one trend in MIR is to incorporate multiple types of descriptors into a shared space (*e.g.* mood, genre, instruments [8, 9]), thus requiring handling even more entanglements.

We thus propose to study a novel problem for concept learning that is relevant for music: **learning hierarchies of concepts**. Indeed, a hierarchy enables navigating thousands of non-independent concepts without cognitive overload by uncovering their mutual relationships with a structure, thus rationalising which concepts share the same class or are derived from one another. In addition, the use of taxonomy – *i.e.* hierarchy – is common in musicology and feels like a natural representation in this context [10, 11].

The problem we study is interesting for *music representation learning* because research is often limited by data labelling, which is labour-intensive and can be ambiguous or tied to a specific dataset. Meanwhile, concepts can be learned with few shots, which open the doors to new sources of music descriptors. In particular, we propose to leverage a dataset of playlists from a music streaming service as a folksonomy of concepts. Playlists are often built with a specific concept; there exists one for any genre, instrument, mood, and many more niche ideas. Yet, there is no overall ground-truth organisation of playlists, making their use cumbersome. That is why we propose to solve this task by first adapting a method from concept learning research to learn music concepts from playlists in a few-shot manner and then, building on top of this method, automatically derive a hierarchy from the learned concepts.

The paper proceeds as follows: we provide background on concept and hierarchy learning (section 2); we adapt the literature of concept-based explanations to the realm of MIR (3); we present a novel way to organise the detected concepts into a hierarchy in an unsupervised manner (4); finally, we experiment and discuss our attempt to provide a new audio concepts hierarchy with mixed types (5).

2. PRELIMINARIES AND RELATED WORK

We first present the literature on concept learning and hierarchy mining, then frame our problem through XAI to uncover common pitfalls to avoid when interpreting data.

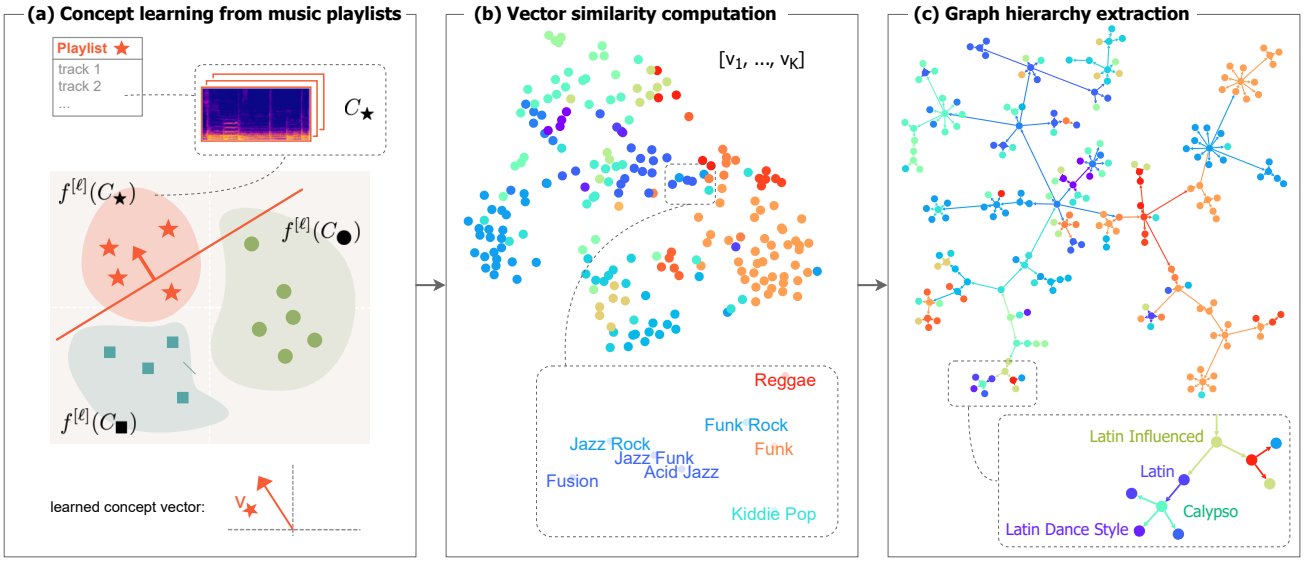

Figure 1. **Overview of our method** *(a)* CAVs learning (section 3.1); *(b)* t-SNE projection of concepts $\mathcal{V}_{\text{genre}}$ learned on a small dataset with 219 music genres; *(c)* resulting extracted hierarchy $H(S(\mathcal{V}_{\text{genre}}))$. Nodes colours denote ground-truth clusters. Experiments on a bigger concept dataset are conducted in section 5.

2.1 Concept learning

Concept learning has witnessed growing interest in recent years [4]. The introduction of concepts stems from the inadequacy of usual *explanation spaces* with human mental models of problems: tabular features, pixels, and words in sentences are sometimes too low-level to build relevant explanations or to interact with. This idea is far from novel from a purely technical standpoint: a stream of papers aim to learn a set of sub-attributes in a fully supervised manner (*e.g.* facial features [12], or animal attributes [13]), that are then combined to predict some end-task labels, typically through a linear regression to track the importance of each attribute. This idea has been revamped recently for *concept bottleneck* learning [14] and *prototypical part learning* [15]. In the same vein, *disentanglement techniques* [16] have sought to realign latent representations to interpretable attributes. All those works rely on the assumption that a somewhat complete set of useful concepts exists, well-defined and rather disentangled, and that a corresponding labelled dataset is readily available to train on.

This is rarely the case in practice. Kim et al. introduced CAVs for images [3] and demonstrated that concepts could be learned reliably from few annotated examples – technical details are provided in section 3.1. Going beyond the previously few datasets annotated for concepts, this work has led to new and diverse applications: skin lesion interpretations [17], emotion recognition [18], interactivity [6, 19], automatic concept learning [20], batch normalisation [21], sufficiency of explanations with shapley values [22, 23], causal inference [24]. To our knowledge, there exists no application in the music domain.

2.2 Unsupervised Hierarchy Mining

Using knowledge graphs and taxonomies is frequent leverage in MIR [25]. Unsupervised learning of hierarchy is, however, less so. Part of it lies in the difficulty of adequately defining the meaning of the hierarchical relations, learning the hierarchy in a tractable way, and evaluating the found hierarchy without ground-truth.

Fortunately, there is active literature coming from *natural language processing (NLP)* research. Many work proposed variants of *topic modelling* [26] to enable sampling hierarchised structures [27, 28]. However, these methods get computationally intractable as the number of nodes, topics, and hierarchical levels grows. Many successes thus came from scalable greedy methods such as hierarchical clustering [29, 30], rule-mining [31, 32] and graph-based analyses [33, 34]. Outside of NLP, hierarchies have been learned based on the latent manifold geometry [35], predefined structure [36], and for images [37, 38].

To our knowledge, these techniques have never been combined with concept learning, as we propose.

2.3 Explainability

Our end goal is to interpret music signals, which is linked to explainability (XAI). Before jumping into our method to learn a hierarchy of concepts, we need to acknowledge that many works have debated the sanity of explanation methods [39–43] and the fact that these methods largely disagree with one another [44]. XAI has vast boundaries, which has also percolated to its music sub-field for which there exists many explanation paradigms and evaluations [45]. Explanations for MIR are thus also prone to exhibiting pathologies that hamper knowledge mining: *e.g.* shortcut learning [46], lack of clear definitions [39, 47], flawed evaluations [48], out-of-distribution artifacts [49, 50] or misalignment with human reasonings [51–54].

Uncovering these failure cases sharpens our understanding of how models make decisions, and shows that we have to go beyond evasive considerations on explain-