Musical Composition Style Transfer via Disentangled Timbre Representations

Yun-Ning Hung[1] , **I-Tung Chiang**[1] , **Yi-An Chen**[2] and **Yi-Hsuan Yang**[1]
[1]Research Center for IT Innovation, Academia Sinica, Taiwan
[2]KKBOX Inc., Taiwan
{biboamy, itungchiang}@citi.sinica.edu, annchen@kkbox.com, yang@citi.sinica.edu

Abstract

Music creation involves not only composing the different parts (e.g., melody, chords) of a musical work but also arranging/selecting the instruments to play the different parts. While the former has received increasing attention, the latter has not been much investigated. This paper presents, to the best of our knowledge, the first deep learning models for rearranging music of arbitrary genres. Specifically, we build encoders and decoders that take a piece of polyphonic musical audio as input, and predict as output its musical score. We investigate disentanglement techniques such as adversarial training to separate latent factors that are related to the musical content (pitch) of different parts of the piece, and that are related to the instrumentation (timbre) of the parts per short-time segment. By disentangling pitch and timbre, our models have an idea of how each piece was composed and arranged. Moreover, the models can realize "composition style transfer" by rearranging a musical piece without much affecting its pitch content. We validate the effectiveness of the models by experiments on instrument activity detection and composition style transfer. To facilitate follow-up research, we open source our code at https://github.com/biboamy/instrument-disentangle.

1 Introduction

Music generation has long been considered as an important task for AI, possibly due to the complicated mental and creative process that is involved, and its wide applications in our daily lives. Even if machines cannot reach the same professional level as well-trained composers or songwriters, machines can cooperate with human to make music generation easier or more accessible to everyone.

Along with the rapid development of deep generative models such as generative adversarial networks (GANs) [Goodfellow and others, 2014], the research on music generation is experiencing a new momentum. While a musical work is typically composed of melody, chord, and rhythm (beats), a major trend of recent research focuses on creating original content for all or some of these three parts [Roberts *et al.*, 2017; Dong *et al.*, 2018a]. That is, we expect machines can learn the complex musical structure and compose music like human composers. This has been technically referred to as *music composition* [Briot *et al.*, 2017].

Another trend is to generate variations of existing musical material. By preserving the pitch content of a musical piece, we expect machines to modify style-related attributes of music, such as genre and instrumentation. This has been technically referred to as *music style transfer* [Dai *et al.*, 2018]. A famous example of style transfer, though not fully-automatic, is the "Ode to Joy" project presented by [Pachet, 2016]. The project aims to rearrange (or reorchestrate) Beethoven's Ode to Joy according to the stylistic conventions of seven other different musical genres such as Jazz and Bossa Nova.[1]

In this paper, we are in particular interested in such a *music rearrangement* task. The work of [Pachet, 2016] is inspiring, but as the target styles are by nature very different, they had to tailor different machine learning models, one for each style. Moreover, they chose to use non-neural network based models such as Markov models, possibly due to the need to inject musical knowledge manually to ensure the final production quality. As a result, the solution presented by [Pachet, 2016] is not general and it is not easy to make extensions. In contrast, we would like to investigate a universal neural network model with which we can easily change the instrumentation of a given music piece fully automatically, without much affecting its pitch content. While we do not think such a style-agnostic approach can work well for all the possible musical styles, it can be treated as a starting point from which style-specific elements can be added later on.

Music rearrangement is closely related to the art of *music arrangement* [Corozine, 2015]. For creating music, it involves not only composing the different parts (e.g., melody, chords) of a musical work but also arranging/selecting the instruments to play the different parts. For example, chord and melody can be played by different instruments. Setting the instrumentation is like *coloring* the music. Therefore, music arrangement is an "advanced" topic, and is needed towards generating fully-fledged music material. Compared to music arrangement, music rearrangement is a style transfer task and is like *re-coloring* an existing music piece. By contributing to music rearrangement, we also contribute indirectly to the

[1][Online] https://www.youtube.com/watch?v=buXqNqBFd6E

more challenging automatic music arrangement task, which has been seldom addressed in the literature as well.

Music rearrangement is not an easy task even for human beings and requires years of professional training. In order to convincingly perform music style transfer between two specific styles, a musician must first be familiar with the annotation, instrument usage, tonality, harmonization and theories of the two styles. The task gets harder when we consider more styles. For machines to rearrange a musical piece, we have to consider not only the pitch content of the given music, but also the pitch range of each individual instrument as well as the relation between different instruments (e.g., some instrument combination creates more harmonic sounds). And, it is hard to find *paired data* demonstrating different instrumentations of the same music pieces [Crestel *et al.*, 2017]. A neural network may not learn well when the data size is small.

To address these challenges, we propose to train *music transcription* neural networks that take as input an audio file and generate as output the corresponding musical score. In addition to predicting which notes are played in the given audio file, the model also has to predict the instrument that plays each note. As a result, the model learns both pitch- and timbre-related representations from the musical audio. While the model is not trained to perform music rearrangement, we investigate "disentanglement techniques" such as adversarial training [Liu *et al.*, 2018b; Lee *et al.*, 2018] to separate latent factors of the model that are pitch- and timbre-related. By disentangling pitch and timbre, the model has an idea of how a musical piece was composed and arranged. If the disentanglement is well done, we expect that we can rearrange a musical piece by holding its pitch-related latent factors fixed while manipulating the timbre-related ones.

We propose and investigate two models in this paper. They share the following two advantages. First, they require only pairs of audio and MIDI files, which are relatively easier to collect than different arrangements of the same music pieces. Second, they can take any musical piece as input and rearrange it, as long as we have its audio recording.

In sum, the main contributions of this work include:

- To the best of our knowledge, this work presents the first deep learning models that realize music rearrangement, a composition style transfer task [Dai *et al.*, 2018] (see Section 2.1), for polyphonic music of arbitrary genres.
- While visual attribute disentanglement has been much studied recently, attribute disentanglement for musical audio has only been studied in our prior work [Hung *et al.*, 2018], to our knowledge. Extending from that work, we report comprehensive evaluation of the models, emphasizing their application to rearrangement.

Musical compositional style transfer is still at its early stage. We hope this work can call for more attention toward approaching this task with computational means.

2 Related Work

2.1 Music Style Transfer

According to the categorization of [Dai *et al.*, 2018], there are three types of music style transfer: *timbre style transfer*, *performance style transfer*, and *composition style transfer*. To our knowledge, timbre style transfer receives more attention than the other two lately. The task is concerned with altering the timbre of a musical piece in the audio domain. For example, [Engel *et al.*, 2017] applied WaveNet-based autoencoders to audio waveforms to model the musical timbre of short single notes, and [Lu *et al.*, 2019] employed a GAN-based network on spectrograms to achieve long-structure timbre style transfer. However, for the latter two types of music style transfer, little effort has been made.

Music rearrangement can be considered as a composition style transfer task, which is defined as a task that "preserve[s] the identifiable melody contour (and the underlying structural functions of harmony) while altering some other score features in a meaningful way" [Dai *et al.*, 2018]. The recent work on composition style transfer we are aware of are that by [Pati, 2018] and [Kaliakatsos-Papakostas *et al.*, 2017], which used neural networks and rules respectively to deal with melody, rhythm, and chord modification. Our work is different in that we aim to modify the instrumentation.

2.2 Disentanglement for Style Transfer

Neural style transfer was first introduced in image style transfer [Gatys *et al.*, 2016]. In this work, they proposed to learn an image representation to separate and recombine content and style. Since then, several methods have been proposed to modify the style of a image. [Liu *et al.*, 2018b] trained an auto-encoder model supervised by content ground truth labels to learn a content-invariant representation, which can be modified to reconstruct a new image. [Lee *et al.*, 2018] showed that it is possible to learn a disentangled representation from unpaired data by using cross-cycle consistency loss and adversarial training, without needing the ground truth labels.

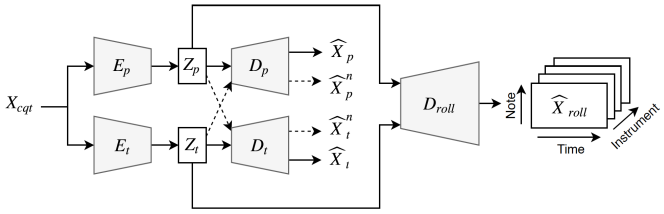
Disentangled representation learning for music is still new and has only been studied by [Brunner *et al.*, 2018] for symbolic data. By learning a style representation through style classifier, their model is able to modify the pitch, instrumentation and velocity given different style codes. Our model is different in that we deal with audio input rather than MIDIs.

3 Proposed Models

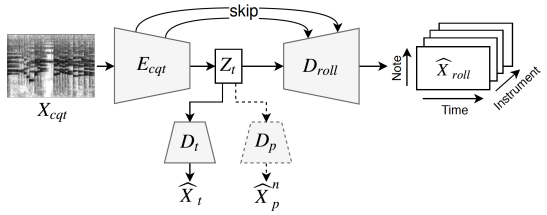
Figure 1 shows the architecture of the proposed models. We opt for using the encoder/decoder structures as the backbone of our models, as such networks learn latent representations (a.k.a., latent codes) of the input data. With some labeled data, it is possible to train an encoder/decoder network in a way that different parts of the latent code correspond to different data attributes, such as pitch and timbre. Both models use encoders/decoders and adversarial training to learn music representations, but the second model additionally uses skip connections to deal with the pitch information. We note that both models perform polyphonic music transcription. We present the details of these models below.

3.1 Input/Output Data Representation

The input to our models is an audio waveform with arbitrary length. To facilitate pitch and timbre analysis, we firstly convert the waveform into a time-frequency representation that shows the energy distribution across different frequency bins

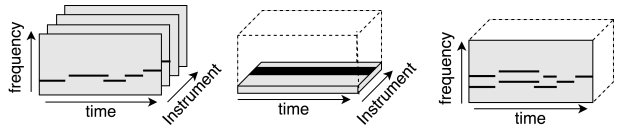


(a) DuoED: Use separate encoders/decoders for timbre and pitch



(b) UnetED: Use skip connections to process pitch

Figure 1: The two proposed encoder/decoder architectures for learning disentangled timbre representations (i.e., $\mathbf{Z}_t$) for musical audio. The dashed lines indicate the adversarial training components. (Notations: CQT—a time-frequency representation of audio; roll–multi-track pianoroll; E—encoder; D—decoder; Z—latent code; t—timbre; p—pitch; skip—skip connections).



(a) Pianoroll (b) Instrument roll (c) Pitch roll

Figure 2: Different symbolic representations of music.

for each short-time frame. Instead of using the short-time Fourier transform, we use the constant-Q transform (CQT), for it adopts a logarithmic frequency scale that better aligns with our perception of pitch [Bittner *et al.*, 2017]. CQT also provides better frequency resolution in the low-frequency part, which helps detect the fundamental frequencies.

As will be described in Section 3.5, our encoders and decoders are designed to be fully-convolutional [Oquab *et al.*, 2015; Liu *et al.*, 2019], so that our models can deal with input of any length in testing time. However, for the convenience of training the models with mini-batches, in the training stage we divide the waveforms in our training set into 10-second chunks (without overlaps) and use these chunks as the model input, leading to a matrix $\mathbf{X}_{cqt} \in \mathcal{R}^{F\times T}$ of fixed size for each input. In our implementation, we compute CQT with the `librosa` library [McFee *et al.*, 2015], with 16,000 Hz sampling rate and 512-sample window size, again with no overlaps. We use a frequency scale of 88 bins, with 12 bins per octave to represent each note. Hence, $F = 88$ (bins) and $T = 312$ (frames).

We use the pianorolls [Dong *et al.*, 2018b] (see Figure 2a) as the target output of our models. A pianoroll is a binary-valued tensor that records the presence of notes (88 notes here) across time for each track (i.e., instrument). When we consider M instruments, the target model output would be $\mathbf{X}_{roll} \in \{0,1\}^{F\times T\times M}$. $\mathbf{X}_{roll}$ and $\mathbf{X}_{cqt}$ are temporally aligned, since we use MIDIs that are time-aligned with the audio clips to derive the pianorolls, as discussed in Section 3.5. As shown in Figures 1 and 2, besides asking our models to generate $\mathbf{X}_{roll}$, we use the *instrument roll* $\mathbf{X}_t \in \{0,1\}^{M\times T}$ and *pitch roll* $\mathbf{X}_p \in \{0,1\}^{F\times T}$ as supervisory signals to help learn the timbre representation. Both $\mathbf{X}_t$ and $\mathbf{X}_p$ can be computed from $\mathbf{X}_{roll}$ by summing along a certain dimension.

3.2 The DuoED Model

The architecture of DuoED is illustrated in Figure 1a. The design is inspired by [Liu *et al.*, 2018b], but we adapt the model to encode music. Specifically, we train two encoders E_t and E_p to respectively convert $\mathbf{X}_{cqt}$ into the *timbre code* $\mathbf{Z}_t = E_t(\mathbf{X}_{cqt}) \in \mathcal{R}^{\kappa\times\tau}$ and *pitch code* $\mathbf{Z}_p = E_p(\mathbf{X}_{cqt}) \in \mathcal{R}^{\kappa\times\tau}$. We note that, unlike in the case of image representation learning, here the latent codes are *matrices*, and we require that the second dimensions (i.e., τ) represent time. This way, each column of $\mathbf{Z}_t$ and $\mathbf{Z}_p$ is a κ-dimensional representation of a temporal segment of the input. For abstraction, we require $\kappa\tau < FT$.

DuoED also contains three decoders D_{roll}, D_t and D_p. The encoders and decoders are trained such that we can use $D_{roll}([\mathbf{Z}_t^T, \mathbf{Z}_p^T]^T)$ to predict $\mathbf{X}_{roll}$, $D_t(\mathbf{Z}_t)$ to predict $\mathbf{X}_t$, and $D_p(\mathbf{Z}_p)$ to predict $\mathbf{X}_p$. The prediction error is measured by the *binary cross entropy* between the ground truth and the predicted one. For example, for the timbre classifier D_t, it is:

$$L_t = -\sum \left[\mathbf{X}_t \cdot \ln\sigma(\widehat{\mathbf{X}}_t) + (1-\mathbf{X}_t)\cdot\ln(1-\sigma(\widehat{\mathbf{X}}_t))\right], \quad (1)$$

where $\widehat{\mathbf{X}}_t = D_t(\mathbf{Z}_t)$, '$\cdot$' denotes the element-wise product, and σ is the sigmoid function that scales its input to $[0,1]$. We can similarly define L_{roll} and L_p.

In each training epoch, we optimize *both* the encoders and decoders by minimizing L_{roll}, L_t and L_p for the given training batch. We refer to the way we train the model as using the "temporal supervision," since to minimize the loss terms L_{roll}, L_t and L_p, we have to make accurate prediction for each of the T time frames.

When the adversarial training strategy is employed (i.e., those marked by dashed lines in Figure 1a), we additionally consider the following two loss terms:

$$L_t^n = -\sum \left[\ln(1-\sigma(\widehat{\mathbf{X}}_t^n))\right], \quad (2)$$

$$L_p^n = -\sum \left[\ln(1-\sigma(\widehat{\mathbf{X}}_p^n))\right], \quad (3)$$

where $\widehat{\mathbf{X}}_t^n = D_t(\mathbf{Z}_p)$, $\widehat{\mathbf{X}}_p^n = D_p(\mathbf{Z}_t)$, meaning that we feed the 'wrong' input (purposefully) to D_t and D_p.

The equation must close to zero, which means when we use the wrong input, we expect D_t and D_p can output nothing (i.e., all zeros), since, e.g., $\mathbf{Z}_t$ is supposed not to contain any pitch-related information.

Please note that, in adversarial training, we use L_t^n and L_p^n to update the *encoders only*. This is to preserve the function of the decoders in making accurate predictions.

3.3 The UnetED Model

The architecture of UnetED is depicted in Figure 1b, which has a U-shape similar to [Ronneberger *et al.*, 2015]. In UnetED, we learn only one encoder E_{cqt} to get a single latent representation $\mathbf{Z}_t$ of the input $\mathbf{X}_{cqt}$. We add skip connections between E_{cqt} and D_{roll}, which enables lower-layer information of E (closer to the input) to be passed directly to the higher-layer of D (closer to the output), making it easier to train deeper models. The model learn E_{cqt} and D_{roll} by minimizing L_{roll}, the cross entropy between $D_{roll}(\mathbf{Z}_t)$ and the pianoroll $\mathbf{X}_{roll}$. Moreover, we promote timbre information in $\mathbf{Z}_t$ by refining E_{cqt} and learning a classifier D_t by minimizing L_t (see Eq. (1)). When the adversarial training strategy is adopted, we design a GAN-like structure (i.e., unlike the case in DuoED) to dispel pitch information from timbre representation. That is, the model additionally learns a pitch classifier D_p to minimize the loss L_p between $D_p(\mathbf{Z}_t)$ and X_p. This loss function only updates D_p. Meanwhile, the encoder E_{cqt} tries to fool D_p, where the ground truth matrices should be zero matrices. That is, the encoder should minimize L_p^n, and the loss function only affects the encoder.

The design of UnetED is based on the following intuitions. First, since $\mathbf{Z}_t$ is supposed not to carry pitch information, the only way to obtain the pitch information needed to predict $\mathbf{X}_{roll}$ is from the skip connections. Second, it is fine to assume that the skip connections can pass over the pitch information, due to the nice one-to-one time-frequency correspondence between $\mathbf{X}_{cqt}$ and each frontal slice of $\mathbf{X}_{roll}$.[2] Moreover, in $\mathbf{X}_{cqt}$ pitch only affects the lowest partial of a harmonic series created by a musical note, while timbre affects all the partials. If we view pitch as the "boundary" outlining an object, and timbre as the "texture" of that object, detecting the pitches may be analogous to image segmentation, for which U-nets have been shown effective in solving [Ronneberger *et al.*, 2015].

We note that the major difference between DuoED and UnetED is that there is a pitch code $\mathbf{Z}_p$ in DuoED. Although the pitch representation may benefit other tasks, for composition style transfer we care more about the timbre code $\mathbf{Z}_t$.

3.4 Composition Style Transfer

While both DuoED and UnetED are not trained to perform music rearrangement, they can be applied to music rearrangement due to the built-in pitch/timbre disentanglement. Specifically, for style transfer, we are given a source clip `A` and a target clip `B`, both audio files. We can realize style transfer by using `A` as the input to the encoder of DuoED or UnetED to get content information (i.e., through the pitch code $\mathbf{Z}_p^{(\mathtt{A})}$ or the skip connections), but then combine it with the timbre code $\mathbf{Z}_t^{(\mathtt{B})}$ obtained from `B` to generate an original pianoroll $\mathbf{X}_{roll}^{(\mathtt{A})\rightarrow(\mathtt{B})}$, from which we can create synthesized audio.

3.5 Implementation Details

Since the input and output are both matrices, we use convolutional layers in all the encoders and decoders of DuoED and UnetED. To accommodate input of variable length, we adopt a fully-convolutional design [Oquab *et al.*, 2015], meaning that we do not use pooling layers at all. All the encoders in our models are composed of four residual blocks. Each block contains three convolution layers and two batch normalization layers. The decoder D_{roll} has the same structure, but use transpose convolution layers to do the upsampling. There are in total twelve layers in both encoder and pianoroll decoder D_{roll}. For the pitch and timbre decoder, we use three transpose convolution layers to reconstruct the pitch roll and instrument roll from latent representation. Moreover, we use leaky ReLU as the activation function for all layers but the last one, where we use the sigmoid function. Both DuoED and UnetED are trained using stochastic gradient descend with momentum 0.9. The initial learning rate is set to 0.005.

We use the newest released MuseScore dataset [Hung *et al.*, 2019] to train the proposed models. This dataset contains 344,166 paired MIDI and MP3 files. Most MP3 files were synthesized from the MIDIs with the MuseScore synthesizer by the uploaders. Hence, the audio and MIDIs are already time-aligned. We further ensure temporal alignment by using the method proposed by [Raffel and Ellis, 2016]. We then convert the time-aligned MIDIs to pianorolls with the `pypianoroll` package [Dong *et al.*, 2018b]. The dataset contains music of different genres and 128 different instrument categories as defined by the MIDI spec.

4 Experiment

As automatic music rearrangement remains a new task, there are no standard metrics to evaluate our models. We propose to firstly evaluate our models objectively with the surrogate task of *instrument activity detection* (IAD) [Gururani and Lerch, 2017]—i.e., detecting the activity of different instruments for each short-time audio segment—to validate the effectiveness of the obtained instrument codes $\mathbf{Z}_t$. After that, we evaluate the performance of music rearrangement subjectively, by means of a user study.

4.1 Evaluation on Instrument Activity Detection

We evaluate the performance for IAD using the 'MedleyDB + Mixing Secret' (M&M) dataset proposed by [Gururani and Lerch, 2017]. The dataset contains real-world music recordings (i.e., not synthesized ones) of various genres. By evaluating our model on this dataset, we can compare the result with quite a few recent work on IAD. Following the setup of [Hung *et al.*, 2019], we evaluate the result for per-second IAD in terms of the area under the curve (AUC). We compute AUC for each instrument and report the per-instrument AUC as well as the average AUC across the instruments.

IAD is essentially concerned with the prediction of the instrument roll shown in Figure 2b. As our DuoED and UnetED models are pre-trained on a synthetic dataset MuseScore, we train additional instrument classifiers D_t' with the pre-defined training split of the M&M dataset (200 songs) [Hung *et al.*, 2019]. D_t' takes as input the timbre code $\mathbf{Z}_t$ computed by the pre-trained models. Following [Gururani *et al.*, 2018], we use a network of four convolution layers and two fully-connected layers for the instrument classifiers D_t'.[3] We use the estimate

[2] That is, both $X_{cqt}(i,j)$ and $X_{roll}(i,j,m)$ refer to the activity of the same musical note i for the same time frame j.

[3] We do not use the M&M training split to fine-tune the original

Method	training set	Piano	Guitar	Violin	Cello	Flute	Avg
[Hung and Yang, 2018]	MuseScore training split	0.690	0.660	0.697	0.774	0.860	0.736
[Liu *et al.*, 2019]	YouTube-8M	0.766	0.780	0.787	0.755	0.708	0.759
[Hung *et al.*, 2019]	MuseScore training split	0.718	0.819	0.682	0.812	**0.961**	0.798
[Gururani *et al.*, 2018]	M&M training split	0.733	0.783	0.857	**0.860**	0.851	0.817
DuoED updated	MuseScore (pre), M&M training split	0.721	0.790	0.865	0.810	0.912	0.815
UnetED updated	MuseScore (pre), M&M training split	**0.781**	**0.835**	**0.885**	0.807	0.832	**0.829**
[Hadad *et al.*, 2018] updated	MuseScore (pre), M&M training split	0.745	0.807	0.816	0.769	0.883	0.804
[Liu *et al.*, 2018a] updated	MuseScore (pre), M&M training split	0.808	0.844	0.789	0.766	0.710	0.793

Table 1: Average AUC scores of per-second instrument activity detection on the test split of the 'MedleyDB+Mixing Secret' (M&M) dataset [Gururani and Lerch, 2017; Hung *et al.*, 2019], for five instruments. For the latter four methods, we pre-train ('pre') the models on the MuseScore dataset [Hung *et al.*, 2019] and then use the training split of M&M for updating the associated timbre classifiers.

evaluated roll	model	Avg AUC
Instrument roll	DuoED w/o Adv	0.731
	DuoED w Adv	**0.741**
	UnetED w/o Adv	0.733
	UnetED w Adv	**0.754**
Sum up from Pianoroll	DuoED w/o Adv	0.778
	DuoED w Adv	**0.781**
	UnetED w/o Adv	0.778
	UnetED w Adv	**0.783**

Table 2: The same evaluation task as that in Table 1. We compare the models trained on MuseScore only here, with ('w/') or without ('w/o') adversarial training ('Adv'). We evaluate both the instrument roll predicted by the timbre decoder D_t, and the instrument roll obtained by summing up the pianoroll predicted by D_{roll}.

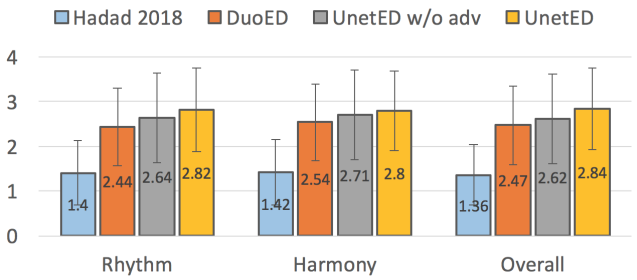


Figure 3: Result of our user study on music rearrangement.

of D'_t as the predicted instrument roll.

Table 1 shows the evaluation result on the pre-defined test split of the M&M dataset (69 songs) [Hung *et al.*, 2019] of four state-of-the-art models (i.e., the first four rows) and our models (the middle two rows), considering only the five most popular instruments as [Hung *et al.*, 2019]. Table 1 shows that the proposed UnetED model can achieve better AUC scores in most instruments and achieve the highest average AUC 0.829 overall, while the performance of DuoED is on par with [Gururani *et al.*, 2018]. We also conduct a paired t-test and found that the performance difference between Gururani's method and the proposed UnetED method is statistically significant (p-value$<$0.05). This result demonstrates the effectiveness of the learned disentangled timbre representation $\mathbf{Z}_t$ for IAD, compared to conventional representations such as the log mel-spectrogram used by [Gururani *et al.*, 2018]. Moreover, we note that these prior arts on IAD cannot work for composition style transfer, while our models can.

The last two rows of Table 1 show the result of two existing disentanglement methods originally developed for images [Liu *et al.*, 2018a; Hadad *et al.*, 2018].[4] We use encoder/decoder structures similar to our models and the same strategy to pre-train on MuseScore and then optimize the timbre classifier on the M&M training split. We see that the proposes models still outperform these two models. This can be expected as their models were not designed for music.

Table 2 reports an ablation study where we do not use the M&M training split to learn D'_t but use the original D_t trained on MuseScore for IAD. In addition, we compare the case with or without adversarial training, and the case where we get the instrument roll estimate from the output of D_{roll} [Hung *et al.*, 2019], which considers both pitch and timbre. We see that adversarial training improves IAD, and that adding pitch information helps. From Tables 1 and 2 we also see that it is helpful to update the instrument classifier by the training split of M&M, possibly due to the acoustic difference between synthetic and real-world audio recordings.

4.2 Evaluation on Music Rearrangement

To evaluate the result of music rearrangement, we choose four types of popular composition style in our experiment: **strings composition** (i.e., violin and cello), **piano**, **acoustic composition** (i.e., guitar, or guitar and melody), and **band composition** (i.e., electric guitar, bass, and melody); the melody can be played by any instrument. We randomly choose four clips in each type from the MuseScore dataset, and then transfer them into the other three types following the approach described in Section 3.4. We intend to compare the result of

instrument classifiers D_t of our models but train a new one D'_t, so as to make sure that our instrument classifier has the same architecture as that used by [Gururani *et al.*, 2018]. In this way, the major difference between our models and theirs is the input feature (the timbre code $\mathbf{Z}_t$ for ours, and the log mel-spectrogram for theirs), despite of some minor differences in for example the employed filter sizes.

[4]We adapt the two methods for music as follows. We employ the pianoroll as the target output for both. And, for [Liu *et al.*, 2018a], we treat 'S' as timbre and 'Z' as pitch; for [Hadad *et al.*, 2018], we replace 'style' with pitch and 'class' with timbre.

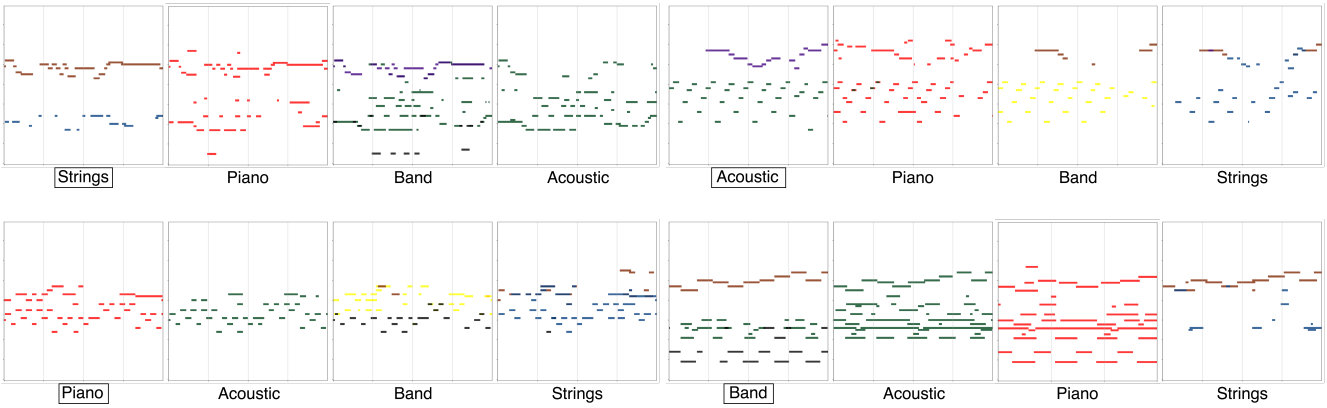


Figure 4: Demonstration of music rearrangement by UnetED (best viewed in color), for four types of styles: strings, piano, acoustic, and band (see Section 4.2 for definitions). The source clips are those marked by bounding boxes, and the generated ones are those to the right of them. [**Purple**: flute, **Red**: piano, **Black**: bass, **Green**: acoustic guitar, **Yello**: electric guitar, **Blue**: cello, **Brown**: violin].

UnetDE, DuoDE, UnetDE w/o Adv, and the method proposed by [Hadad *et al.*, 2018]. We treat the last method as the baseline, as it is designed for image style transfer, not for music.

We invite human subjects to listen to and rate the rearrangement result. Each time, a subject is given the source clip, and the four rearranged results by different models, all aiming to convert the style of that clip to one of the other three. The subject rate the rearranged results in the following three dimensions in a four-point Likert scale:

– whether the composition sounds *rhythmic*;

– whether the composition sounds *harmonic*;

– and, the *overall quality* of the rearrangement.

The scores are the higher the better. Since there is no ground truth of music style transfer, the rating task is by nature subjective. This process is repeated $4 \times 3 = 12$ times until all the source-target pairs are evaluated. Each time, the ordering of the result of the four modes are random.

We distributed the online survey through emails and social media to solicit voluntary, non-paid participation. 40 subjects participated in the study, 18 of which have experience in composing music. Figure 3 shows the average result across the clips and the subjects. It shows that, while DuoED and UnetED perform similarly for IAD, UnetED performs better in all the three metrics for music rearrangement. Moreover, with the adversarial training, the UnetED can generate more rhythmic and harmonic music than its ablated version. And, [Hadad *et al.*, 2018] does not work well at all.[5] Moreover, we found that there is no much difference between the ratings from people with music background and people without music background. The only difference is that people with music background averagely tends to give slightly lower ratings.

Figures 4 shows examples of the rearrangement result by UnetED. We can see that when rearranging the music to be played by band, UnetED finds the low-pitched notes for the bass to play and the melody notes for the flute or violin to play. This demonstrates that UnetED has the ability to choose suitable instruments playing certain notes combination. This result is also showed in arranging string instruments. More demo results of UnetED can be found at https://biboamy.github.io/disentangle_demo/.

[5]The poor result of [Hadad *et al.*, 2018] can be expected, since the pre-trained instrument decoder can only guarantee Z_t to contain timbre information, but cannot restrict Z_t not to have any pitch information. We observe that this method tend to lose much pitch information in the rearranged result and create a lot of empty tracks.

5 Conclusion

In this paper, we have presented two models for learning disentangled timbre representations. This is done by using the instrument and pitch labels from MIDI files. Adversarial training is also employed to disentangle the timbre-related information from the pitch-related one. Experiment show that the learned timbre representations lead to state-of-the-art accuracy in instrument activity detection. Furthermore, by modifying the timbre representations, we can generate new score rearrangement from audio input.

In the future, we plan to extend the instrument categories and also include the singing voice to cope with more music genres. We want to test other combinations of instruments to evaluate the performance of our models. We also want to further improve our models by re-synthesizing the MP3 files of the MuseScore data with more realistic sound fonts.

References

[Bittner *et al.*, 2017] Rachel M. Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan P. Bello. Deep salience representations for $f0$ tracking in polyphonic music. In *Proc. Int. Soc. Music Information Retrieval Conf.*, 2017.

[Briot *et al.*, 2017] Jean-Pierre Briot, Gaëtan Hadjeres, and François Pachet. Deep learning techniques for music generation: A survey. *arXiv preprint arXiv:1709.01620*, 2017.

[Brunner *et al.*, 2018] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer. MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer. *arXiv preprint arXiv:1809.07600*, 2018.

[Corozine, 2015] Vince Corozine. *Arranging Music for the Real World*. Mel Bay Publications, 2015.

[Crestel *et al.*, 2017] Léopold Crestel, Philippe Esling, Lena Heng, and Stephen McAdams. A database linking piano and orchestral MIDI scores with application to automatic projective orchestration. In *Proc. Int. Soc. Music Information Retrieval Conf.*, pages 592–598, 2017.

[Dai *et al.*, 2018] Shuqi Dai, Zheng Zhang, and Gus Xia. Music style transfer issues: A position paper. *arXiv preprint arXiv:1803.06841*, 2018.

[Dong *et al.*, 2018a] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proc. AAAI Conf. Artificial Intelligence*, 2018.

[Dong *et al.*, 2018b] Hao-Wen Dong, Wen-Yi Hsiao, and Yi-Hsuan Yang. Pypianoroll: Open source Python package for handling multitrack pianoroll. In *Proc. Int. Soc. Music Information Retrieval Conf.*, 2018. Late-breaking paper.

[Engel *et al.*, 2017] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with wavenet autoencoders. In *Proc. Int. Conf. Machine Learning*, pages 1068–1077, 2017.

[Gatys *et al.*, 2016] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[Goodfellow and others, 2014] Ian Goodfellow et al. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[Gururani and Lerch, 2017] Siddharth Gururani and Alexander Lerch. Mixing Secrets: A multi-track dataset for instrument recognition in polyphonic music. In *Proc. Int. Soc. Music Information Retrieval Conf.*, 2017. Late-breaking paper.

[Gururani *et al.*, 2018] Siddharth Gururani, Cameron Summers, and Alexander Lerch. Instrument activity detection in polyphonic music using deep neural networks. In *Proc. Int. Soc. Music Information Retrieval Conf.*, 2018.

[Hadad *et al.*, 2018] Naama Hadad, Lior Wolf, and Moni Shahar. A two-step disentanglement method. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 772–780, 2018.

[Hung and Yang, 2018] Yun-Ning Hung and Yi-Hsuan Yang. Frame-level instrument recognition by timbre and pitch. In *Proc. Int. Soc. Music Information Retrieval Conf.*, pages 135–142, 2018.

[Hung *et al.*, 2018] Yun-Ning Hung, Yi-An Chen, and Yi-Hsuan Yang. Learning disentangled representations for timber and pitch in music audio. *arXiv preprint arXiv:1811.03271*, 2018.

[Hung *et al.*, 2019] Yun-Ning Hung, Yi-An Chen, and Yi-Hsuan Yang. Multitask learning for frame-level instrument recognition. In *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2019.

[Kaliakatsos-Papakostas *et al.*, 2017] Maximos Kaliakatsos-Papakostas, Marcelo Queiroz, Costas Tsougras, and Emilios Cambouropoulos. Conceptual blending of harmonic spaces for creative melodic harmonisation. *J. New Music Research*, 46(4):305–328, 2017.

[Lee *et al.*, 2018] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proc. European Conf. Computer Vision*, 2018.

[Liu *et al.*, 2018a] Yang Liu, Zhaowen Wang, Hailin Jin, and Ian Wassell. Multi-task adversarial network for disentangled feature learning. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 3743–3751, 2018.

[Liu *et al.*, 2018b] Yu Liu, Fangyin Wei, Jing Shao, Lu Sheng, Junjie Yan, and Xiaogang Wang. Exploring disentangled feature representation beyond face identification. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2080–2089, 2018.

[Liu *et al.*, 2019] Jen-Yu Liu, Yi-Hsuan Yang, and Shyh-Kang Jeng. Weakly-supervised visual instrument-playing action detection in videos. *IEEE Trans. Multimedia*, 2019.

[Lu *et al.*, 2019] Chien-Yu Lu, Min-Xin Xue, Chia-Che Chang, Che-Rung Lee, and Li Su. Play as you like: Timbre-enhanced multi-modal music style transfer. In *Proc. AAAI Conf. Artificial Intelligence*, 2019.

[McFee *et al.*, 2015] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proc. Python in Science Conf.*, 2015.

[Oquab *et al.*, 2015] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proc. Int. Computer Vision and Pattern Recognition*, pages 685–694, 2015.

[Pachet, 2016] François Pachet. A joyful ode to automatic orchestration. *ACM Trans. Intell. Syst. Technol.*, 8(2):18:1–18:13, 2016.

[Pati, 2018] Ashis Pati. Neural style transfer for musical melodies, 2018. [Online] https://ashispati.github.io/style-transfer/.

[Raffel and Ellis, 2016] Colin Raffel and Daniel P. W. Ellis. Optimizing DTW-based audio-to-MIDI alignment and matching. In *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 81–85, 2016.

[Roberts *et al.*, 2017] Adam Roberts, Jesse Engel, and Douglas Eck. Hierarchical variational autoencoders for music. In *Proc. NIPS Workshop on Machine Learning for Creativity and Design*, 2017.

[Ronneberger *et al.*, 2015] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proc. Int. Conf. Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.