Supervised Symbolic Music Style Translation Using Synthetic Data

Ondřej Cífka, Umut Şimşekli, Gael Richard

SUPERVISED SYMBOLIC MUSIC STYLE TRANSLATION USING SYNTHETIC DATA

Ondřej Cífka **Umut Şimşekli** **Gaël Richard**
LTCI, Télécom Paris, Institut Polytechnique de Paris
{ondrej.cifka, umut.simsekli, gael.richard}@telecom-paris.fr

ABSTRACT

Research on style transfer and domain translation has clearly demonstrated the ability of deep learning-based algorithms to manipulate images in terms of artistic style. More recently, several attempts have been made to extend such approaches to music (both symbolic and audio) in order to enable transforming musical style in a similar manner. In this study, we focus on *symbolic music* with the goal of altering the 'style' of a piece while keeping its original 'content'. As opposed to the current methods, which are inherently restricted to be unsupervised due to the lack of 'aligned' data (i.e. the same musical piece played in multiple styles), we develop the first fully *supervised* algorithm for this task. At the core of our approach lies a *synthetic data* generation scheme which allows us to produce virtually unlimited amounts of aligned data, and hence avoid the above issue. In view of this data generation scheme, we propose an encoder-decoder model for translating symbolic music *accompaniments* between a number of different styles. Our experiments show that our models, although trained entirely on synthetic data, are capable of producing musically meaningful accompaniments even for real (non-synthetic) MIDI recordings.

1. INTRODUCTION

Artistic style transfer has become a well-established topic in the computer vision literature and is becoming of increasing interest in other areas of computer science, especially music and natural language processing. More generally, we are dealing with a family of *style transformation* tasks, where the goal is to alter the *style* of a piece of data (e.g., an image, a musical piece, a document) while preserving – to some extent – its *content*. In the music domain, a solution to these problems would have exciting industrial applications, not only as a way to generate new music automatically (as an alternative to fully automatic music composition, which still seems to be a distant goal), but also as a tool for music creators, allowing them to easily incorporate new styles and ideas into their work.

In computer vision, the most popular task in this direction is *style transfer*, where the algorithm has two inputs: the 'content' image to transform and a 'style' image, bearing the style that we wish to impose on (or *transfer* to) the content image. On the other hand, work done on music so far has mostly focused on a different task, which we refer to as *style translation*. Contrary to style transfer, only the 'content' input is given, and the goal is to render it in a target style which is known in advance and usually *learned* from a large set of examples. Note that although this second task is often also referred to as 'style transfer' in the context of music and text generation, we claim that this conflicts with how the term is traditionally understood [11,13,38], and that the term 'translation' is more appropriate and in line with other prior work [17, 24, 28, 40].

The focus of our work is on the latter task, and more specifically, on *accompaniment style translation* for *symbolic music*. In particular, given a piece of music in a symbolic representation, our goal is to generate a new accompaniment for it in a different arrangement style while preserving the original harmonic structure. Even though our approach is generic, to narrow down our scope, we focus on generating bass and piano tracks.

A major difficulty of the music style translation task is that there are no publicly available 'aligned' or 'parallel' datasets (containing examples of the same music played in different styles). As a result, recent works closely related to ours [4, 5] have adopted unsupervised learning frameworks – variational autoencoders (VAE) [19] and CycleGANs [40] – and applied them to genre-labeled datasets. However, these extensions to symbolic music have not yet permitted to obtain results as compelling as those on images [22, 40], text [20, 39], and music audio [28].

In this study, we adopt a different strategy to overcome the lack of aligned data, which is to *synthesize* it. Synthetic training data has proven useful for music information retrieval tasks such as chord recognition [21] and fundamental frequency estimation [25, 32], and is also popular for tasks like semantic segmentation in computer vision [30, 36]. In our case, synthetic data opens up the possibility for supervised learning techniques known from the machine translation field. Moreover, it allows us to work with fine-grained style labels, as opposed to genre labels, which may be too vague or ambiguous for such purposes.

Our main contributions are as follows:

- We propose a supervised, end-to-end neural model for symbolic music style translation, along with a

training data generation scheme.

- Our model is able to translate into a large number of different styles by conditioning a single decoder on the target style. To our knowledge, this is the first time this technique has been applied to music translation with some success.
- To evaluate the performance of our model, we propose an objective metric of music style similarity.
- We show that an approach to music style translation based entirely on synthetic data is viable and generalizes well to more 'natural' inputs, even in unrelated styles.

We believe that our approach will foster new directions in this line of research; some of these will be briefly discussed in the conclusion. The source code of our system, built using TensorFlow, is available online. [1]

2. RELATED WORK

The work performed so far in the area of music style transformation is relatively small in volume but fairly diverse, since, as noted in [8], the transformations can work with different music representations as well as on different conceptual levels.

To our knowledge, the only work on music style transfer – in the original sense, as discussed in the introduction – has been done on audio. Some approaches [9, 35] combine signal decomposition techniques with *musaicing* [41] (a form of concatenative synthesis). In [14], the authors attempt to transfer 'sound textures' from a recording by means of techniques adapted from image style transfer, but without specific focus on the musical aspects. In both cases, the transformation is largely limited to timbre.

The problem of unsupervised music audio translation is tackled in [28], where the authors train a neural network to translate between a number of domains. For symbolic music, style translation is studied in [4, 5], adapting unsupervised learning techniques from computer vision. A different approach is proposed in [23], consisting in training a model on the target style only and then using pseudo-Gibbs sampling to transform a given piece of music.

Finally, we should mention more 'constrained' problems from the symbolic music domain which can also be framed as style translation tasks, e.g. (re-)harmonization [16,29] and expressive performance generation [12,24,37].

3. SYNTHETIC DATA GENERATION

Since we are in a supervised setting, our approach requires a large amount of *paired* examples where each pair consists of one musical fragment arranged in two different styles. Given that no such dataset is currently available, we created a synthetic one, generated using RealBand from the Band-in-a-Box (BIAB) software package [2].

First, we downloaded chord charts of around 3.5K songs in the BIAB format from a popular online archive [3]. We used BIAB to generate arrangements of these songs in different styles and filtered the resulting MIDI files to keep only those in $\frac{4}{4}$ or $\frac{12}{8}$ time. [2] We then chopped those files into segments of 8 bars, splitting notes that overlap segment boundaries.

We selected a total of 70 styles from the '0 MIDI' and '1 MIDI' style packs included in Band-in-a-Box 2018, representing a wide variety of popular music genres. Each style contains up to 5 accompaniment tracks (drums, bass, piano, guitar, strings). [3] We generated each song in 3 randomly picked styles, providing $2 \times \binom{3}{2} = 6$ training pairs per segment, or around 658K training examples in total. An example of a possible training pair is shown in Fig. 1.

In all experiments, we used 2,809 songs for training, 46 songs as a validation set and 46 songs for evaluation, each in 3 examples in different styles. The song names, along with the styles used for each song, are included in the supplementary material [7].

4. PROPOSED MODEL

We propose an architecture based on RNN encoder-decoder sequence-to-sequence models with attention [1], commonly employed in machine translation and other areas of natural language processing. This choice is motivated by the successes of RNNs on symbolic music generation [10, 15, 33, 34] and by the ability of the attention mechanism to condition the generation on arbitrary input data without a prior alignment.

Our model is designed so that it is capable of translating music between a potentially large number of different styles. This is achieved by conditioning the decoder on the target style. An obvious advantage of this design is efficiency: to translate between n styles, we only need to train a single model, compared to n models (one for each target style; possibly with a shared encoder as in [28]) or even $\Theta(n^2)$ models (one for each pair of styles, e.g. [4,5]). Other implications of this choice are investigated in Section 6.2.

On the other hand, to simplify the task and facilitate evaluation, we train a dedicated model for each target instrument track. Our output representation and decoder architecture are chosen accordingly and would not necessarily be suitable for generating several independent tracks.

Input and output representation. A common choice of representation of symbolic non-monophonic music for neural processing is a piano roll. We use a binary-valued piano roll with 128 pitches and 4 columns per beat (quarter note) to encode our input.

For representing the output (and also as an alternative input representation), we opted for a MIDI-like encoding, which – unlike a piano roll – is straightforward to model using an RNN decoder. Specifically, following [33], we encode the music as a sequence of 3 types of events, each with one integer argument:

- `NoteOn(pitch)`: start a new note at the given pitch;

[1] `https://git.io/musicstyle`

[2] The time signature depends on the style as well as on the song itself. A song originally in $\frac{4}{4}$ may have a $\frac{12}{8}$ arrangement and vice versa.

[3] These 5 labels are not always accurate; for example, some styles have two guitar tracks, one of which is labeled as piano.

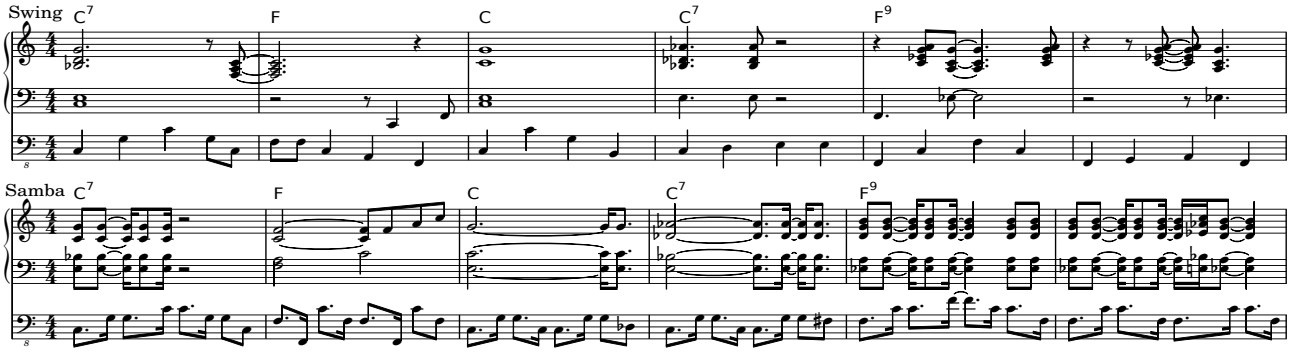
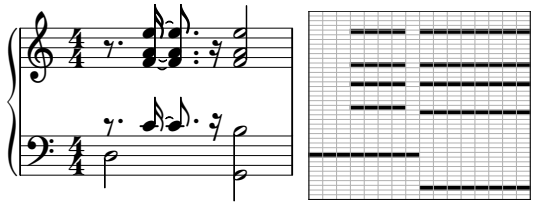


Figure 1: Six bars of an accompaniment (piano and bass) for a 12-bar blues, generated using BIAB in a 'jazz swing' style (top) and a 'samba' style (bottom). The timing is only approximate. The input chord sequence is displayed at the top.



```
NoteOn(50) TimeShift(9) NoteOn(60) NoteOn(65)
NoteOn(69) NoteOn(76) TimeShift(12) NoteOff(60)
NoteOff(65) NoteOff(69) NoteOff(76) TimeShift(3)
NoteOff(50) NoteOn(43) NoteOn(59) NoteOn(65)
NoteOn(69) NoteOn(76) TimeShift(24) NoteOff(All)
```

Figure 2: A bar of music, represented as a piano roll (top right) and as a sequence of 20 event tokens (bottom).

- `NoteOff(pitch)`: end the note at the given pitch;
- `TimeShift(delta)`: move forward in time by the specified amount, measured in 12ths of a beat.

`NoteOn` and `NoteOff` take values in the range 0–127, whereas `TimeShift` is within 1–24. [4] In contrast to [33], our representation is tempo-invariant and we do not model dynamics. Fig. 2 illustrates both representations.

Model architecture and training. The proposed model consists of an encoder and a decoder; the former serves to compute a dense representation of the input, while the latter generates the output event sequence, conditioned on the encoded input and the target style.

The architecture of the encoder depends on the type of input representation:

- If the input is a piano roll, we use a two-layer convolutional network (CNN), followed by a bidirectional RNN with a gated recurrent unit (GRU) [6]. The CNN serves to compress the input, resulting in a sequence of 1280-dimensional vectors with 2 vectors per bar. The bidirectional GRU then adds the ability to incorporate information from a wider context.
- If the input is a sequence of tokens, we use an embedding layer, also followed by a bidirectional GRU.

We refer to the two variants of the model as 'roll2seq' and 'seq2seq', respectively.

The decoder is also implemented using a GRU, conditioned on the target style and equipped with a feed-forward

[4] When encoding the piano track, we compress the sequences by also including a `NoteOff(All)` event which ends all currently active notes.

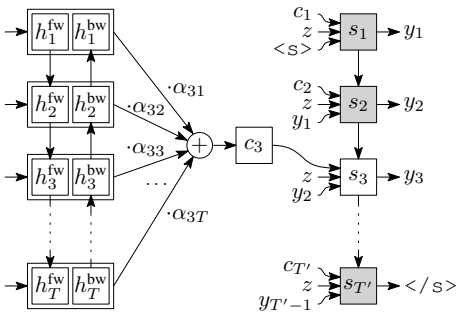


Figure 3: The attention-based decoder. During the i-th decoding step (here $i = 3$), a set of coefficients α_{ij} is computed and used to weight the encoder states $h_j = [h_j^{\text{fw}}, h_j^{\text{bw}}]$ to obtain the context vector c_i, which in turn is used as input for the decoder cell to compute the next state, s_i.

attention mechanism [1] acting on the encoder outputs. More precisely, as illustrated in Fig. 3, the i-th decoder state s_i is computed as

$$s_i = \text{GRU}([c_i, W^{\text{s}} z, W^{\text{e}} y_{i-1}], s_{i-1}),$$

where $[\cdot]$ denotes concatenation, z and y_{i-1} respectively denote the one-hot encoded representations of the target style and the previous output event, $W^{\text{s}}, W^{\text{e}}$ are the corresponding embedding matrices, and c_i is the context vector. The latter is a weighted average of the encoder outputs, computed by the attention mechanism. The purpose of attention is to provide an *alignment* between the encoder and decoder states. The need for this alignment arises from the fact that the positions in the output sequence are not linear in time (due to the chosen encoding), and the decoder therefore needs to be able to move its focus flexibly over the input. For a complete description of attention, see [1].

The training pipeline is portrayed in Fig. 4b. Each training example consists of a song segment x in one style (the source style) along with the corresponding segment y in a different style (z, the target style). We train the model by minimizing the loss on y while passing x to the encoder and conditioning the decoder on z.

The models are trained using Adam [18] with learning rate decay and with early stopping on the development set. Our configuration files with complete hyperparameter set-

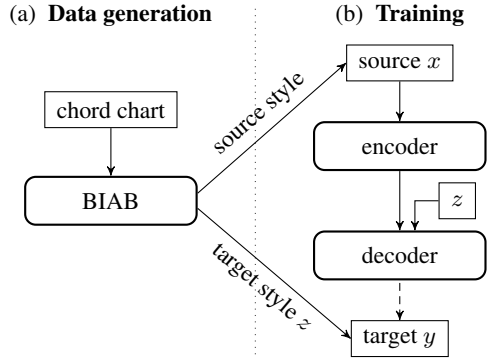


Figure 4: A scheme of the training pipeline. (a) We use BIAB to generate each song in different arrangement styles (see Section 3). (b) The model is trained to predict the target-style segment y given a source segment x and the target style z (see Section 4).

tings are included with the source code.

Once the model is trained, we perform style translation using greedy decoding, i.e. by taking the most likely output token at every step (and using that as input in the next step). We also explored random sampling with different softmax temperatures, but found that this leads to a higher number of errors (i.e. invalid sequences or incorrect timing) and does not significantly improve the quality of the outputs.

5. EVALUATION METRICS

When evaluating a style transformation, we need to consider two complementary criteria: how well the transformed music fits the desired style (*style fit*) and how much content it retains from the original (*content preservation*). Note that it is trivial (but useless) to achieve perfect results on either of these two criteria *alone*, so it is essential to evaluate both of them.

In this section, we describe 'objective', automatically computed metrics for both criteria. Even though we believe these metrics are sound and well-motivated, we acknowledge the limitations of automatic metrics in general and encourage the reader to listen to the provided example outputs [7] to get a real sense of their quality.

Content preservation. We use a content preservation metric similar to the one proposed by [23], computed by correlating the chroma representation of the generated segment with that of the corresponding segment in the source style. This is motivated by the fact that we expect the output to follow the same sequence of chords as the input. More precisely, we compute chroma features for each segment at a rate of 12 frames per beat and smooth each of them using an averaging filter with a window size of 2 beats (24 frames) and a stride of 1 beat (12 frames). Finally, we calculate the average frame-wise cosine similarity between the two sets of chroma features.

Style fit. In some of the recent music style transformation works [4,5], the quality of a transformation is measured by means of a binary style classifier trained on a pair of styles. However, the merit of such evaluation is limited, since a high classifier score merely demonstrates that the output has some of the distinguishing features of the target style, and not necessarily that it actually fits the style. For this reason, we aim for a more interpretable metric of style fit.

As observed by [16, 26, 31], musical style is well captured in pairwise statistics between neighboring events. Drawing inspiration from the features proposed in [26], we devise a key- and time-invariant style representation which we call the *style profile*.

To compute the style profile, we consider all pairs of note onsets less than 4 beats apart and at most 20 semitones apart, and record the time difference and interval for each pair. In other words, we define the following multiset of ordered pairs:

$$\mathcal{S} = \{(t_b - t_a,\ p_b - p_a) \,|\, a, b \in \text{notes},\ a \neq b,\\ 0 \leq t_b - t_a < 4,\ |p_b - p_a| \leq 20\},$$

where t_x is the onset time of the note x (measured in fractional beats) and p_x is its MIDI note number. We then obtain the style profile as a normalized 2D histogram of $\mathcal{S}$ with 6 bins per beat and one bin per semitone, and flatten it to get a 984-dimensional vector.

Finally, to quantify the style fit of a particular set of outputs, we compute their style profile and measure its cosine similarity to a reference profile. Note that an 8-bar segment may not be sufficient to obtain a reliable style profile; instead, we always aggregate the statistics over a number of segments. In particular, we put forward two variants of the style fit metric, obtained as follows:

(a) Compute a style profile aggregated over all outputs of a model in a given target style and measure its cosine similarity to the reference.

(b) Compute a style profile for each translated song separately and measure its cosine similarity to the reference. We report the mean and standard deviation over all songs.

We refer to (a) and (b) as 'macro-style' and 'song-style', respectively. In both cases, the reference style profile is extracted from the training set, separately for each track.

While we do not claim that this metric is able to distinguish between broad style categories (such as genres), yet it can definitely capture the differences and similarities between specific 'grooves', which makes it well-suited for our purpose. This is illustrated in Fig. 5, showing the pairwise similarities between the profiles of the bass tracks of different BIAB styles, with clearly visible clusters of jazz, rock or country styles.

6. EXPERIMENTAL RESULTS

In our experiments, we focus on generating the bass and piano tracks, and we train a dedicated model for each of them. For each track, we consider two scenarios: generating the track given only the corresponding source track (BASS→BASS, PIANO→PIANO), and using all non-drum accompaniment tracks from the input (ALL→BASS, ALL→PIANO).

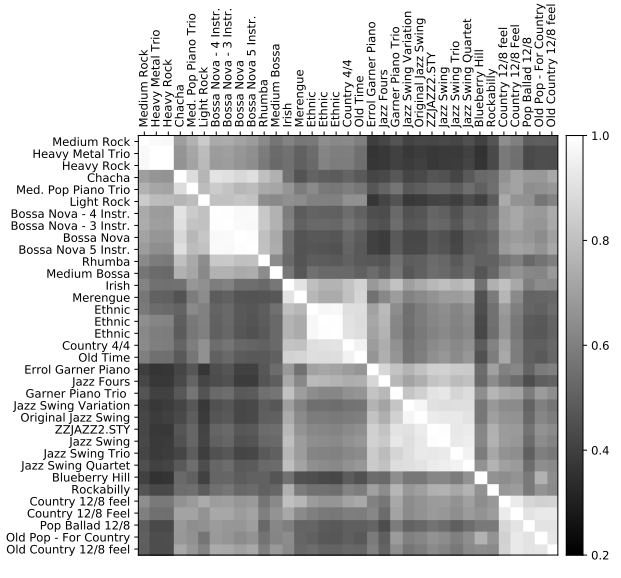


Figure 5: Pairwise cosine similarities of selected style profiles computed on training bass tracks. The styles are ordered based on a hierarchical clustering of the profiles.

For BASS→BASS, we compare the seq2seq and roll2seq architectures defined in Section 4. For all other pairs, where the input is non-monophonic, we only employ roll2seq, since the sequential representation grows disproportionately in length in these cases and the computational cost of the attention mechanism becomes too heavy.

We evaluate our models on our synthetic test set generated by BIAB and on the Bodhidharma MIDI dataset [27]. The latter is a diverse collection of 950 MIDI recordings annotated with genre labels. We filtered and pre-processed the dataset in the same way as the synthetic test set and we extracted the bass and piano tracks. [5]

We also made extensive attempts to train the recent models of [4, 5, 23] on our data using the source code published by the authors, but unfortunately without success. This has prevented us from comparing these models with our proposal. Nonetheless, the provided outputs [7] can serve as a basis for perceptual comparison.

6.1 Evaluation

For a comprehensive evaluation of each model, we translated all inputs to all 70 styles and calculated the content preservation and style fit metrics. The results (averaged) are presented in Fig. 6.

We provide two baselines for each track (bass and piano): 'source', which is simply the same track before the translation, and 'reference', which is a track generated by BIAB based on the chord chart (only available for the synthetic test set). As expected, the style fit is low for the source track (measured with respect to the target style) and close to 1 for the reference track. Our models' outputs generally do not fit the target style as perfectly as the reference does, but still score high compared to the source.

[5] To form the bass track, we retrieve all notes assigned to any Bass instrument. For the piano track, we use the Piano and Organ classes.

As for content preservation, we can notice that the reference value is quite low (0.78 for BASS and 0.79 for PIANO). This should not be too surprising, since we are comparing accompaniments in two different styles, which might have different pitch-class distributions; moreover, there is some random harmonic variation within each style (see e.g. bars 5–6 in Fig. 1). The results achieved by our models on the synthetic test set are very close to the reference. To illustrate the value range of the metric, we provide the results obtained by a 'randomized' baseline (shown as 'random' in Fig. 6), where we randomly permuted the reference segments for each style (obtaining a reference with the correct style, but the wrong content). The resulting value is very low (0.16 for BASS and 0.31 for PIANO) compared both to the true reference and to our models, indicating that the metric is useful and the models are performing well.

On Bodhidharma, content preservation is generally weaker than on the synthetic test set. One interpretation can be that the encoder simply fails to extract the content information accurately, since it was trained on a different domain. However, we also find that the models often make timing errors on Bodhidharma inputs, leading to misalignment between the input and the output, which may also cause the content preservation metric to drop.

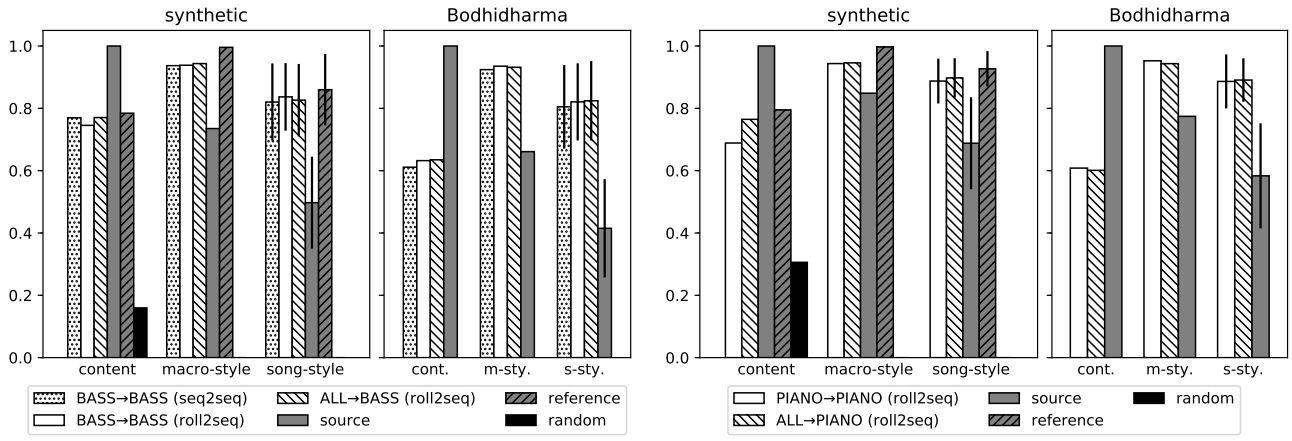
On the other hand, the style fit on Bodhidharma is close to the results on the synthetic test set (and not consistently lower or higher), and the difference to 'source' (i.e. the corresponding input track) is more marked, perhaps reflecting a higher style variability in the Bodhidharma data.

Upon listening, we clearly observe that the outputs are musical and seem to both fit the target style and follow the harmonic structure of the inputs. Besides, even though the piano and the bass tracks are generated independently, they sound surprisingly coherent. However, as mentioned above, we also observe occasional timing errors (especially in heavily syncopated grooves), which become more prominent when the bass and piano tracks are combined. A potential remedy for this issue would be to modify the encoding to make it more robust, e.g. by representing the timing in a beat-aware manner.

We also note that the single-track models output harmonically incorrect notes more often than the ALL models; this is expected, since their *input* is less harmonically rich. This effect is clearly audible (especially in BASS, where important scale degrees are often missing in the input), but cannot be captured by the content preservation metric, which is computed against the same input.

6.2 Comparison with a single-pair model

All models presented so far were trained on music in 70 different styles, as opposed to a single style pair. To investigate the effect of this choice, we picked a pair of fairly dissimilar styles – ZZJAZZSW ('Jazz Swing Variation') and TWIST ('Twist Style', categorized as 'Lite Pop') – and generated a new training, validation and test set with each song rendered in these two styles only. To increase the amount of data, we performed this twice for each song (with different results), obtaining $2 \times 2 = 4$ training pairs



(a) BASS (b) PIANO

Figure 6: Evaluation results on content preservation and style fit. 'Source' is the original track (bass or piano), 'reference' is a track generated by BIAB in the target style and 'random' is a random permutation of the references. For 'song-style', we plot the mean and the standard deviation over all songs and target styles.

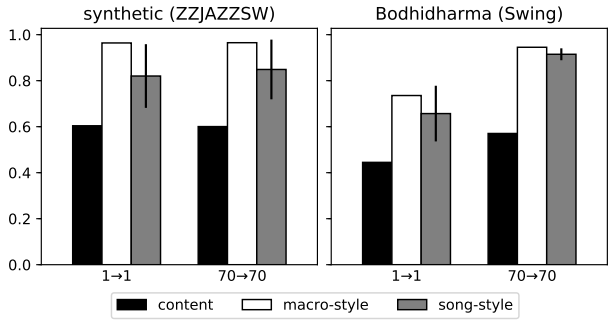


Figure 7: Comparison of a single-style-pair model (1→1) and a full model (70→70) on the ZZJAZZSW→TWIST style pair.

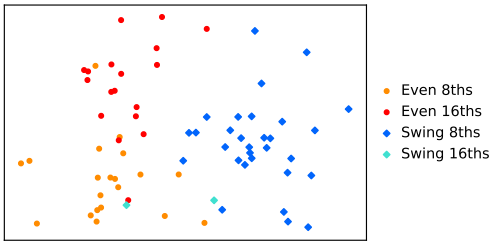


Figure 8: Style embeddings learned by the ALL→PIANO model, labeled with 'feel' annotations provided by BIAB. Dimensionality reduction was performed using linear discriminant analysis (LDA) with the feel labels as targets.

per segment.

We used this new dataset to train single-style-pair versions of all models (in the ZZJAZZSW→TWIST direction only), preserving the original architectures except for the conditioning on the target style. We compare these '1→1' models to the full versions (70→70) on two sets of inputs:

- the synthetic test set in the ZZJAZZSW style;
- the 'Swing' section of Bodhidharma (23 songs).

In Fig. 7, we show the results for the two variants of the ALL→BASS model. While the performance on the synthetic data seems to be the same, the scores of the 1→1 model drop considerably on the Bodhidharma data, suggesting that the model is overfitted to the 'synthetic' swing style. On the other hand, the performance of the 70→70 model stays high, showing that training on many different styles helped the model generalize to real swing.

6.3 Style embedding analysis

Neural representation spaces are often found to exhibit a meaningful geometry, and our learned style embedding space is no exception. As an example, Fig. 8 shows a projection of the embeddings labeled by the 'feel' of each style, with 'even' and 'swing' feel styles being clearly separated. We include more plots in the supplementary material and also make available an interactive visualization. [6]

7. CONCLUSION

In this study, we focused on symbolic music accompaniment style translation. As opposed to the current methods, which are inherently restricted to be unsupervised due to the lack of aligned datasets, we developed the first fully supervised algorithm for this task, leveraging the power of synthetic training data. Our experiments show that our models are capable of producing musically meaningful accompaniments even for real MIDI recordings.

We believe that these results point to interesting research directions. First, synthetic data seem to be an excellent resource for music style translation, and could be used as a starting point even for unsupervised learning, allowing to validate a given approach before moving on to more challenging, unaligned datasets. Second, our supervised approach could be used to address more general music transformation tasks, and we are already working on an extension in this direction.

[6] https://bit.ly/2G5Jgnq

8. ACKNOWLEDGEMENT

This research is supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 765068 (MIP-Frontiers) and by the French National Research Agency (ANR) as a part of the FBIMATRIX (ANR-16-CE23-0014) project.

9. REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[2] Band-in-a-Box. PG Music Inc. https://www.pgmusic.com/.

[3] Band-in-a-Box (BIAB) file archive. https://groups.yahoo.com/group/Band-in-a-Box-Files/.

[4] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer. MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer. In *ISMIR*, 2018.

[5] Gino Brunner, Yuyi Wang, Roger Wattenhofer, and Sumu Zhao. Symbolic music genre transfer with CycleGAN. *CoRR*, abs/1809.07575, 2018.

[6] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, 2014.

[7] Ondřej Cífka. Supplementary material: Supervised symbolic music style translation using synthetic data. Zenodo, 2019. https://doi.org/10.5281/zenodo.3250606.

[8] Shuqi Dai, Zheng Zhang, and Gus Xia. Music style transfer: A position paper. *CoRR*, abs/1803.06841, 2018.

[9] Jonathan Driedger, Thomas Prätzlich, and Meinard Müller. Let it Bee – towards NMF-inspired audio mosaicing. In *ISMIR*, 2015.

[10] Douglas Eck and Jürgen Schmidhuber. Finding temporal structure in music: blues improvisation with LSTM recurrent networks. In *NNSP*, 2002.

[11] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, 2001.

[12] Sebastian Flossmann and Gerhard Widmer. Toward a multilevel model of expressive piano performance. In *ISPS*, 2011.

[13] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, pages 2414–2423, 2016.

[14] Eric Grinstein, Ngoc Q. K. Duong, Alexey Ozerov, and Patrick Pérez. Audio style transfer. In *ICASSP*, pages 586–590, 2018.

[15] Gaëtan Hadjeres and François Pachet. DeepBach: a steerable model for Bach chorales generation. In *ICML*, 2017.

[16] Gaëtan Hadjeres, Jason Sakellariou, and François Pachet. Style imitation and chord invention in polyphonic music with exponential families. *CoRR*, abs/1609.05152, 2016.

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, pages 5967–5976, 2017.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[19] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. *CoRR*, abs/1312.6114, 2014.

[20] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *CoRR*, abs/1711.00043, 2017.

[21] Kyogu Lee and Malcolm Slaney. Acoustic chord transcription and key extraction from audio using key-dependent hmms trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16:291–301, 2008.

[22] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[23] Wei-Tsung Lu and Li Su. Transferring the style of homophonic music using recurrent neural networks and autoregressive models. In *ISMIR*, 2018.

[24] Iman Malik and Carl Henrik Ek. Neural translation of musical style. *CoRR*, abs/1708.03535, 2017.

[25] Matthias Mauch and Simon Dixon. PYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *ICASSP*, pages 659–663, 2014.

[26] Cory McKay. Automatic genre classification of MIDI recordings. M.A. Thesis, McGill University, 2004.

[27] Cory McKay and Ichiro Fujinaga. The Bodhidharma system and the results of the MIREX 2005 symbolic genre classification contest. In *ISMIR*, 2005.

[28] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman. A universal music translation network. *CoRR*, abs/1805.07848, 2018.

[29] François Pachet and Pierre Roy. Non-conformant harmonization: the Real Book in the style of Take 6. In *ICCC*, 2014.

[30] Germán Ros, Laura Sellart, Joanna Materzynska, David Vázquez, and Antonio M. López. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, pages 3234–3243, 2016.

[31] Jason Sakellariou, Francesca Tria, Vittorio Loreto, and François Pachet. Maximum entropy models capture melodic styles. *Scientific Reports*, 2017.

[32] Justin Salamon, Rachel M. Bittner, Jordi Bonada, Juan J. Bosch, Emilia Gómez, and Juan Pablo Bello. An analysis/synthesis framework for automatic F0 annotation of multitrack datasets. In *ISMIR*, 2017.

[33] Ian Simon and Sageev Oore. Performance RNN: Generating music with expressive timing and dynamics. Magenta Blog, 2017. `https://magenta.tensorflow.org/performance-rnn`.

[34] Bob L. Sturm, João Felipe Santos, Oded Ben-Tal, and Iryna Korshunova. Music transcription modelling and composition using deep learning. *CoRR*, abs/1604.08723, 2016.

[35] Christopher J. Tralie. Cover song synthesis by analogy. In *ISMIR*, 2018.

[36] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *CVPR*, pages 4627–4635, 2017.

[37] Gerhard Widmer, Sebastian Flossmann, and Maarten Grachten. YQX plays Chopin. *AI Magazine*, 30:35–48, 2009.

[38] Xuexiang Xie, Feng Tian, and Seah Hock Soon. Feature guided texture synthesis (FGTS) for artistic style transfer. In *DIMEA*, 2007.

[39] Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. Adversarially regularized autoencoders. In *ICML*, 2018.

[40] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2242–2251, 2017.

[41] Aymeric Zils and François Pachet. Musical mosaicing. In *COST G-6 Conference on Digital Audio Effects (DAFX-01)*, 2001.