Improving Polyphonic Music Models with Feature-Rich Encoding

Omar Peracha

Humtap, Inc.†

INTRODUCTION

Recent years have seen great progress in computational models' ability to capture the semantics of a musical corpus. Much of this progress is due to advances in artificial neural network algorithms and their application to the music domain.

Some of the most significant breakthroughs of late have experimented with:

- Applying newly-developed architectures to the problem of modelling symbolic music
- Training or pre-training on a large cross-domain corpus
- Introducing Gibbs-like sampling methods to orderless models

We instead focus on the data representation being modelled itself. We provide a simple improvement that might benefit results across a wide range of approaches.

We run experiments on the JSB chorales dataset, and obtain **state-of-the-art results** with a simple recurrent neural network based model.

CODE

All code for this paper is made publicly available for reproducibility, including the ability to evaluate pre-trained models and train a model from scratch: https://github.com/omarperacha/TonicNet

METHOD & RESULTS

We formulate the problem as a sequence modelling task. As is common with this JSB chorales dataset, we consider each semiquaver to be one time-step, and we encode the pitch of each voice at that time-step in a serialised order. We therefore represent an entire piece as a linear sequence in the following way, where N is the total number of time-steps in the piece:

$$S_0, A_0, T_0, B_0, S_1, A_1, T_1, B_1 \ldots S_{N-1}, A_{N-1}, T_{N-1}, B_{N-1}, \text{<END>}$$

The model is tasked with predicting each token, taking only the previous tokens of the given piece as input. Our simple contribution is to include the chord at each time-step in the sequence, tasking the model to first predict this chord before then predicting the actual pitches for each voice at that time-step, while maintaining the linear sequence structure and fully autoregressive prediction:

$$C_0, S_0, A_0, T_0, B_0, C_1, S_1, A_1, T_1, B_1 \ldots C_{N-1}, S_{N-1}, A_{N-1}, T_{N-1}, B_{N-1}, \text{<END>}$$

During preprocessing, we use existing tools to calculate the chords from the pitches at each time-step so we can then include these in the sequence being fed to the model. We augment the dataset by transposing each chorale as far as possible while ensuring no individual voice's range exceeds the limits seen in the unaugmented dataset. We evaluate our results on a holdout set, using the canonical train/validation/test split of 229, 76 and 77 samples respectively.

We run experiments with two architectures: a five-layer Transformer encoder with input masking and a ~1.25m parameter model based on the Gated Recurrent Unit, which we give nickname **TonicNet**. The models are trained using cross-entropy as the objective function. Both models' results improve when chords are included in the sequence, and in general seem to perform better when predicting sequences with more voice parts encoded, as opposed to seemingly simpler sequences with fewer voice parts.

In Table 1, each model variant shows the voice parts trained and evaluated on in parentheses, where NLL = Negative Log Likelihood, C = Chords, S = Soprano, A = Alto, T = Tenor and B = Bass. NCL stands for No Chord Loss, indicating that the model was only evaluated on the note predictions, ignoring the loss at time-steps corresponding to chord predictions when averaging NLL. **TonicNet_Z** indicates the inclusion of a note-level repetition encoding as an extra conditioning input when training the model. **TonicNet_Z** with data augmentation obtains state-of-the-art results for the JSB chorales at 16th-note resolution, which corresponds to encoding the dataset in full with no lost values.

Model & Dataset Variation	Validation NLL
Transformer (SATB)	0.544
Transformer (CSATB)	0.503
Transformer (CSATB, NCL)	0.394
Music Transformer* (SATB)	0.335
COCONET* (SATB, chronological)	0.436
COCONET* (SATB, orderless)	$\leq$ 0.238
TonicNet (C)	0.936
TonicNet (B)	0.716
TonicNet (S)	0.521
TonicNet (CS)	0.588
TonicNet (SB)	0.555
TonicNet (CSB)	0.516
TonicNet (SATB)	0.523
TonicNet_Z (SATB)	0.497
TonicNet_Z (CSATB)	0.422
TonicNet_Z (CSATB, Tr)	0.321
TonicNet_Z (CSATB, Tr+MM)	0.317
TonicNet_Z (CSATB, Tr, NCL)	**0.224**
TonicNet_Z (CSATB, Tr+MM, NCL)	**0.220**

Table 1. Validation loss on JSB chorales at 16th-note time-steps.

REFERENCES

C. Payne, "MuseNet", OpenAI, 2019. [Online]. Available: https://openai.com/blog/musenet/. [Accessed: 5- May- 2019].

C. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. Dai, M. Hoffman, M. Dinculescu and D. Eck, "Music Transformer", arXiv:1809.04281 [cs], 2018 .

C. Donahue, H. Mao, Y. Li, G. Cottrell and J. McAuley, "LakhNES: Improving Multi-Instrumental Music Generation with Cross-Domain Pre-Training", arXiv:1907.04868 [cs], 2019.

G. Hadjeres, F. Pachet and F. Nielsen, "DeepBach: a Steerable Model for Bach Chorales Generation", in International Conference on Machine Learning, 2017, pp. 1362-1371.

C. Huang, T. Coojimans, A. Roberts, A. Courville and D. Eck, "Counterpoint by Convolution", in Proc. of the International Conference on Music Information Retrieval, 2017, pp. 211–218.

Contact: omar.peracha@gmail.com

†research conducted independently