ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

Calliope - A Polyphonic Music Transformer

Andrea Valenti, Stefano Berti and Davide Bacciu *

University of Pisa - Dept. of Computer Science
Largo B. Pontecorvo, 3, 56127 Pisa - Italy

Abstract. The polyphonic nature of music makes the application of deep learning to music modelling a challenging task. On the other hand, the Transformer architecture seems to be a good fit for this kind of data. In this work, we present Calliope, a novel autoencoder model based on Transformers for the efficient modelling of multi-track sequences of polyphonic music. The experiments show that our model is able to improve the state of the art on musical sequence reconstruction and generation, with remarkably good results especially on long sequences.

1 Introduction

Language and music are often considered to be characterized by an high degree of resemblance. For instance, both can be represented by sequences of symbols: words for text, notes for music. Those symbolic sequences present well-defined structures that are often articulated at different layers of abstraction. However, music poses additional challenges over text: unlike the latter, a musical sequences can contain chords instead of simple notes, with more than one symbol being simultaneously played. A song is also often the combination of a series of instruments, each defining a different sub-sequence of note symbols played at the same time. The polyphonic nature of music, both at note and track level, poses several challenges to recent deep learning models, making their application to music modelling a challenging task.

Notable models that tackle the problem of automatic music generation are the MusicVAE [1] and MusAE [2], both of which use a LSTM-based architecture to generate sequences of monophonic music. MuseGAN [3] combines generative adversarial networks (GAN) [4] with a convolutional architecture to model multi-track MIDI polyphonic music. However, results are limited to relatively short (4-bar) sequences. One of the first attempts to use Transformer architecture [5] for music is the Music Transformer [6]. The new architecture allows the model to generate longer single-track sequences of piano music. Transformer Autoencoder [7] use the Transformer to generate new songs in the style of a given performance. Despite the promising results, these models are only evaluated on simple datasets, only containing songs played on a piano with homogeneous stylistic characteristics.

In this work, we present Calliope, a novel deep learning model for the automatic generation of music. To the best of our knowledge, this is the first Transformer-based architecture for challenging multi-track polyphonic music reconstruction and generation, comprising songs from multiple artists and genres.

*This research was partially supported by H2020 TAILOR (GA 952215)

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

2 Calliope: Polyphonic Music with Transformers AAE

Calliope's architecture fits into the framework of Adversarial Autoencoders (AAE) [8]. That is a variational autoencoder [9] augmented with an additional network (the *Discriminator*) that performs the variational approximation of the posterior in an adversarial way. The model is trained to maximize the Evidence Lower BOund (ELBO): given a data sample x, the ELBO is defined as

$$\mathcal{L}(x, \theta, \phi) = \mathbb{E}_{q_\phi(z|x)} \left[p_\theta(x|z) \right] - \beta D_{KL} \left(q_\phi(z|x) || p(z) \right) \tag{1}$$

where $p_\theta(x|z)$ is the decoder's distribution, $q_\phi(z|x)$ is the encoder's distribution and $p(z)$ is the chosen prior distribution of the latents, β is an hyperparameter that controls the amount of desired regularization. Calliope's training algorithm comprises two main phases: the reconstruction phase and regularization phase. In the reconstruction phase, only the first term of Eq.1 is optimized. A batch of songs is fed in input to the encoder to get their respective latent encodings, which are then decoded back into the original songs. The model is trained to have the reconstructed songs to be as close as possible to the originals. The regularization phase optimizes the second term of Eq.1. The KL divergence is approximated with the aid of an additional discriminator network $d_\psi(z)$ using the density-ratio trick [10]. This induces a min-max adversarial game between the encoder $q_\phi(z|x)$ and the discriminator $d_\psi(z)$ similar to the one of GANs:

$$\min_\phi \max_\psi \mathbb{E}_{q_\phi(z|x)}[\log d_\psi(z)] + \mathbb{E}_{p(z)}[\log(1 - d_\psi(x))].$$

While the *Discriminator* is implemented as a multi-layer perceptron, the encoder and the decoders use purposely developed and tailored Transformer architectures [5], described in the following sections.

2.1 Data Representation

A note is represented as a tuple $(time, pitch, duration)$, where *time* denotes the note onset time (measured as the offset, in number of timesteps, from the start of the measure), *pitch* is the actual pitch played by the note, and *duration* is the number of timesteps during which the note keeps playing. Each element of this tuple is represented by a symbolic token, taken from the following vocabulary: 128 pitch values (tokens from 0 to 127), 96 time values (tokens from 128 to 223), 96 duration values (tokens from 224 to 319), pad (token 320), start-of-song and end-of-song (tokens 321 and 322). Before feeding the sequence in input to the encoder, the tokens are embedded in a vector of N_x element, which is learned during training along with the other model parameters.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

2.2 The Calliope Encoder

The encoder compresses the song x, into a single latent code z of dimension N_z. The encoder's forward pass is the following:

$$\begin{aligned} h_{i,t} &\leftarrow \text{Encoder}(x_{i,t}) \\ z_{i,t} &\leftarrow \text{Comp}(h_{i,t}) \\ z_i &\leftarrow \text{BarCompressor}([z_{i,t_1} ... z_{i,t_M}]) \\ z &\leftarrow \text{SongCompressor}([z_1 ... z_N]) \end{aligned}$$

where N is the song length, M the number of tracks, t is the track index and i is the measure index. First, x is split along the time axis into a series 1-measure sequences x_i. Then, each x_i is further split into each track that composes a song $x_{i,t}$. These single-track, single-measure sequences are finally encoded separately via the *Encoder* module, which is implemented as a Transformer with the addition of Relative Positional Encoding in the Self-Attention Layer [11]. A different Encoder for each instrument is used. Since the *Encoder* preserves the original dimensions of the input, we use the Comp module to compress $h_{i,t}$ along the time dimension in order to obtain a single $z_{i,t}$ for the measure. The $z_{i,t}$ of the individual tracks are then merged together: the *BarCompressor* concats the $z_{i,t}$ along the track dimension and passes them through a linear layer to get a single code for the multi-track measure z_i. Each z_i is further concatenated with the other codes of the sequence along the time dimension, and the resulting tensor is the fed to the *SongCompressor*, consisting in a combination of a linear and layernorm[5] layers, in order to get the final latent representation of the song z.

2.3 The Calliope Decoder

The decoder's task is specular to the encoder's. It processes a single latent vector z in order to generate a multi-track polyphonic musical sequence x:

$$\begin{aligned} [z_1, ... z_i, ... z_N] &\leftarrow \text{SongDecompressor}(z) \\ [z_{i,t_1} ... z_{i,t_M}] &\leftarrow \text{BarDecompressor}(z_i) \\ h_{i,t} &\leftarrow \text{Decomp}(z_{i,t}) \\ x_{i,t} &\leftarrow \text{Decoder}(h_{i,t}) \end{aligned}$$

where N, i and t are, again, the song length, the measure index and the track index respectively. First, z is sent into a *SongDecompressor* module that repeats z for the length of the sequence (in measures) and projects it through a linear layer to get a series of z_i, each of them responsible for encoding a single measure. The single-measure, single-track codes $z_{i,t}$ are computed from the z_i by the *BarDecompressor* module, that splits the z_i into equal parts (one for each track) and projects each part into the appropriate shape using a linear layer. The *Decomp* module restores the time dimension of the $z_{i,t}$, making them ready to be processed by the *Decoder*, which is again a Transformer. The output of the *Decomp* module is used as memory into the source attention layer of the decoder to guide the generation of the final song.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

	Calliope	**MusAE**	**MusicVAE**	**MusicVAE H**
16-bar drums (seq)	**0.961**	0.773	0.641	0.895
16-bar melody (seq)	**0.890**	0.710	0.660	0.760
2-bar drums (seq)	0.989	**0.999**	0.917	-
2-bar melody (seq)	0.967	**0.991**	0.951	-
16-bar drums (next)	**0.952**	-	0.884	0.928
16-bar melody (next)	**0.923**	-	0.919	0.919
2-bar drums (next)	**0.995**	-	-	0.979
2-bar melody (next)	**0.991**	-	-	0.986

Table 1: Reconstruction accuracy for Callipoe, MusAE, flat and hierarchical versions of Music VAE. (seq) is whole sequence reconstruction accuracy, while (next) is the next step prediction accuracy.

3 Experiments

We performed an experimental analysis to asses Calliope's capabilities using the Lakh MIDI Dataset [12], a collection of about 100k MIDI songs of various artists and genres. The model is trained using the Adam optimizer [13] with a learning rate of 10^{-4}. We trained three models with sequence length of 1-bar (96 timesteps), 2-bars (192 timesteps) and 16-bars (1536 timesteps), respectively. The embeddings dimension is 256. The *Encoder* and *Decoder* modules have 6 Transformer layers of size 512. We use 4 heads in the multi-head attention layer. The dimension of the latent space is 256. The batch size is 20 for the 1-bar and 2-bars models and 2 for the 16-bars model, due to memory limitation of the training hardware. The dataset has been split into 70% songs for training, 10% for validation and the remaining 20% for test. For decoding we use a scheduled sampling strategy [14] with K=1 and teacher forcing probability of 0.5. After 50k training steps for the 1-bar and 2-bars models and 25k steps for the 16-bars model, the value of β is gradually annealed from 0 to 0.1. In the following experiments, we chose the prior distribution $p(z)$ to be an isotropic standard Gaussian $\mathcal{N}(0, I)$. The source code can be found online[1]. Finally, recognizing that "writing about music is like dancing about architecture", we provide additional samples of the generated songs[2].

3.1 Sequence Reconstruction

The first set of experiments explores the auto-encoding abilities of the model. The reconstruction accuracy of our model is compared with the MusicVAE [1] and MusAE [2] models. Table 1 reports the results on the test set. In general, our model exhibits a higher reconstruction accuracy than its competitors, with only a slightly lower reconstruction accuracy than MusAE on shorts sequences.

[1] `https://www.github.com/StefanoBerti/MusAE`
[2] `https://stefanoberti.github.io/musae`

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

	EB				UPC		
	B	**D**	**G/P**	**S**	**B**	**G/P**	**S**
jamming	6.59	2.33	20.45	6.10	**1.53**	3.91	4.09
composer	0.01	28.9	1.35	0.01	2.51	4.55	5.19
hybrid	2.14	29.7	14.75	6.04	2.35	5.11	5.24
Calliope	**0.0**	**0.0**	**0.0**	**0.0**	2.08	**3.87**	**2.52**

	QN			DP
	B	**G/P**	**S**	**D**
jamming	71.5	59.6	63.1	93.2
composer	49.5	48.65	52.5	75.3
hybrid	44.6	44.35	52.0	71.3
Calliope	**99.0**	**96.15**	**96.21**	**94.84**

Table 2: Comparison of generation metrics between the jamming, composer and hybrid models of MuseGAN and Calliope.

This can be explained by the fact that we are now dealing with polyphonic tracks, while the other two models are capable of processing monophonic music only. The accuracy is significantly higher in the case of 16-bar sequences, showing that the new architecture is particularly suited for the processing of long-range dependencies in the input. From a qualitative point of view, the discrepancies between the reconstructions and the originals still preserve the tonal characteristics of the songs and generally do not negatively affect the quality of the reconstructed sequence in a significant way.

3.2 Song Generation

The second aspect that we wish to explore is the ability of our model to generate from scratch new musical sequences by decoding new latent codes directly sampled from the prior. Therefore, we decode a set of 20000 latent codes sampled directly from the prior and compute a set of metrics on them: 1) EB: ratio of empty measures. A high EB could indicate "holes" into the latent space and, in general, can be a symptom of problems during the training procedure. 2) UPC: number of used pitch classes per bar. Lower UPC means the the model is able to choose (and stick to) a specific tonality, reducing the probability of generating dissonant notes and chords. 3) QN: ratio of "qualified" notes. A note no shorter than three time steps is a qualified note. QN shows whether the music is overly-fragmented with poor or absent structure. 4) DP: drum patterns. Ratio of notes in 8 or 16-beat patterns, common ones for songs in 4/4 time. High DP shows that the model knows how to emphasize the correct timesteps of generated measures. Results are reported in Table 2, compared to the MuseGAN [3] model. Our model scores very well on all the considered metrics, showing that the generated sequences have everything it takes to be appealing for the human listener.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

4 Conclusion and Future Work

In this paper we presented Calliope, a novel Transformer-based architecture for the efficient modelling of multi-track polyphonic music. The experiments show that our model is able to improve the state of the art on musical sequence reconstruction and generation, with remarkably good results especially on long sequences. The generated songs are musically meaningful and have similar characteristics of human-generated music. In the future, we plan to increase the number of tracks that the model is able to process. Furthermore, we plan to add more controllable musical properties such as notes velocity, tempo and key changes.

References

[1] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning*, pages 4364–4373. PMLR, 2018.

[2] A. Valenti, A. Carta, and D. Bacciu. Learning style-aware symbolic music representations by adversarial autoencoders. In *ECAI 2020*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 1563–1570. IOS Press, 2020.

[3] H. Dong, W. Hsiao, L. Yang, and Y. Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[6] Y. Huang and Y. Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1180–1188, 2020.

[7] K. Choi, C. Hawthorne, I. Simon, M. Dinculescu, and J. Engel. Encoding musical style with transformer autoencoders. In *International Conference on Machine Learning*, pages 1899–1908. PMLR, 2020.

[8] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[9] D. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[10] L. Mescheder, S. Nowozin, and A. Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pages 2391–2400. PMLR, 2017.

[11] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[12] C. Raffel. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. PhD thesis, Columbia University, 2016.

[13] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] T. Mihaylova and A. Martins. Scheduled sampling for transformers. *arXiv preprint arXiv:1906.07651*, 2019.