

Composing like a human: Adapting generative networks to few-shot learning in the musical domain

Tudor Paisa

SNR: 2019551 | ANR: 315146

t.paisa@tilburguniversity.edu

Tilburg University

School of Humanities and Digital Sciences

Department of Cognitive Science & Artificial Intelligence

Tilburg, The Netherlands

Thesis Supervisor:

Dr. Menno van Zaanen

Author Note

Thesis proposal submitted in partial fulfillment of the requirements for the degree of Master of Science in Data Science and Society, at the School of Humanities and Digital Sciences of Tilburg University

Composing like a human: Adapting generative networks to few-shot learning in the musical domain

Introduction

The reemergence of neural networks (NN) has brought new ways to tackle Machine Learning problems (see LeCun, Bengio, and Hinton, 2015 for examples), as well as new possibilities to further artificial intelligence research (such as general-purpose AI systems that play video-games; Vinyals et al., 2017 respectively). Currently, two exciting areas of research in the field of Deep Learning (DL) are in designing generative models (Goodfellow, 2016) and training NNs on small datasets (Finn, Abbeel, & Levine, 2017; Nichol, Achiam, & Schulman, 2018). This study will combine both tasks by exploring the extent to which generative models can create novel and qualitative samples of music under the constraints posed by the few-shot learning problem. In other words, when data is scarce, can a generative model create samples of music that are new (i.e., not in the training set), consistent with common scales, and diverse (tone sequences are not repeated over and over, combines chord structures with single-note sequences, is not restricted to - say - one octave). This research will evaluate the generated samples of two generative models (C-RNN-GAN of Mogren, 2016 and Performance-RNN of Oore, Simon, Dieleman, Eck, and Simonyan, 2018) that have been trained under the Reptile learning framework (Nichol et al., 2018) on small amounts of MIDI data.

DL as a field, promises to discover rich hierarchical models (LeCun et al., 2015), over various artificial intelligence applications, such as speech recognition, image classification, or text generation. The majority of developments usually involved discriminative models (Goodfellow et al., 2014), where high-dimensional input is assigned to a class label (e.g., Krizhevsky, Sutskever, & Hinton, 2012). In contrast, successes in generative models have been scarce until the discovery of the Generative Adversarial Network (GAN) by Goodfellow et al. (2014). GANs have had significant successes in image super-resolution (Ledig et al., 2016), style-based image generation (Karras, Laine, & Aila, 2018), or image-to-image translation (such as a sketch into a photorealistic image; Isola, Zhu, Zhou, & Efros, 2016). These developments lie in the field of Computer Vision, but that is not to say that we can generate only images. GANs (and others) have been applied in Natural Language Processing where the scope was to generate high-quality text with sufficient diversity (e.g., L. Chen et al., 2018; Yu, Zhang, Wang, & Yu, 2016). Less explored, but still successful, have been the efforts in generating music (e.g., Dong, Hsiao, Yang, & Yang, 2017; Mogren, 2016). This paper will focus on the problem of generating music with the models proposed by Mogren (2016) and Oore et al. (2018), trained under the few-shot learning problem (detailed below).

It is fairly well known that DL models tend to be "data-hungry", a caveat imposed by its optimization algorithm (W.-Y. Chen, Liu, Kira, Wang, & Huang, 2018; Ravi & Larochelle, 2016). DL usually relies on a procedure called stochastic gradient descent (SGD) which implies computing for a given input vector the output and error, followed by calculating the input's average gradient and consequently adjusting the network's weights (LeCun et al., 2015). Ravi and Larochelle (2016) argue that the iterative nature of gradient-optimization algorithms does not allow them to perform well under the constraint of a set number of updates. Finally, they add by saying that for each new problem (e.g., new dataset), the model needs to reinitialize with a new set of random parameters which hurts the network's ability to converge to a good solution when constrained by a limited number of updates (Ravi & Larochelle, 2016).

Besides the promise of making DL algorithms easier to train, primary reasons why this issue needs to be addressed are as follows. Firstly, humans have the ability to generalize after one (or few) example(s) of a given object (Ravi & Larochelle, 2016; Vinyals, Blundell, Lillicrap, Kavukcuoglu, & Wierstra, 2016), something which DL algorithms do not. Secondly, this would alleviate the constant need for data collection to ensure reasonable performances. Finally, there are many fields where the data exhibits a large number of classes and few examples per class; approaching human-type learning could allow models to properly capture this sort of data (Larochelle, Finn, & Ravi, 2017; Ravi & Larochelle, 2016). This is known as the few-shot problem (Vinyals et al., 2016).

In spite of its proposed benefits, developments and experiments in few-shot learning are still scarce (Larochelle et al., 2017) and evaluations are largely concentrated on image data (e.g., W.-Y. Chen et al., 2018; Clouâtre & Demers, 2019; Lake, Salakhutdinov, & Tenenbaum, 2019). Few-shot experiments with generative networks are even less frequent (Clouâtre & Demers, 2019; Zhang, Che, Ghahramani, Bengio, & Song, 2018). This research aims to explore the viability of this learning approach in the context of generating music data. More concretely, in light of recent developments, I will evaluate two generative networks trained on MIDI data under the constraints of few-shot learning, and compare their performance to the baseline proposed by Larochelle et al. (2017) and properties extracted from real music performances.

Related Work

Lake, Salakhutdinov, and Tenenbaum (2015) popularized the idea of learning from a few class examples using a probabilistic learning framework capable of generalizing and learning a large number

of concepts from a single class example (i.e., one-shot learning). Rezende, Mohamed, Danihelka, Gregor, and Wierstra (2016) were among the first to provide a solution to the one-shot problem in the context of a neural network (NN) by implementing Bayesian reasoning into a deep NN embedded within hierarchical latent variable models. The Reptile algorithm (Nichol et al., 2018) is a recent approach to this problem. It counteracts the shortcomings of gradient-optimization algorithms by learning the model’s initial parameters in order to maximize its performance on novel tasks. Furthermore, it allows for any type of network to be used and does not place any restrictions on the type of loss function that can be used (Nichol et al., 2018). Moreover, it avoids the computational slowdowns of its counterpart, MAML (Finn et al., 2017). The algorithm for Reptile is as follows:

Algorithm 1: Reptile (serial version)

Initialize ϕ , the vector of initial parameters

for $iteration = 1, 2, \dots$ **do**

 Sample task τ , corresponding to loss L_τ on weight vectors $\tilde{\phi}$

 Compute $\tilde{\phi} = U_\tau^k(\phi)$, denoting k steps of SGD or Adam

 Update $\phi \leftarrow \phi + \epsilon(\tilde{\phi} - \phi)$

end

Turning over to the generative side of things, a popular form of generating data is through the use of a Generative Adversarial Network (GAN; Goodfellow et al., 2014). Here a generative model G captures the data distribution and creates samples from it, whereas a discriminative model D estimates the probability that the input is real (comes from the training data) rather than fake (comes from G). Formally, given x sampled from real data, and z being input noise, D tries to keep $D(x)$ near 1, while making $D(G(z))$ near 0. Conversely, G tries to make $D(G(z))$ near 1 (Goodfellow, 2016). A notable implementation is C-RNN-GAN (Mogren, 2016), a continuous recurrent neural network with adversarial training. It consists of a network with two LSTM layers and 350 hidden units per layer, in both G and D . However, D has a bidirectional layout which allows it to take context from both past and future for its decisions, while G is unidirectional (Mogren, 2016).

Notwithstanding GANs, other generative successes have made use of adaptations on the Long Short-Term Memory (LSTM) network (Hochreiter & Schmidhuber, 1997). Here, the cells of a network are designed to allow for information to persist by implementing a recursive state unit, an input gate, a forget gate, and an output gate. The input of the network is allowed into the state unit if it is allowed to pass through by the input gate. Next, the forget gate controls the self-loop of the state cell. Finally, the

output gate has the power to shut off the output cell (Goodfellow, Bengio, Courville, & Bach, 2016). Here, a notable implementation is Performance-RNN (Oore et al., 2018); a three layer unidirectional LSTM network with 512 units each.

GANs and LSTM networks have been tested and adapted to the few-shot learning problem (see Vinyals et al., 2016; Zhang et al., 2018) however, no such evaluation has been made in the musical domain. Therefore, this study will seek to answer the following research questions:

- To what extent is the music created by a few-shot generative model comparable to the music of a generative model that is trained on the entire dataset?
- To what extent is the music created by a few-shot generative model comparable to real music?

Methods

To provide an answer to the above research questions, this study will develop (in Keras) two generative models based on C-RNN-GAN and Performance-RNN, where the learning procedure will incorporate the Reptile algorithm. Other than that, the models will not differ from their original implementation. These models will firstly be compared to a baseline (a LSTM network of depth 1 and 200 units trained on the entire train set) and then with the music from the dataset. The evaluation between the models with the baseline and with the music in the dataset will be based on the number of statistically different bins (NDB; Richardson & Weiss, 2018) and domain specific measurements (polyphony, scale consistency, repetitions, tone span). Refer to the evaluation section for more details on these metrics. The data for the models will be one-hot encoded as per Oore et al. (2018); a vector for each time slice containing Note-On (128 values), Note-Off (128 values), Time-Shift (125 values), and Velocity (32 values) events.

Dataset

The dataset for this research will be similar to that of Oore et al. (2018) namely, the MAESTRO Dataset (Hawthorne et al., 2018). The dataset contains MIDI recordings from nine years of the International Piano-e-Competition. This amounts to 1,184 piano performances, approximately 430 compositions, 6.18 million notes played, and approximately 172 hours of playback. The dataset also features a recommended train/validation/test split. Moreover, each performance comes with additional metadata namely, the name of the composer, the title of the performance, the suggested

train/validation/test split, year of the performance, name of the file, and duration in seconds of the performance. The name of the performer is not included (Hawthorne et al., 2018).

Evaluation

Most DL models work by the principle of likelihood maximization, which simply says to choose the parameters that maximize the probability that the model assigns to the training data (Goodfellow, 2016). Formally, this means selecting the parameters that maximize $\sum_{i=1}^N \log p_{\text{model}}(\mathbf{x}^{(i)}; \boldsymbol{\theta})$:

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} \sum_{i=1}^N \log p_{\text{model}}(\mathbf{x}^{(i)}; \boldsymbol{\theta}). \quad (1)$$

That being said, calculating maximum likelihood can be thought of minimizing the Kullback-Leibler (KL) divergence. Thus, $\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\log p_{\text{data}}(\mathbf{x}) - \log p_{\text{model}}(\mathbf{x})]$. With this in mind, $\log p_{\text{data}}$ is a result of the data-generating process, and not the model. Therefore, a final simplification is applied where the maximum likelihood estimate would be calculated as:

$$\boldsymbol{\theta}^* = -\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\log p_{\text{model}}(\mathbf{x})]. \quad (2)$$

Note that Eq. 1 is the same as Eq. 2. The reason why this is brought up, is because negative log-likelihood (NLL; Eq. 2) is used both as a loss function and as an evaluation metric (Borji, 2018; Yu et al., 2016). However, as Borji (2018) notes, NLL is uninformative about the quality of the samples generated and it does not allow to answer whether the generative network is simply memorizing training examples. As such, this study will also include other evaluation metrics when assessing generative models: the number of statistically different bins (NDB; Richardson & Weiss, 2018), polyphony, scale consistency, repetitions, and tone span (Mogren, 2016).

NDB is an evaluation metric specifically designed for generative models. It follows the intuition that given two samples from the same distribution, the number of samples that fall into a bin (read as: cluster) should be the same (Richardson & Weiss, 2018). Let $I_B(\mathbf{x})$ be the indicator function for bin B . Then $I_B(\mathbf{x}) = 1$ if the sample falls into B , and zero otherwise. Let also $\{\mathbf{x}_i^p\}$ define N_p samples from a p distribution (e.g., test set samples) and $\{\mathbf{x}_i^q\}$ be the N_q samples from a q distribution (e.g., generated samples). If $p = q$ then the followings is also true:

$$\frac{1}{N_p} \sum_i I_B(\mathbf{x}_i^p) \approx \frac{1}{N_q} \sum_i I_B(\mathbf{x}_i^q) \quad (3)$$

The pooled sample proportion P_B is the proportion of samples (from the joined sets) that fall into B . The standard error for bin B is given by:

$$SE_B = \sqrt{P_B(1 - P_B)[1/N_p + 1/N_q]} \quad (4)$$

The test statistic is a z -score $z = \frac{P_B^p - P_B^q}{SE_B}$ where P_B^p and P_B^q are the proportions of each sample that fall into B . If z is smaller than a significance level, then the samples within that bin are statistically different. This process is repeated for each bin, and the number of statistically different bins is reported. The selection of bins is done through K -means clustering ($K \ll N_p, N_q$) on the \mathbf{x}^p samples. Each of the generated samples \mathbf{x}^q is by assigned to the nearest (L_2) K centroid (Richardson & Weiss, 2018).

The more domain-specific evaluation measures are adapted from Mogren (2016). Polyphony measures how often a minimum of two tones are played simultaneously (when the start time is the same). Scale consistency is the fraction of notes that are part of a standard scale (e.g., major, minor, lydian, etc.). Repetitions counts consecutive subsequences of notes, and tone span is the difference in semitones between the lowest note and highest note (Mogren, 2016).

The choice of NDB comes from the fact that it is good at detecting overfitting (Borji, 2018), whereas the domain-specific metrics would favor models that generate high fidelity samples. These evaluation metrics also have the benefit of being model-agnostic; they do not require a specific generative model.

Milestones

Task	Date
Develop one-hot encoder for MIDI data	March 15th
Develop domain-specific evaluation tools	March 22nd
Create a Keras implementation of Performance-RNN	March 24th
Create baseline	March 24th
Create a Keras implementation of C-RNN-GAN	March 29th
Augment models with Reptile	April 5th
Train models	April 5th - April 19th
Evaluate models	April 26th
Write thesis	April 26th - May 13th

References

- Borji, A. (2018). Pros and Cons of GAN Evaluation Measures. *arXiv:1802.03446 [cs]*. arXiv: [1802.03446 \[cs\]](#)
- Chen, L., Dai, S., Tao, C., Shen, D., Gan, Z., Zhang, H., ... Carin, L. (2018). Adversarial Text Generation via Feature-Mover's Distance. *arXiv:1809.06297 [cs]*. arXiv: [1809.06297 \[cs\]](#)
- Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., & Huang, J.-B. (2018). A Closer Look at Few-shot Classification.
- Clouâtre, L., & Demers, M. (2019). FIGR: Few-shot Image Generation with Reptile. *arXiv:1901.02199 [cs, stat]*. arXiv: [1901.02199 \[cs, stat\]](#)
- Dong, H.-W., Hsiao, W.-Y., Yang, L.-C., & Yang, Y.-H. (2017). MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. *arXiv:1709.06298 [cs, eess]*. arXiv: [1709.06298 \[cs, eess\]](#)
- Finn, C., Abbeel, P., & Levine, S. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:1703.03400 [cs]*. arXiv: [1703.03400 \[cs\]](#)
- Goodfellow, I. (2016). NIPS 2016 Tutorial: Generative Adversarial Networks. *arXiv:1701.00160 [cs]*. arXiv: [1701.00160 \[cs\]](#)
- Goodfellow, I., Bengio, Y., Courville, A., & Bach, F. (2016). *Deep Learning*. Cambridge, Massachusetts: The MIT Press.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative Adversarial Networks. *arXiv:1406.2661 [cs, stat]*. arXiv: [1406.2661 \[cs, stat\]](#)
- Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., ... Eck, D. (2018). Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset. *arXiv:1810.12247 [cs, eess, stat]*. arXiv: [1810.12247 \[cs, eess, stat\]](#)
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735. doi:[10.1162/neco.1997.9.8.1735](#)
- Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2016). Image-to-Image Translation with Conditional Adversarial Networks. *arXiv:1611.07004 [cs]*. arXiv: [1611.07004 \[cs\]](#)
- Karras, T., Laine, S., & Aila, T. (2018). A Style-Based Generator Architecture for Generative Adversarial Networks. *arXiv:1812.04948 [cs, stat]*. arXiv: [1812.04948 \[cs, stat\]](#)

- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 1097–1105). Curran Associates, Inc.
- Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266), 1332–1338. doi:[10.1126/science.aab3050](https://doi.org/10.1126/science.aab3050)
- Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2019). The Omniglot Challenge: A 3-Year Progress Report. *arXiv:1902.03477 [cs]*. arXiv: [1902.03477 \[cs\]](https://arxiv.org/abs/1902.03477)
- Larochelle, H., Finn, C., & Ravi, S. (2017). *Few-Shot Distribution Learning for Music Generation*. AI-ON.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. doi:[10.1038/nature14539](https://doi.org/10.1038/nature14539)
- Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., . . . Shi, W. (2016). Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *arXiv:1609.04802 [cs, stat]*. arXiv: [1609.04802 \[cs, stat\]](https://arxiv.org/abs/1609.04802)
- Mogren, O. (2016). C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv:1611.09904 [cs]*. arXiv: [1611.09904 \[cs\]](https://arxiv.org/abs/1611.09904)
- Nichol, A., Achiam, J., & Schulman, J. (2018). On First-Order Meta-Learning Algorithms. *arXiv:1803.02999 [cs]*. arXiv: [1803.02999 \[cs\]](https://arxiv.org/abs/1803.02999)
- Oore, S., Simon, I., Dieleman, S., Eck, D., & Simonyan, K. (2018). This Time with Feeling: Learning Expressive Musical Performance. *arXiv:1808.03715 [cs, eess]*. arXiv: [1808.03715 \[cs, eess\]](https://arxiv.org/abs/1808.03715)
- Ravi, S., & Larochelle, H. (2016). Optimization as a Model for Few-Shot Learning.
- Rezende, D. J., Mohamed, S., Danihelka, I., Gregor, K., & Wierstra, D. (2016). One-Shot Generalization in Deep Generative Models. *arXiv:1603.05106 [cs, stat]*. arXiv: [1603.05106 \[cs, stat\]](https://arxiv.org/abs/1603.05106)
- Richardson, E., & Weiss, Y. (2018). On GANs and GMMs. *arXiv:1805.12462 [cs]*. arXiv: [1805.12462 \[cs\]](https://arxiv.org/abs/1805.12462)
- Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching Networks for One Shot Learning. *arXiv:1606.04080 [cs, stat]*. arXiv: [1606.04080 \[cs, stat\]](https://arxiv.org/abs/1606.04080)

- Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., . . . Tsing, R. (2017). StarCraft II: A New Challenge for Reinforcement Learning. *arXiv:1708.04782 [cs]*. arXiv: [1708.04782 \[cs\]](#)
- Yu, L., Zhang, W., Wang, J., & Yu, Y. (2016). SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. *arXiv:1609.05473 [cs]*. arXiv: [1609.05473 \[cs\]](#)
- Zhang, R., Che, T., Ghahramani, Z., Bengio, Y., & Song, Y. (2018). MetaGAN: An Adversarial Approach to Few-Shot Learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 31* (pp. 2371–2380). Curran Associates, Inc.