Deep Reinforcement Learning for Music Generation

Fall 2018 Update

Benjamin Genchel
GTCMT
benjiegenchel@gmail.com

December 5, 2018

ABSTRACT

Symbolic music generation using deep learning based models has garnered significant attention in recent years. A diverse set of deep models have been applied to the task, from simple feed forward networks to GANs. One might assume, despite the lack of an agreed upon metric for evaluating quality, that the music produced by these models is improving over time. However, these methods all have known failure modes: most prominently, the inability to learn coherent long term structure. The output of such systems is generally agreed upon to have yet to learn perceptually relevant higher order features, such as phrasing, metric grouping, tension, and resolution. Deep reinforcement learning systems have shown great promise in learning complex high dimensional structure and achieving far higher levels of performance in domains where previous artificial systems had been impotent. Unfortunately, there have been few attempts to apply such a system to music, due to the abstract and subjective nature of the material; it is not clear what defines quality of output for music, or art in general. In this work, we seek to expand the body of work applying deep reinforcement learning to music generation, via the development and application of potential reward functions, both musical and otherwise, to a modern sequence based reinforcement learning system.

1. INTRODUCTION

The majority of modern deep learning based music generation models are variants of Recurrent Neural Networks (RNNs), and are trained using Maximum Likelihood Estimation (MLE) to predict the next event in a sequence, whether that event be a pitch, note-length, or embedded representation [6]. These models produce sequences by repeatedly sampling from their output distribution and feeding that output back into the input after initializing them with a seed sequence. This approach learns a conditional distribution of the occurrence of events given a preceding sequence which, especially when using modern gated architectures such as Long Short-Term Memory (LSTM) [10] or Gated Recurrent Unit (GRU) [3], can produce sensible output at short time scales. Other deep learning approaches include Convolutional Neural Networks (CNN) [23], which learn sets of spatial filters over data, Variational Autoencoders (VAE) [17], which learn parameters of a multivariate distribution describing the data, and Generative Adversarial Networks (GAN) [5], which allow the learning of a potentially complex and abstract loss function in the form of a separate network trained in parallel to tell the difference between generated samples and real samples. These model types have been arranged and combined in numerous ways in application to music generation, with each study reporting degrees of improved performance on some set of metrics. However, there has yet to be a model which overcomes the common, known weaknesses in deep learning based approaches to sequence generation, namely, the lack of ability to learn coherent long-term structure and perceptually relevant higher level features. Some examples of perceptually relevant higher level features with respect to music are phrasing, metrical grouping, tension and relaxation, and call and response.

Because music is abstract and subjective, these difficulties in the context of applications to music generation have often been ignored, or interpreted in a positive light (e.g. raising the question of whether the model is being creative). However, in text generation applications, these difficulties manifest as output that contains words and has correct local grammatical structure but no clear narrative, structure or meaning.

Deep Reinforcement Learning (DRL) presents an attractive alternative, allowing experience driven autonomous learning of unfamiliar environments [2]. DRL presents a general-purpose method for learning through trial and error, and allows for models to correlate immediate actions with delayed returns, an important aspect of many human-like tasks. Models based on DRL have become increasingly popular, and have been successfully applied to a number of complex tasks which appear to necessitate learning long term structure and higher level features (features which are abstract and not immediately available in data) within a given domain. Notable examples include driving [21] and Go [19].

DRL models problems that can be formulated as a Markov Decision Process (MPC), a learning framework which describes a process by which an agent in a particular state performs an action with respect to an environment,

resulting in a reward signal which then brings it to a new state [1]. It is readily intuitive to a composer, producer or musician that the creation of music could be formulated as an MPC in the context of DRL; The act of music creation is fraught with decisions spanning multiple dimensions from individual pitch and duration choices up to song structure, arrangement and engineering, each of which can lead to more positive or more negative "outcomes" with respect to the preferences of the creator and the creator's audience. Thus, one possible way to map the musical creation process to an MPC would be as follows:

- **Agent**: The composer or music creator.
- **State**: The current state of the composition, the music generated prior to reaching the current state.
- **Environment**: The musical expectations and preferences of the creator or the creator's target audience.
- **Rewards**: Judgments of quality by the creator or creator's target audience in accordance with their musical expectations and preferences.

Though, as above, we can conceptually define at a high level what the environment and rewards are, the abstract and subjective nature of musical preference, interest, and enjoyment make it difficult to formalize and quantify them into systems which can be implemented in a model.

One might naturally look to research in music perception and cognition for more concrete descriptions of these guiding factors. Musical preference, interest and enjoyment has been shown to be guided and sustained via a system of rewards in the brain [18], motivating the application of DRL to this domain further than the ready simplicity of conceptual mappings like the one presented above. However, the rewarding aspects of music that have been uncovered in prior research are still too abstract to be readily translated into implementable rules and valuations.

Music Theory, which encompasses a set of formal musical rules and frameworks, may be the next best thing. Music theory on the whole represents the endeavor to systematize and formalize the underlying rules that govern how music is arranged and composed; a set of concepts which can be used to create music. Though general western theory principles as well as those formulated for specific broad genre classes such as classical theory and jazz theory do not typically address cognitive features directly, they can be interpreted as representing cognitive rewards which govern musical preference. Lerdahl and Jackendoff's Generative Theory of Tonal Music (GTTM) [14]is a work of music theory which appears particularly apt for this type of interpretation. GTTM attempts to describe a formal musical grammar based on Gestalt principles which provides theory and methodology around a listener's understanding of music through structure [10]. There have been several attempts to develop computational systems that implement the ideas in GTTM, mostly for use in music segmentation [8, 9, 15]. There have been no attempts thus far to operationalize GTTM as an objective or reward function for music generation, though it has clear potential due to its specification of preference rules which are meant to reflect an experienced listener's understanding of a piece.

An alternate route to explicitly defining reward functions is to learn a reward system in a way that allows the encoding of these abstract informal preferences. Yu et al.'s SeqGAN model [25] does just that, converting the standard GAN method [7] into a reinforcement learning system in which judgment by the discriminator of a generation as realistic or unrealistic determines the reward given out. Thus, the discriminator is free to learn any set of abstract features or processes as long as they prove optimal for determining if a sequence is real or generated.

The long term goal of this work is to develop a model which integrates multiple reward functions both learned and specified with the goal of moving closer to learning long term structure and higher level perceptual features via the application of reinforcement learning. We will train the model using a dataset of over 200 bebop Jazz lead sheets, using harmony as a conditioner. We begin by building off of SeqGAN, which as mentioned allows the learning of a reward function from data, and plan to modify it with rule based reward functions developed from Jazz Theory and GTTM.

2. NOVELTY OF PROPOSED WORK

As mentioned previously, there have been few attempts to apply DRL to the task of music generation in general. This project will hopefully contribute significant work towards exploring that application and help pave the way for future projects in music generation using this approach. Though the original SeqGAN paper contained a small section on music generation, there was little information provided on the implementation and evaluation of that task. As music generation is the main focus of this work, we hope to produce a more complete report evaluating and implementing SeqGAN for music generation. Finally, to the best of our knowledge, there are no systems using GTTM as an evaluation metric for music, and none applying it as an evaluative metric for reinforcement learning.

3. RELATED WORK

There have been two recent attempts of note at utilizing or integrating DRL with deep learning for the task of sequence generation. The first, by Google's Magenta, attempts to combine RL with MLE by pre-training an LSTM-RNN on a dataset, then guiding its generation using an RL reward that combines the MLE with a custom reward system based on music theory regarding 18th century classical counterpoint [11]. The study compares the use of on-policy learning (Deep-Q Networks) with off-policy methods based on Kullback-Leibler Control (KL-Control), which attempt to maximize reward on a current task while minimizing deviation from a prior policy. The second, SeqGAN, presents a Generative Adversarial Network (GAN) based model which has been modified to generate sequences of discrete tokens by replacing the generator network with a policy- gradient RL system whose

rewards are given based on the discriminator network's judgment of the current state as real or fake [25]. This relieves the architecture of the need for continuous output, which allows the discriminator weight update to propagate through the generator. SeqGAN is a general sequence generator approach, and music generation was only one small test the authors performed with it. Both these systems have desirable properties which we plan to integrate into our own final model. SeqGAN, as mentioned, allows the learning of a loss function from data, which is useful given the previously mentioned difficulty in defining formal systems for judging the quality of music. In the other direction, TunerRL claims to have successfully defined a reward function from Music Theory and integrated it with the training of a generative model. At present, we have only worked with SeqGAN, but plan to use the design of TunerRL to integrate our own rule based systems later on in the process.

However, the concept of using RL for music generation, or framing music generation as a MDP is not entirely new. Le Groux et al. attempted to integrate reinforcement learning into a non-deep learning based system, with the ultimate goal of leading the generation towards increased musical tension [12]. Rewards for the system were calculated live via ratings given by listeners on how much tension they perceived in the music. Collins' Improvagent model [4] is an online model that plays along with live musicians. It uses the Sarsa algorithm [20] combined with a nearest neighbor search, and presents a reward function based on how much it changes over time and how much it influences the rest of musicians with its playing.

DRL has also been considered for other forms of creative generation, such as Sumi-e painting. Xie et al. developed a DRL model for this task in which the brush was considered as the agent, actions were rotations, movements and pressure on the canvas of the brush, and rewards were based on canvas boundaries as well as stroke overlap [22].

As previously mentioned, there is a substantial body of research into how rewards govern musical interest and preference in the brain. Though we have decided to focus our study on explicit reward systems derived from music theory, we feel the following is still worth mention in its relevance to our work. Salimpoor et al. describe how the brain derives reward in music listening via a process of tension and release, expectation and violation. The reward schema discussed is fairly abstract and difficult to implement as a set of rules, though it provides some higher-level insight that may provide some guidance in the development or adaptation of other reward functions [18].

One of our aims is to develop a reward function based on GTTM. There is a very large body of work done by Hamanaka et al. around the formalization and implementation of GTTM for computational purposes [8,9,15]. These works primarily focus on imposing structure over musical data in a supervised fashion, using human annotations. At the outset, it is difficult to tell how this work can be used in composition rather than strict analysis (cite all the papers here), though the sheer breadth of the work implies probable usefulness.

4. METHOD

The remainder of the paper will focus on current progress, specifically the implementation and initial training results of the vanilla SeqGAN model.

4.1 SeqGAN

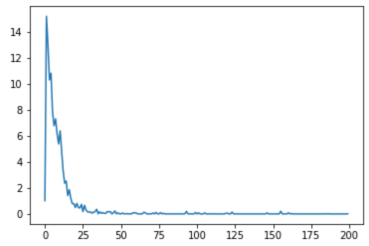
SeqGAN was developed to solve the problem of using GAN based models for discrete sequence generation. In a typical GAN setup, the output of the generator needs to be continuous (and thus differentiable) in order for the loss determined by the discriminator to be able to backpropagate through the generator, allowing the system to train end to end. Sampling from a distribution to get a discrete token is a non differentiable operation, making it difficult to use discrete sequence models as generators in a GAN framework. SeqGAN gets around this issue by reframing the system as a reinforcement learning problem, in which the current state is given by how much of a sequence has already been generated, an action is defined as the generation of a token, and the rewards are determined by the probability given by the discriminator that a sequence generated by the generator is real. SeqGAN uses an LSTM-RNN as its generator, and a CNN as its discriminator.

In a reinforcement learning problem, a reward is needed for every action taken, which in SeqGAN corresponds to a step of generation. However, the discriminator is only able to judge a full length generation; even with a discriminator that could handle variable length inputs, it would not make sense to judge the realism of an incomplete sample. SeqGAN solves this using Monte Carlo Rollouts (MCR). During adversarial training, the generator first generates a single full length sample starting from an input of one step of zeros. This sample is copied and converted into a preceding input by dropping its final step and appending a step of zeros to its beginning. Passing this copy into the network then gives a sequence of probability distributions corresponding to the original generated sample. MCR is then applied to this sequence of distributions - for each subsequence starting from step zero, "rollout" the remainder of the sequence multiple times using the subsequence as a seed and average the probability of realism given by the discriminator for each rollout. This gives an average reward for the current state of the generator for each time step. These per step rewards are then element wise multiplied with the probabilities of the tokens sampled in the original full sample generation to determine the final rewards.

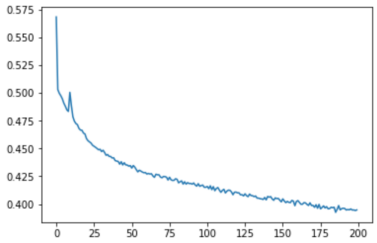
4.2 Training

We implemented SeqGAN in PyTorch[1], building off of an adaptation of the author's original TensorFlow[2] implemen-

[1] https://pytorch.org/
[2] https://www.tensorflow.org/



(a) Adversarial Generator Loss



(b) Adversarial Discriminator Loss

Figure 1: The loss over 200 epochs for the generator and discriminator of SeqGAN during adversarial training on the Nottingham dataset.

tation[3] by ZiJianZhao[4]. We applied the model to the Nottingham Folk Dataset [], which was the datset used by Yu et al. in the original paper in giving an example of applying the architecture to music generation, and 57 Bebop Jazz lead sheets from the Charlie Parker Omnibook []. For each dataset, musical notes were provided in the form of numbers ranging from note A0, or MIDI note number 21 and C8, or MIDI note number 88, with an extra token, 89, provided for rests.

4.2.1 Nottingham

When training on Nottingham, we aimed to keep to the author's original implementation as possible in order to try to recreate and validate their reported results. The dataset was parsed into overlapping 32 token subsequences, with each token representing a sample of 0.4 seconds. Only the monophonic melody was considered for each track. The generator was pre-trained by maximum likelihood estimation (MLE) using negative log-likelihood loss for 120 epochs with a learning rate of .001. Pre-training was done over the full sequence; teacher forcing [] was not used. The discriminator was then pre-trained on 10 generations of data by the generator for 6 epochs each, also using a learning rate of .001. Following pre-training, the model was trained adversarially for 200 epochs. During each adversarial epoch, the generator was trained by reward for 1 step and the discriminator trained for 2 epochs each on 4 data generations from the generator. Loss curves for each constituent model are presented in **Fig. 1**.

4.2.2 Charlie Parker Omnibook

The model was additionally trained on 57 Bebop Jazz Lead sheets from the Charlie Parker Omnibook [16]. Each lead sheet was converted from its original MusicXML[5] format into a discrete time format in which 4 ticks was equivalent to one beat. We parsed the dataset into overlapping 32 tick sequences (2 bars each) and only considered the monophonic melody. The generator was pre-trained for 8 epochs, achieving an extremely low loss which forced early stopping. As with the Nottingham Dataset, the discriminator was pre-trained for 6 epochs each on 10 data generations from the generator. The model was again trained adversarially for 200 epochs, with generator training for 1 step and the discriminator training for 2 epochs each on 4 data generations from the generator. Pre-training loss curves and adversarial loss curves are presented in **Fig. 2**.

[3] https://github.com/LantaoYu/SeqGAN

[4] https://github.com/ZiJianZhao/SeqGAN-PyTorch

5. DATA

5.1 Nottingham Dataset

The Nottingham Folk Music dataset is a database of 1037 british and american folk tunes. Yu et al. originally used Nottingham to provide a brief example and evaluation of SeqGAN's potential as a music generation model.

5.2 Charlie Parker Omnibook Dataset

The Charlie Parker Omnibook dataset [16] is comprised of 57 transcriptions of live recordings of Charlie Parker songs, including improvisations. Towards the pursuit of training on lead sheets specifically, each transcription in the set was edited down to only contain composed sections of the piece as monophonic melodies with chord labels. Improvisations were dropped. Each song was copied and transposed up a semi tone 6 times and down by a semitone 5 times, creating a version in every key.

5.3 Bebop Dataset

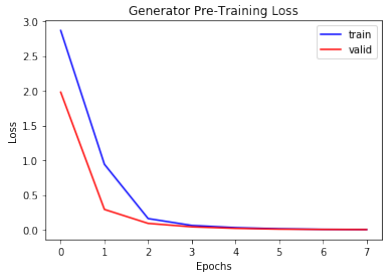
A new dataset was created for this task containing over 200 Jazz Bebop lead sheets combining the Charlie Parker Omnibook sheets described above with sheets transcribed from Hal Leanord's "The Real Bebop Book" [13]. Each lead sheet contains a monophonic melody accompanied by chord labels. We plan to train models going forward using this dataset, with a beat resolution of 24 ticks per beat, or 96 ticks per bar. We were unable to test out the dataset using SeqGAN due to an inability to load the entire set into memory with our desired resolution. Of course, loading an entire dataset into memory is a naive approach to data handling in general, and will be rectified in our code in the near future.

6. EVALUATION

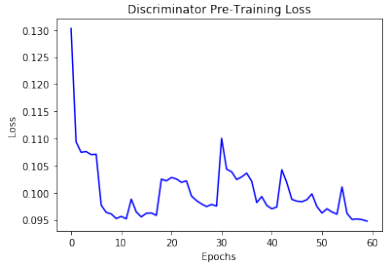
6.1 BLEU Score

In accordance with the work of Yu et al. in the original SeqGAN paper, we evaluate the performance of our implementation using BLEU score. BLEU score is a modified precision metric used in machine translation. BLEU

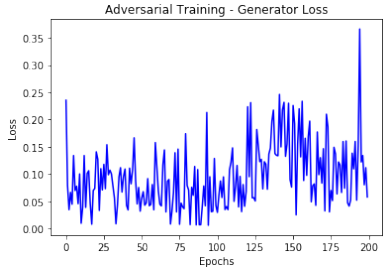
[5] https://www.musicxml.com/



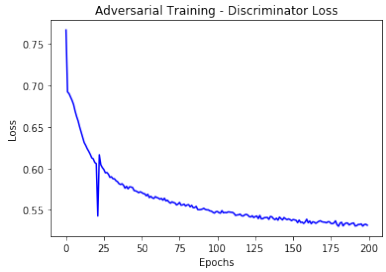
(a)



(b)



(c)



(d)

Figure 2: Loss curves for pre-training and adversarial training of the generator and discriminator networks of SeqGAN during training on the Charlie Parker Omnibook dataset. 2a presents the validation and training loss during the MLE pretraining. One can see how heavily the model overfits over its 7 epochs of training. 2b shows the pretraining loss for the discriminator over 60 combined epochs. As expected, there is a spike in loss each time a new set of fake samples is made by the generator. 2c shows the loss of the generator during adversarial training. The oscillations in the loss appear to reflect a well played game against the discriminator. 2d shows the loss of the discriminator during adversarial training. The nearly monotonic downward slope appears to indicate that the generator was unsuccessful at playing an even game with the discriminator. However, the range over which the generator loss decreased was not substantial.

compares constant length sub-sequences between a machine translated sentence and an accurate reference translation. Specifically, in the absence of a specification by the original authors, we used BLEU-4, which compares sub-sequences of length 4. We used sub-sequences from the original dataset as inputs, the next step in time following those sequences as the target, and the outputs of the model given the inputs as the translation. In general, a score of 1 indicates a perfect translation, while 0 indicates a completely incorrect translation.

6.2 MGEval

We additionally evaluated our models using distribution comparisons from the MGEval toolbox [24], which provides intraset and interset probability distributions for a number of different music specific metrics such as "Total Number of used Pitches" and "Note Length Transition Matrix". For intraset calculations, MGEval compares each sample in a dataset to the others, creating a distribution for the distance of one sample to the others on a given metric. For interset calculations, MGEval compares each sample in a dataset to the samples in another dataset, creating a similar distribution. For two datasets on a single metric, the distributions for each intraset distance and the distribution for interset distance are each plotted on the same graph. If the distributions are highly overlapped and similar in shape, this indicates either a complete overfit of the data or a good generalization. However, if there is not much overlap, or one set's distribution is narrow and only overlaps with a small region of the other, this could indicate a mode collapse, in which the model has overfit on a particular type of stimulus which is more populous in the data.

7. RESULTS

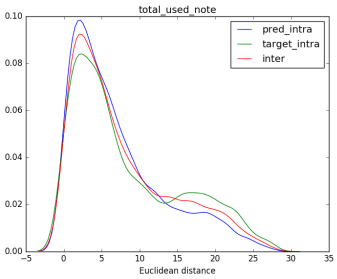
We evaluated both our pre-trained generator and fully trained SeqGAN generator using BLEU score in an attempt to replicate Yu et al.'s original reported results. They reported very high results both for their baseline, an MLE trained recurrent model, and their fully trained model. Their results and ours are reported in **Table 1**.

	BLEU-4 Score
Paper Reported	0.9406
Pretrained Generator	0.3419
Adversarially Trained Generator	0.3834

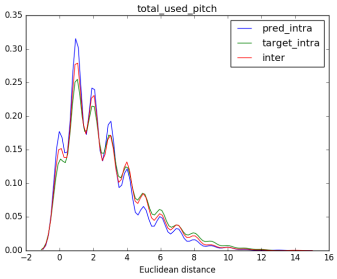
Table 1: BLEU scores for SeqGAN trained on the Nottingham Dataset.

BLEU score, if our interpretation is correct, seems an inappropriate metric for this task, as it rewards overfitting - the highest score is achieved by perfectly predicting the next step of a sequence every time. It makes even less sense as an application to a GAN based model, which is not trained towards a particular output but instead learns based on the abstract concept of realism, as defined by

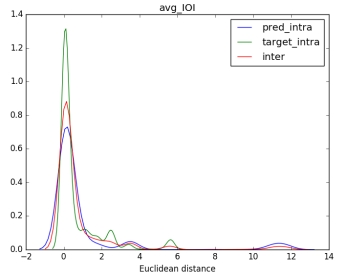
the task of fooling the discriminator. It appears that, because the MLE pre-trained generator is trained specifically to predict the next step of a sequence, it should achieve a higher BLEU score than the adversarially trained generator. Thus, we suspect that the original authors heavily overfit the training data intensely to achieve their reported scores. Using metrics in MGEval (**Fig. 3**), we found that our model's distributions closely matched those of the Nottingham dataset, which in combination with lower BLEU score seems to indicate a good generalization.



(a) Total Number of Used Notes



(b) Total Number of Used Pitches



(c) Average Inter-Onset Interval

Figure 3: Results of MGEval Cross-Validation on the Nottingham Dataset.

Most likely owing to the much smaller size of the Charlie Parker dataset, the model heavily overfit the training data during pre-training after only 8 epochs, and as a result the BLEU scores we achieved training with that data set were much closer to those reported by Yang et al. (**Table 2**).

8. DELIVERABLES

Current deliverables include a working baseline model in the form of SeqGAN, with preliminary objective results reported on two datasets.

	BLEU-4 Score
Our Pretrained Generator	0.9998
Our Adversarially Trained Generator	0.9999

Table 2: BLEU scores for SeqGAN trained on the Charlie Parker Dataset.

9. CONCLUSION AND FUTURE WORK

At present, we have managed to implement a working version of our baseline reinforcement learning system and provide some evaluation on its performance. We were unable to train the model on our full planned dataset at our desired resolution due to naive data processing methods, though once proper data processing is in place, training with that dataset should be straight forward.

Now that we have a working baseline system, our next steps are to begin development of rule based systems which implement ideas from GTTM and music theory. We plan to reach out to the authors of TunerRL in order to gain a better understanding of how they codified their own theory rules, as well as seek out existing implementations of the rules described in GTTM.

In addition, though the combination of MGEval, which can indicate a model's performance in learning the data distribution in musically significant ways but cannot tell complete overfitting from generalization, and BLEU score, which can indicate whether a model has completely overfit, seems like a useful objective metric, we still desire to conduct a subjective evaluation given the nature of the task. Towards that end, we aim to propose a listening test experiment design in the near future.

10. ACKNOWLEDGEMENTS

We would like to acknowledge Ryan Rose for his contribution to the work presented here thus far. In service of a separate but related project, Ryan contributed lead sheet transcriptions, evaluation implementations and general debugging help which helped this project along. We would also like to thank IRCAM for graciously sharing the Charlie Parker Omnibook dataset.

11. REFERENCES

[1]

[2] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[3] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Computation and Language*, 2014.

[4] Nick Collins. Reinforcement learning for live musical agents. In *ICMC*, 2008.

[5] 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. *arXiv:1709.06298 [cs, eess]*, September 2017. arXiv: 1709.06298.

[6] Douglas Eck and Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks. *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, 103, 2002.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Implementing "a generative theory of tonal music". *Journal of New Music Research*, 35(4):249–277, 2006.

[9] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Melody expectation method based on gttm and tps. In *ISMIR*, pages 107–112, 2008.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Natasha Jaques, Shixiang Gu, Richard E Turner, and Douglas Eck. Tuning recurrent neural networks with reinforcement learning. 2017.

[12] Sylvain Le Groux and PFMJ Verschure. Towards adaptive music generation by reinforcement learning of musical tension. In *Proceedings of the 6th Sound and Music Conference, Barcelona, Spain*, volume 134, 2010.

[13] Hal Leonard. *The Real Bebop Book*. Hal Leonard Corporation, 2017.

[14] Fred Lerdahl and Ray S Jackendoff. *A generative theory of tonal music*. MIT press, 1985.

[15] Eita Nakamura, Masatoshi Hamanaka, Keiji Hirata, and Kazuyoshi Yoshii. Tree-structured probabilistic model of monophonic written music based on the generative theory of tonal music. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 276–280. IEEE, 2016.

[16] Charlie Parker and Jamey Aebersold. *Charlie Parker Omnibook*. Alfred Music Publishing, 1978.

[17] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. 2018.

[18] Valorie N Salimpoor, David H Zald, Robert J Zatorre, Alain Dagher, and Anthony Randal McIntosh. Predictions and the brain: how musical sounds become rewarding. *Trends in cognitive sciences*, 19(2):86–91, 2015.

[19] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[20] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[21] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[22] Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. *IEICE TRANSACTIONS on Information and Systems*, 96(5):1134–1144, 2013.

[23] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, pages 324–331, 2017.

[24] Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music. *Neural Computing and Applications*, November 2018.

[25] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.