Coarse-To-Fine Framework For Music Generation via Generative Adversarial Networks

Dan Ma
School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu, China
182448161@qq.com

Bin Liu
Center of Statistics Research, School of Statistics
Southwestern University of Finance and Economics
Chengdu, China
liubin@swufe.edu.cn

Xiyu Qiao
College of Science and Engineering
The university of Edinburgh
Edinburgh, UK
Telephone number, incl. country code
tim.qiao@outlook.com

Danni Cao
Center of Statistics Research, School of Statistics
Southwestern University of Finance and Economics
Chengdu, China

Guosheng Yin
Department of Statistics and Actuarial Science
The University of Hong Kong

ABSTRACT

Automatic music generation is highly related to Natural Language Processing (NLP). A current note in melody always depends on its context, just like a word in NLP. Yet the difference is that music is built upon a set of special chords that formulates the skeleton of the melody. To enhance automatic music generation, we propose a two-step adversarial procedure: Step 1 learns to generate chords via a chord generative adversarial networks (GANs); and step 2 trains a melody GAN to generate music for which the input is conditioned on the chords produced through the first step. Under such a two-step procedure, the chords generated in the first step formulate a basic framework of the music, which can theoretically and practically improve the performance of melody generation in the second step. Experiments demonstrate that such a cascading process is able to generate high-quality music samples with both acoustical and music theoretical guarantees.

CCS Concepts

Applied computing→Sound and music computing;•Computing methodologies→Neural networks

Keywords

melody generation; adversarial learning; Bi-LSTM

1. INTRODUCTION

The goal of automatic music generation is to develop music samples in a fully automatic way without composers' interactions. Through extensive training, the resultant music should sound pleasant acoustically and realistically, similar to those produced by composers. Traditionally, the temporal model is the most popular approach to solve this problem [1, 16]. Recently, the generative adversarial networks (GANs) [6] has demonstrated remarkable power in image generation [5, 7, 12] and natural language processing [14, 19]. Inspired by rapid advances in GANs, the temporal model has been used in conjunction with GANs to synthesize higher-quality music [3, 9, 17, 18]. The C-RNN-GAN [9] makes an analogy of melody to simple sequential data and uses the adversarial training to model the joint probability of music sequences. However, music data are composed of both high-level phrases and low-level notes(bars), and thus it is not suitable to model music data with traditional sequential methods [3]. Also, some focuses are put on utilizing the low-level structure of music [3, 18]. The work by Yang et al. [18] trains GANs to generate melody by learning current music bars with the priors of previous bars. Dong et al. [3] develop a multitrack melody generating model by leveraging multiple levels of information in pieces of music. In most cases[7], the music is composed of multiple tracks, such as bass, guitar, piano, and chord etc., and a melody is hierarchically composed of pitches, beats, bars, and phrases. However, it is expensive and impossible to model all these characteristics listed above with an existing learning framework. Among all the relevant musical characteristics, chords typically set the profile of the music, which dominate the basic outline of the melody. Due to this fact, chords are not only an array of reasonable notes pleasant to listen to but it also forms an outline or base for a dulcet piece of melody. Under an appropriately constructed chord array, there exists an infinite number of possibilities of note permutations that are harmonious with each other. On the other hand, if the chords are not constructed properly, notes may be settled far from being pleasant,
sometimes they even become noise. Therefore, an interesting piece of melody is based on two key factors: a reasonable array of chords and a set of notes that obeys the rule of the chords. Following this theory and inspired by [20], we propose a two-step adversarial learning model to generate high-quality melodies from the coarse to fine stages. In the first step, our model focuses on

HPCCT & BDAI 2020, July 3–6, 2020, Qingdao, China
©2020 Association for Computing Machine
ACM ISBN 978-1-4503-7560-3/20/07. . . $15
http://dx.doi.org/10.1145/3409501. 3409534

capturing the outline of a melody with adversarial learning. More specifically, we train a standard GANs to learn the distribution of chords (named as the chord GANs). In the subsequent step, a conditional GANs [8] is trained to generate a series of notes to form the melody. The discriminators and generators in both steps are implemented by Bi-LSTMs to capture the temporal dependency of the music data. The proposed two-step adversarial training process divides the music generation into 2 phrases: the chord learning and the melody learning conditioned on the generated chords. Our approach is related to [18], however we utilize the hierarchical conditional relationship from the chords to melody (i.e., generating melody from the given chords). This has not been considered in earlier studies. The contributions of this paper falls into three aspects:

(1) We propose a novel coarse-to-fine music generation framework that makes the ‘coarse’ chords as the prior stage for the ‘fine’ melody generation;

(2) the proposed algorithm is easy to train owning to its lightweight architecture of stacking two GANs;

(3) our model is trained on the Nottingham Music Database and the generated melodies satisfy both acoustical and music theoretical guarantees.

We open all source code for those who are interested in this area, and all code of the proposed model can be found at https://github.com/punkcure/stack_chord_gan

2. RELATED WORK

2.1 Chord recognition

As the chord system plays a significant role in music generation, extensive research works have been conducted to process chords from the algorithmic perspective. [15] introduced a method called COCHONUT to recognize chords flow from the symbolic music data: MIDI (Musical Instrument Digital Interface). COCHONUT aims to estimate the most probable time point that would trigger a chord change within a song. Then the song is split into segments based on the arrays of triggering time points. For each segment, a score is calculated to predict which chord is the most suitable. The key basis in this prediction involves contextual harmonic information, decision theory, pattern matching and rule-based recognition. To extract the most-common-use patterns of chords, [15] expressed each chord with a root note and a type of chord patterns, and we introduce the same encoding method for chords to train.

2.2 Music generation

With the growing computational ability and large number of musical data, more and more researchers are involved to the work of comprehending and generating music. [4] introduced LSTM to replace RNN to write blues songs [1] combining LSTM (focuses time dependency) and RBM (Restricted Boltzmann Machine) together to output a group of chords. Song From Pi [2] is based on the Recurrent Neural Network and divides complete music into four layers (Key, Press, Chord and Drum). The model uses RNNs to generate the key and the duration of press. In terms of another study published in the same year, the C-RNN-GANs [9], which mix the core of Song From Pi, RNNs, and the idea of C-GANs, creates the possibility to make the model become self-improving by adding the discriminator to identify whether the sample is real or not. Based on the idea of GANs to leverage both discriminator and generator, there are two articles published in 2017 respectively. The [3] introduces 3 models to generate music. The other one is MidiNet[18], which also takes advantage of the idea of GANs to produce multi-channel MIDI based on CNNs. Given that CNNs are faster to run, Wavenet[13] published by Deepmind also models to generate music based on CNNs. While different from all studies shown before, Wavenet[13] generates music from the aspect of modeling raw audio instead of using other symbol to represent music like MidiNet or Song From Pi .

2.3 Generative Adversarial Networks

GANs(Generative Adversarial Networks)[6] introduces an adversarial learning concept to generate data that follows the prior distribution from noise samples. It contains two correlative nets: the generator net and discriminator net. The generator is trained to produce data that is hard to tell from ground-truth and the discriminator tries its best to not to be fooled and keep the ability to recognize the fake data. While this competing process of training between two nets is going on, a two-player minmax game has come into being.

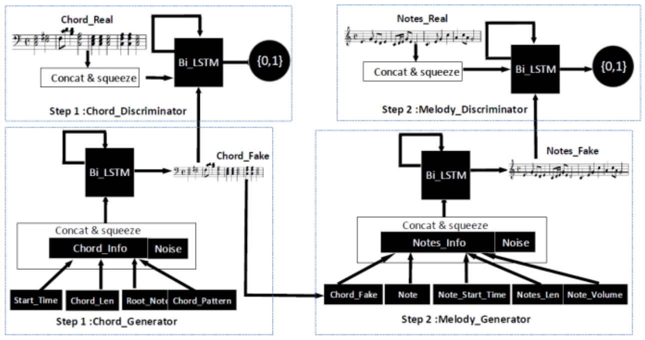

Figure 1. Demonstration of the pipeline of the proposed model.

3. DATA PREPROCESSING

We encode the data of musical concepts (i.e., notes and chords) that relevant to our model in some concise and efficient manner based on the MIDI standard.

3.1 MIDI

MIDI, which is short for ’Musical Instrument Digital Interface’ is a digital music standard that described the channels of electronic instruments. A MIDI file records no audio or wave data, instead, it contains a set of at least one MIDI channel, each of which carries the pitches, duration, velocity and some other attributes of the melody. All these attributes are encapsulated as particularly defined MIDI messages including system messages and channel messages.

3.2 Notes

In music, a note is the pitch and duration of a sound, which also highlights its representation in musical notation 1. For ease of exposition, we prefer to representing a musical note as a quaternion (NOTE_START_TIME, NOTE_START_LEN, NOTE_FREQ, NOTE_VELOCITY), which concerns about the following questions:(1). when a note begins to press in ticks; (2). how long a note lasts in ticks; (3). Which note is pressed; (4). the volume of a note. The values of the quaternion terms can be extracted from the MIDI pattern. Following the MIDI Tuning Standard (MTS), note keys are expressed by 128 discrete turnings. To simplify calculation, the MIDI tuning value can be transformed into frequency as follows,

$$f = 2^{(d-69)/12} * 440Hz$$

The inverse operation of equation 1 is:

$$d = 69 + 12 * \log_2(f / 440Hz)$$

3.3 Chords

Typically, a single chord is composed of 3 to 6 notes, and those notes together express some specific patterns conforming to the musical theory. Scholz and Ramalho [15] proposed to build a chord pattern to score the probability of the chord's appearance in a melody. Motivated by this work, we split a chord into two basic units: a root note and a chord pattern. The root note is typically a bass that dominates the hearing sense. On the other hand, the chord pattern contains a series of notes, and there are about 30 chord patterns, such as maj, min, dim, aug etc. Taking the min as an example, it is a common pattern that is composed of a root note, a minor third note, and a perfect fifth note. If we make the root note aligned with the origin 0, to keep the relative interval of the other notes within the twelve-tone scale, the minor third note must be coded as 3. Analogously, the perfect fifth note is 7. Following this route, we obtain the min chord pattern (0, 3, 7). Similarly, the maj chord is (0, 4, 7).

By categorizing the most commonly used patterns of chords into about 30 types (such as maj, min, dim, aug, . . .), we obtain a list of basic patterns. These patterns emphasize on describing the relationship of the musical interval among the note members in a chord.

By appending all pattern vectors in a uniform list, every single pattern can be indicated by its index of the list. We make one step further to encode a single chord pattern into a group of bits, which is analogous to one-hot encoding. All values of this combined group are 0, except that the n-th bit of the pattern list is 1 (the integer n denotes the pattern's index in the pattern list), and the length of this group of bits is relative to the length of the pattern list.

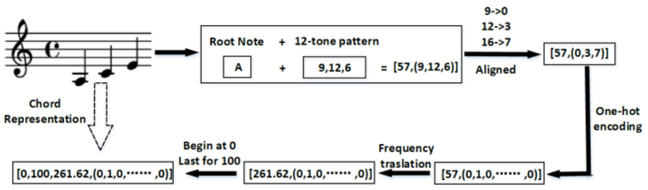

Figure 2: Illustration of how we encode a single chord from stave to a training vector

By linking a root note (expressed in frequency) and a given chord pattern (expressed in one-hot encoding), we can uniquely define a chord. For example, chord Amin can be split as root A and chord pattern min. The root note A is 57 based on the MIDI standard (assuming in a one-lined octave scale Nakamura et al.[10, 11]). According to (1), the frequency of 57 is 261.62Hz. As discussed above, pattern min, which contains the group of values (0, 3, 7), is the second item in the pattern list, and we encode it to a 32-dimensional vector [0, 1, 0, . . .]. Combined with the root key, we obtain (261.62, [0, 1, 0, . . .]), which is the representation of a single chord Amin in our model. In addition, we append the static chord data with 'begin ticks' and 'length in ticks' in the same way as that for the note data. Finally, chord Amin, which starts at 0 ticks and continues in 100 ticks, is represented as (0, 100, 261.62, [0, 1, 0, . . .]). We illustrate the whole translation process in Fig. 2.

4. MODEL OVERVIEW

As shown in Fig. 1, we develop a novel coarse-to-fine melody generation system. The proposed framework comprises of a chord learning step and a melody learning step, with each trained by a GANs process. In Step 1, we learn to output the chord 'skeleton' of a melody, and in step 2, the generated notes act as the 'blood and flesh' to fill up the 'skeleton'.

4.1 Step 1: Chord GANs

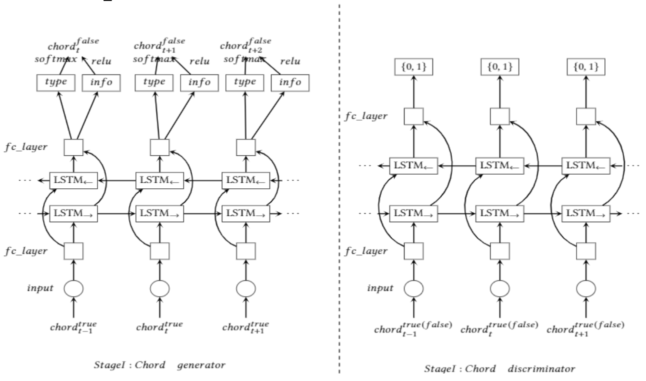

Figure 3: The architecture of our models to generate chords progression in stage 1

In the first step, we produce a series of dulcet chords progression by training a chord GANs. Both the generator and discriminator are implemented by Bi-LSTMs. The training data flow pass the model (both the generator and discriminator) in the bidirectional channels, and thus the contextual information about the current chord is connected not only to the prior chords but also to the subsequent chords.

Although the number of notes in every chord is not identical (3 to 6 as we mentioned before), we take these multi-elements chords as a whole block, with each block being related to its context in temporality.

We unfold the architecture of generator and discriminator through a timeline and demonstrated them in Fig. 3. The training data flow pass the model (both generator and discriminator) in the bidirectional channel, thus, the contextual information about current chord is connected not only to prior chords but also the subsequent chords.

To construct the adversarial system, we follow the original GANs loss function in [6].

Thus, the loss functions for the discriminator and generator of step 1 can be written as:

$$L_{D_1} = E[D_1(chord_{true}) - D_1(G_1)]$$

$$L_{G_1} = E[-D_1(G_1)]$$

where G1 denotes the chord generator, D1 denotes the chord discriminator in step 1, and $chord$true represents the training samples.

4.2 Step 2:Melody GANs

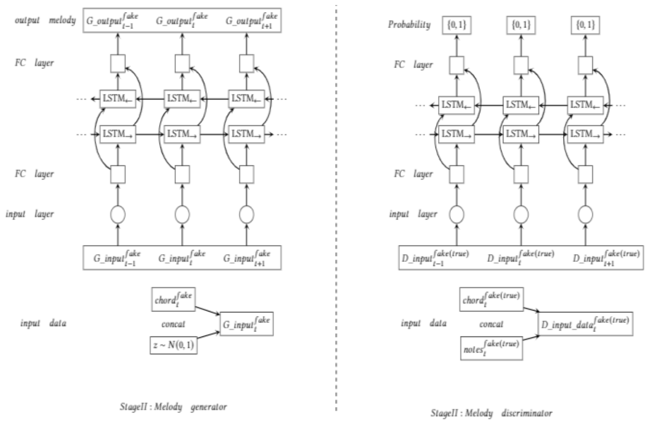

Figure 4: The architecture of our models to generate melody in stage 2.

In step 2, we train another GANs(named the melody GANs) for melody generation, as shown in Fig. 4. The generator of the melody GAN accepts the fake chord (learned by the chord GANs) as the condition part of its input. In fact, the melody GANs has a conditional structure that these fake chords learned from step 1 are treated as conditions to participate the melody generation in step 2. The notes are strictly related to the temporality and context dependence, so we build a similar adversarial system based on Bi-LSTM as the first stage does. We construct the conditional architecture inspired by the method of controlling GAN's result with conditional input vector such as CGANs [8]. As for melody generator, we make the chord data together with the latent code Z sampled from normal distribution($Z \sim N$ (0, 1)) as the input data. Thus we can construct series of input data for generator along the time-line, one block of mixed-code(chord data and Z data) for one time step. We still use a bi-directional LSTM to work on these input data, after forward layer and backward layer's recurrent operation, a full connect layer is provided to compose the output note data, a four dimensional vector represents the start time of note (NOTE_START_TIME), the time the note lasts (NOTE_START_LEN),the frequency that indicates which note is playing(NOTE_FREQ) and the velocity of the note(NOTE_VELOCITY). The predicted fake notes are then join together with the conditional chords as the negative samples for the notes discriminator. The positive samples are extracted from the dataset. The discriminator tries to seek the connection between the chords and notes through the time line, and make the judgement if the input data is real or fake.

The loss functions in step 2 are the conditional settings [8] that extend the inputs of both the generator and discriminator with conditional constraints. In our model of step 2, the conditional constraints are chords data corresponding to the notes. The loss functions can be constructed as follows,

$$L_{D_2} = E[D_2(note_{true} \mid chord_{true}) - D_2(G_2 \mid chord_{fake})]$$

$$L_{G_2} = E[-D_2(G_2 \mid chord_{fake})]$$

where $note_{true}$ and $chord_{true}$ are real samples from the training set, $chord_{fake}$ is the output of the chord GANs from step 1, and G2 and D2 correspond to the generator and discriminator of the melody GANs. We minimize both loss functions during the training process.

5. EXPERIMENT

We perform our experiment to check the performance of our model by training it with Nottingham Music Database(NMD for short). To avoid being subjective and one-sided, we introduce several quantitative metrics along with statistics of acoustical perception from volunteers to evaluate the model thoroughly. The results show that our model is competent in generating melody that follow the rule of chord theory reasonably. Besides, a small-scale study with 14 human listeners is conducted to check our production sensuously.

Fig. 5 display the generated chord and melody segments respectively. As shown in Fig. 5, the samples output by step 1 are a series of chords (such as the root note, the third note, and the fifth note). The trend of these chords is in accordance with some regular patterns. With the restriction of the given chords, the generated notes are produced in a harmonious style.

Table 1. Comparison of the models to be checked

	C-RNN-GAN	**MidiNet**	**Song from PI**	**Ours**
Backbone	RNN	CNN	RNN	Bi-LSTM
Previous notes	Yes	Yes	No	Yes
Adversarial training	Yes	Yes	No	Yes
Chords as condition	No	Yes	Yes	Yes

5.1 Dataset

We use NMD which contains more than 1k traditional British related folk dance tunes as the training resource of our model. The NMD uses ASCII notation 3 to store music data, which is unfriendly to perform but makes the analysis work of music more easier. It is convenient to be converted to printed stave format, yet owing to the need of data format for our training model, we make a conversion to alter the origin ASCII format files to MIDI files.

About 1400 ASCII notation format files are converted to MIDI files among the total dataset, each of which has both chord track and melody track, thus we get more than 5000 minutes' length of MIDI training data.

5.2 Metrics

For the purpose of intuitively checking the quality of melody output from our model, we split the melody into small segments, each has 50 notes. These output segments are evaluated with music theoretical metrics discussed later. Since the proposed model is symbolic based, We choose 3 famous symbolic-domain music generator model C-RNN-GAN [9], MidiNet[18], Song From Pi [2] as the baseline to measure the performance of our model. By examining characteristic of models to be checked in some relative aspects, we list them in Table 1. In addition, we take

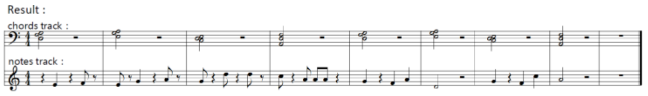

Figure 5: Results generated by the proposed model.

training dataset Nottingham into account to check if our model is competent in grasping the features of the ground truth data. We randomly sample 100 melody segments (each has 50 notes) from the five sources including NMD(ground truth) to evaluate the results with the following quantitative metrics.

Tone using distribution counts the types of tone appeared in the generated melody segments. For a single segment, using too many (greater than 40 may lead to messy) or too few (less than 20 may sound monotonous) types of tones are usually not acceptable. We visualize the histograms of the types of tones in the five sources as shown in Fig. 6. The closer to Nottingham, the better the performance. As the Fig. 6 shows, the number of kind of tones generated from stack_chord_gan are mostly drop in the range[30,40], which is more consistent with Nottingham. On the other hand, C-RNN-GAN generates melodies consist of less than 20 notes in most cases. Notes produced by MidiNet seems distributed more proportional in all types than others. As for Song from PI models, it generates notes that mostly drop in range[20,30], a little fewer than the ground truth dataset. Obviously, our proposed model shows more likeness to ground truth music in notes type distribution than others. We argues that the chord-based mechanism together with the adversarial training process helps proposed model to constrain the produced notes types in a more reasonable manner. Since a chord is given as the base for the process, it acts as a guide for the notes production that the notes must obey the rules to make itself compatible with the given chord.

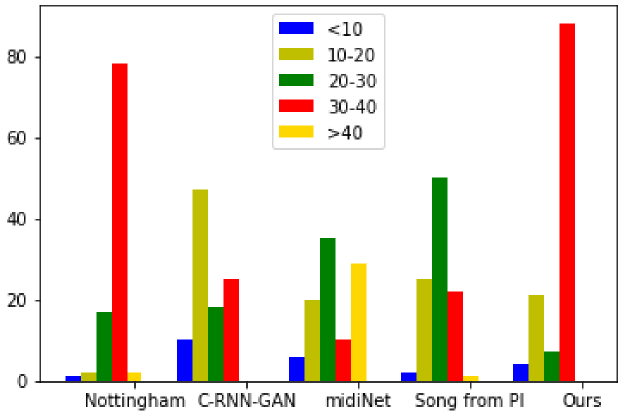

Figure 6: Tone histograms of 100 segments within the five bins to show the number of tones(notes) types appears in all segments

Tone span [9] measures the gap between the highest and the lowest tone values (in frequency, Hz) within a segment. Extreme high or low gap values indicate an unreasonable melody. Fig. 7 shows that in most cases, the distance between highest and lowest frequency value in Nottingham and stack_chord_gan tend to be $600Hz \sim 700Hz$. MidiNet and Song from Pi also show strong ability to create notes most of which belong to the range $600Hz \sim 700Hz$. And few keys produced by these 3 models drop in the abnormal notes frequency area($< 500Hz$ or $> 800Hz$). Unlike this, the C-RNN-GAN generated notes whose tone span distributed uniformly in all sections. We suggest that C-RNN-GAN generates notes without the constraints of chords progression, thus makes the results more free but unreasonable from the view of music theory. Melodies from Nottingham dataset and stack_chord_gan are definitely conditioned by the basic theory of chords, so the distribution of tone-span forms almost the same pattern to each other.

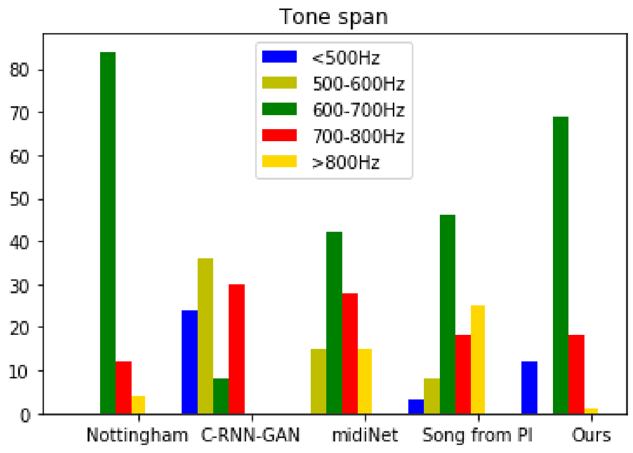

Figure 7: Tone span histograms of 100 segments within the five bins of frequency

Scale estimation [9] predicts the best matching scale type for the target melody. Usually, the standard scales include major, natural minor, harmonic minor. We calculate the probability for each category with $so\ f\ tma$. We classify a segment as unknown if it is not suitable to be claimed as a melody (probability less than 0.7). Fig. 8 shows that 71% of the melodies generated by C-RNN-GAN cannot be accepted, while the unknown ratio for the others is much lower. Melody generated by Song from Pi gets the best performance in this test by taking the scale as a forced condition in training process. We argue that although the proposed model does not explicitly specify the scale attribute of each sample, the scale pattern hidden in the data features is absolutely grasped by the time-sequential model: Bi-LSTM, which keeps the current note in accordance with both previous and later scale of the whole melody. Thus, the proposed model gets almost the similar result with Song from PI, which is much better than others.

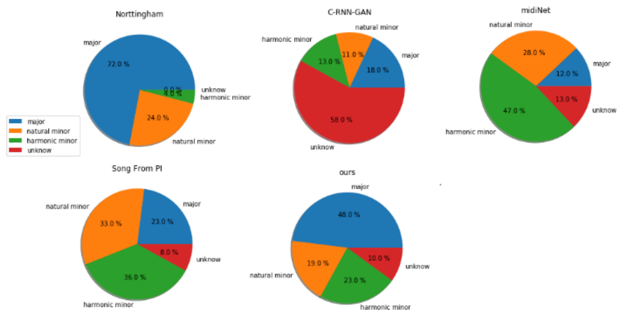

Figure 8: Melody classification; the fewer the unknowns, the better.

5.3 Acoustic perception

Music is the art of manipulating sounds. It tries to present human's perception with pleasant in a harmonious way. However, its appeal depends upon some subjective factors of human. Music perception therefore depends upon both the sound itself and the person that hears it. In this section, we evaluate the quality of generated melody with a human perception survey involving 14

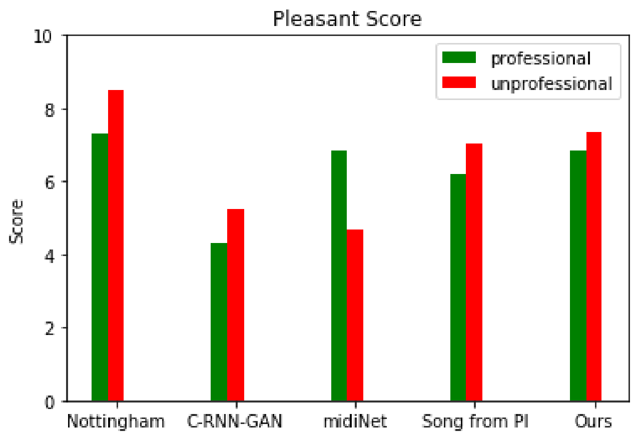

Figure 9: Estimation of samples' pleasant from the volunteers composed of professional group and unprofessional group. Higher score represents the better tuneful melody.

audience. Four of them are professional musician with plenty of basic music knowledge, the others selected at random from non-professional people. We randomly prepare 50 clips, each consists 100 notes. The left 40samples are from other baselines averagely. Each volunteer is asked to listen to these samples once, and then tries to rate these samples in terms of if they: 1) sound pleasant; 2) have harmonic chords with notes in each segment (for professional volunteers only); 3) have harmonic chords progression over the whole structure(for professional volunteers only). Since C-RNN-GAN outputs no chords and doesn't process notes with the help of chords, we omit this source from the baseline sets when the last 2 items are checked which are related to chords generation. All items are rated in 10-point scale. The survey of the human perception study is presented in Fig. 9, 10. For the pleasant test(Fig. 9), the proposed model obtain higher scores both in professional and unprofessional's view. Both group tend to be more satisfy with the music produced by MidiNet, Song from PI and proposed model rather than by the C-RNN-GAN. The upper part of Fig. 10shows the evaluation of the harmony between chords and notes. We find that our method obtains almost the same score as the other baseline(0.38 higher than MidiNet, 0.09 lower than Song from PI).And the lower part demonstrates the chord progress score. We see our method has stronger ability to grasp the harmonious chords progression than other methods.

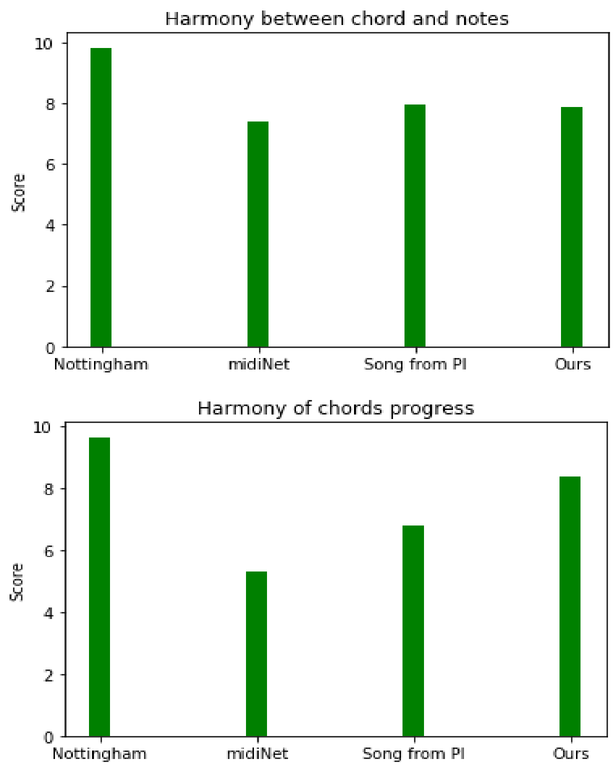

Figure 10: Evaluation of samples' quality related to chords. We test samples in 2 aspects: harmony between chords and notes(illustrated in the left part), harmony of chords progress(illustrated in the right part). The higher score shows the better harmonious result.

6. CONCLUSION

We propose a stack adversarial procedure to generate melody in an automatic way. To make the generated melody sound more pleasant and realistic, we first learn the chords via a chord GANs to outline the melody, then we train a melody GANs conditioned on the chords learned from the previous step to complete the music generation. Our method is shown to perform better than existing ones.

7. REFERENCES

[1] M Abboud, B Németh, and JC Guillemin. 2012. Modeling temporal dependenciesin high-dimensional sequences: Application to polyphonic music generation andtranscription.Chem. Eur. J18, 13 (2012), 3981–3991.

[2] Hang Chu, Raquel Urtasun, and Sanja Fidler. 2016. Song from PI: A musicallyplausible network for pop music generation.arXiv preprint arXiv:1611.03477(2016).

[3] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2018.MuseGAN: Multi-track sequential generative adversarial networks for symbolicmusic generation and accompaniment. InThirty-Second AAAI Conference onArtificial Intelligence. 34–41.

[4] Douglas Eck and Juergen Schmidhuber. 2002.A First Look at Music Composi-tion using LSTM Recurrent Neural Networks. Istituto Dalle Molle Di Studi SullIntelligenza Artificiale.

[5] Jon Gauthier. 2014. Conditional generative adversarial nets for convolutionalface generation.Class Project for Stanford

CS231N: Convolutional Neural Networksfor Visual Recognition, Winter semester2014, 5 (2014), 2.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley,Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarialnets. InAdvances in neural information processing systems. 2672–2680.

[7] Christian Ledig, Lucas Theis, Ferenc Husz ár, Jose Caballero, Andrew Cunning-ham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, ZehanWang, et al.2017. Photo-realistic single image super-resolution using a generativeadversarial network. InProceedings of the IEEE conference on computer vision andpattern recognition. 4681–4690.

[8] Mehdi Mirza and Simon Osindero. 2014. Conditional Generative AdversarialNets.Computer Science(2014), 2672–2680.

[9] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks withadversarial training.Constructive Machine Learning Workshop on NIPS(2016).

[10] Eita Nakamura, Emmanouil Benetos, Kazuyoshi Yoshii, and Simon Dixon. 2018.Towards complete polyphonic music transcription: Integrating multi-pitch detec-tion and rhythm quantization. In2018 IEEE International Conference on Acoustics,Speech and Signal Processing (ICASSP). IEEE, 101–105.

[11] Eita Nakamura, Kazuyoshi Yoshii, and Shigeki Sagayama. 2017. Rhythm tran-scription of polyphonic piano music based on merged-output HMM for multiplevoices.IEEE/ACM Transactions on Audio, Speech, and Language Processing25, 4(2017), 794–806.

[12] Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional imagesynthesis with auxiliary classifier gans. InProceedings of the 34th InternationalConference on Machine Learning. JMLR. org, 2642–2651.

[13] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, OriolVinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu.2016. Wavenet: A generative model for raw audio.arXiv preprint arXiv:1609.03499(2016).

[14] Scott E. Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele,and Honglak Lee. 2016. Generative adversarial text to image synthesis.Interna-tional Conference on Machine Learning(2016), 1060–1069.

[15] Ricardo Scholz and Geber Ramalho. 2008. COCHONUT: RECOGNIZING COM-PLEX CHORDS FROM MIDI GUITAR SEQUENCES. InInternational Conferenceson Music Information Retrieval. 27–32.

[16] Bob L Sturm, Joao Felipe Santos, Oded Ben-Tal, and Iryna Korshunova. 2016.Music transcription modelling and composition using deep learning.arXivpreprint arXiv:1604.08723(2016).

[17] Yiming Wu and Wei Li. 2018. Music Chord Recognition Based on Midi-TrainedDeep Feature and BLSTM-CRF Hybird Decoding. In2018 IEEE InternationalConference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 376–380.

[18]Li Chia Yang, Szu Yu Chou, and Yi Hsuan Yang. 2017. MidiNet: A ConvolutionalGenerative Adversarial Network for Symbolic-domain Music Generation.arXivpreprint arXiv:1703.10847(2017).

[19] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: SequenceGenerative Adversarial Nets with Policy Gradient.. InAAAI. 2852–2858.

[20] Han Zhang, Tao Xu, and Hongsheng Li. 2017. StackGAN: Text to Photo-RealisticImage Synthesis with Stacked Generative Adversarial Networks. In2017 IEEE International Conference on Computer Vision (ICCV). IEEE, 5908–5