2022

NBP 2.0: Updated Next Bar Predictor, an Improved Algorithmic Music Generator

Belinda M. Dungan

Proceso L. Fernandez Jr

Philippine Journal of Science
151 (5): 1893-1905, October 2022
ISSN 0031 - 7683
Date Received: 22 Mar 2022

NBP 2.0: Updated Next Bar Predictor, an Improved Algorithmic Music Generator

Belinda M. Dungan[1]* and **Proceso L. Fernandez**[2]

[1]College of Computer Science, Don Mariano Marcos Memorial State University
South La Union Campus, Agoo, La Union, Region I 2504 Philippines
[2]Department of Information Systems and Computer Science,
Ateneo De Manila University, Quezon City, National Capital Region 1108 Philippines

Deep neural network advancements have enabled machines to produce melodies emulating human-composed music. However, the implementation of such machines is costly in terms of resources. In this paper, we present NBP 2.0, a refinement of the previous model next bar predictor (NBP) with two notable improvements: first, transforming each training instance to anchor all the notes to its musical scale, and second, changing the model architecture itself. NBP 2.0 maintained its straightforward and lightweight implementation, which is an advantage over the baseline models. Improvements were assessed using quantitative and qualitative metrics and, based on the results, the improvements from these changes made are notable.

Keywords: algorithmic melody generator, NBP, scale-based data transformation

INTRODUCTION

Music generation has been an active domain in machine learning, spawning a variety of algorithms used. The approaches in algorithmic music generation generally differ in the objective of the composition, the architecture used, and the strategies employed. Even the representations of the inputs and outputs differ. These approaches are considered guidelines for classifying and identifying existing music generation algorithms (Briot 2021; Nierhaus 2009).

Three factors must be considered to create a music generation system capable of mimicking human-composed melodies. First, one must determine whether the data representation of the dataset is signal-based or symbolic. Signal-based instances are thought to be closer to the natural form, which is represented through waveforms and spectra. A few studies have used .wav (McAllister and Gambäck 2022), .mp3 (Goren *et al.* 2022), and AIFF – which are the common formats under this category. However, signal-based representation is often not used in studies because each instance may contain 16,000 samples per second, making the training difficult (Lee *et al.* 2017). In contrast, symbolic-based representation contains semantic meaning and is commonly used by researchers. This type of representation includes the use of MIDI (Hung *et al.* 2021; Walter *et al.* 2021; Ren *et al.* 2020), piano rolls (Minu *et al.* 2022; Hoshi *et al.* 2022), lead sheets (Czyż and Kędziora 2021; Choi *et al.* 2021; Pachet *et al.* 2013), and text files (Yang 2021; Sabitha *et al.* 2021).

Second, it is necessary to consider how data is transformed. Since most studies use symbolic-based data, pitches are encoded using a variety of techniques, such as tokens corresponding to the 88 keys on a piano (Majidi and Toroghi 2022), MIDI note numbers (Choi *et al.* 2021; Guo *et al.* 2021; Walter *et al.* 2021), matrices equivalent to the 128 MIDI note numbers (Yang *et al.* 2017), among others.

*Corresponding author: belinda.dungan@obf.ateneo.edu

Although these representations may be augmented with transposition, the transformations are not transposition invariant.

Last, the impact of the machine learning model and the specific architecture on the algorithmic music generator must be considered. Over the years, neural networks such as multilayer perceptron (Kurniawati *et al.* 2020), recurrent neural networks (Keerti *et al.* 2022; Czyż and Kędziora 2021; Jiang *et al.* 2019; Yu *et al.* 2021), variational autoencoders (Grekow and Dimitrova-Grekow 2021; Chen *et al.* 2020; Pati *et al.* 2019), restricted Boltzmann machine (Lyu *et al.* 2015), reinforcement learning (Liu *et al.* 2021; Jiang *et al.* 2020; Jaques *et al.* 2017), and generative adversarial network (GAN) (Huang *et al.* 2022; Kumar *et al.* 2019) have been explored. Similarly, a variety of compound architectures such as conditional GAN (Yang *et al.* 2017), a combination of the recurrent neural network and GANs (Li *et al.* 2021; Yu *et al.* 2021), and hybrid generative models (Dong *et al.* 2018; Yamshchikov and Tikhonov 2020) were used.

GAN models have proven to be useful in visual computing (Gonog and Zhou 2019; Wang 2017); however, very lately, they have also been used in generating melodies. Although significant difficulties in training GAN – which include mode collapse, non-convergence, and hyper-parameterization – were encountered, they may be overcome with stability solutions (Saxena and Cao 2021). Training time could also be a challenge because basically two models, the generator and the discriminator, are being trained.

Conditioning or conditional architecture is another type of generative model that uses extra information to modify the training in such a way that this extra information, or so-called "conditioners," may influence the generation process (Guo *et al.* 2021; Roberts *et al.* 2018; Yang 2017; Genchel *et al.* 2019). These conditioners could either be a separate input integrated in-between the model's layers, or they may be embedded in the input before it is used by the model. Chord progression (Choi *et al.* 2021; Yang *et al.* 2017; Roberts *et al.* 2018), musical genre or style (Colombo *et al.* 2017; Mao *et al.* 2018; Liang 2016), bass line or beat structure (Makris *et al.* 2017) are examples of conditioners. Conditioning could either be local or global depending on whether the conditioning is shared in all or on specific timesteps only (Guo *et al.* 2021). Choi *et al.* (2021) claimed that conditioning had an impact on the generated melodies in their study where chord progressions were utilized as conditioners, which is evident in the harmony criteria on the subjective evaluation. In contrast, Genchel *et al.* (2019) found that chord conditioning on folk tunes was unnecessary since notes might remain on the scale even without conditioning. With the numerous implementations, the same objective is aimed, and that is to algorithmically generate music that can mimic human composition.

In this paper, we present NBP 2.0, an updated version of NBP 1.0 (Dungan and Fernandez 2020) with two significant contributions. First, we modified the data transformation by "clipping" all pitches to the scale – that is, representing each pitch relative to the data instance's musical scale. Given that data transformation is a contributing factor in generative systems, this technique in data transformation could be an option for further improving the composed melodies. Aside from allowing machine models to learn the melodic contour (and, thus, being transposition invariant), it also includes scale information for the data instance. Second, NBP 2.0 includes a simple and lightweight generative model that predicts pitch and duration based on the previous bar's information. It is an upgraded version of NBP 1.0, which already yielded good results according to objective and subjective evaluations.

BACKGROUND OF THE STUDY

MidiNet (Yang *et al.* 2017)

Data representation. One instance of the dataset contains exactly eight bars and in each bar, the smallest allowable note is the 16th note. As rest notes are not represented in this model, previous (non-rest) notes were precisely prolonged to deal with the rest notes. Tracks are normalized into two octaves and are transposed to one or more keys to augment the data. Notes are quantized to the 16th note and represented using a binary h x w matrix, where h specifies the number of possible pitches and w denotes timesteps. Figure 1 depicts a visual representation of the binary matrix, with pixels representing entries of ones. Even though this format encompasses all octaves, it is not transposition invariant.

Architecture. The MidiNet, as shown in Figure 2, features a GAN-based architecture that is specifically built on DCGAN (Radford *et al.* 2015). The generator uses four layers of transposed convolution, with 1D and 2D conditioners added to each layer. The 1-D conditioner is a binary vector of size 13 representing the previous bar's chord, and the 2D conditioner is a 128 x 16 matrix representing the previous bar. The discriminator is a convolutional neural network with two convolutional layers joined to a fully connected layer. The 2D and 1D conditioners are added between the discriminator's convolutional layers. The generator was trained twice to reduce the possibility of the discriminator outperforming the generator. Human evaluators assessed the MidiNet

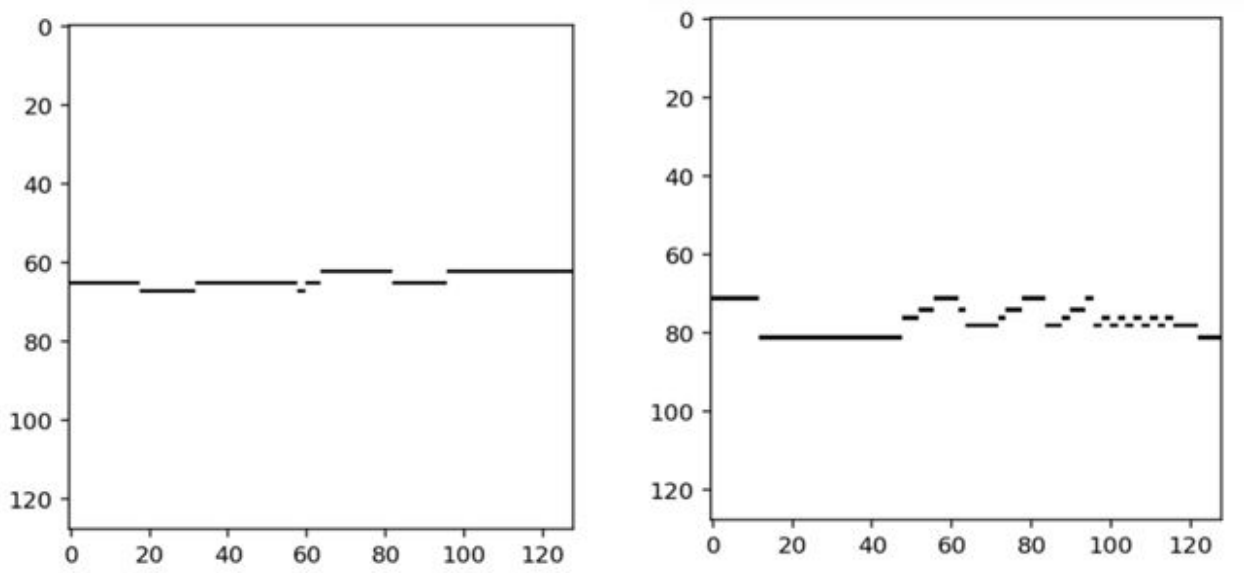

Figure 1. Example of a MidiNet dataset instance representation.

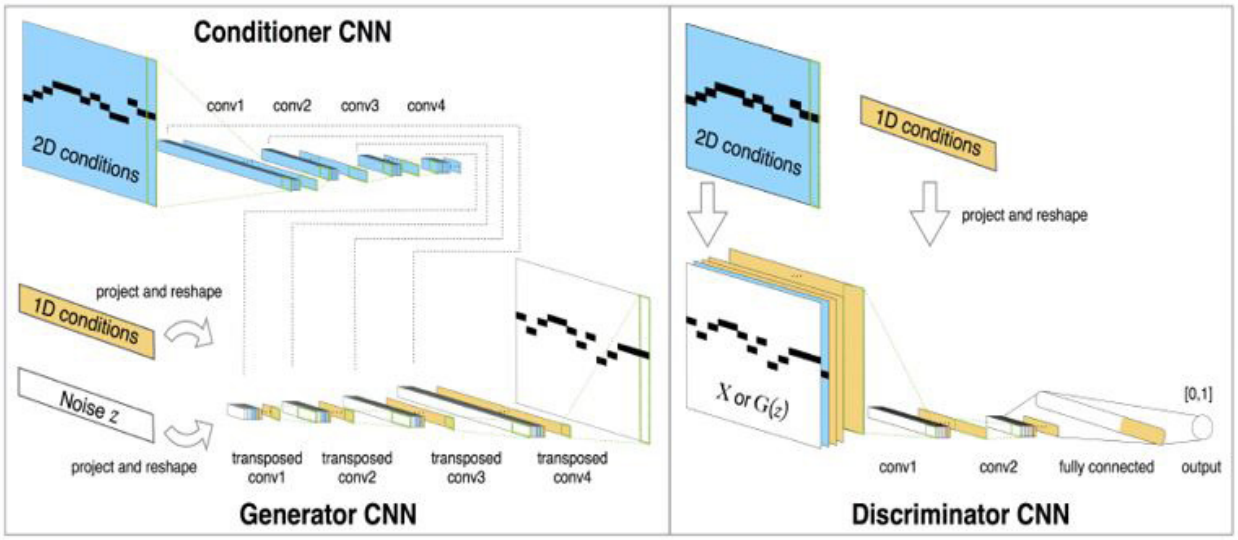

Figure 2. MidiNet architecture (Yang *et al.* 2017).

output in terms of how pleasing, interesting, and realistic the generated sounds are. In some criteria, MidiNet outperformed Melody RNN from Google.

NBP 1.0 (Dungan and Fernandez 2020)

Data representation. Pitch and duration are represented independently by a 16-element vector corresponding to one bar. The pitch is represented as a relative number, with the first pitch set to zero and the subsequent values equal to the difference between the current and previous note numbers. The duration is represented using binary values: 1 indicates that the pitch is played and 0 indicates that the pitch is sustained. In contrast to MidiNet, the data representation in NBP 1.0 is transposition invariant and can distinguish between sustained and hammered notes (Dungan and Fernandez 2020).

Architecture. The NBP 1.0 (Dungan and Fernandez 2020), whose overall flow is depicted in Figure 3, is a simpler type of generative model that produce melodies iteratively based on the preceding bar. It is trained using a set with many instances of two adjacent bars – the first bar acts as input and the second bar as the target output. In generating the output melody, a seed bar or the first bar in a song

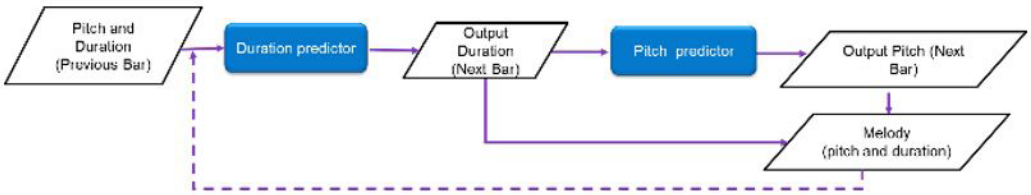

Figure 3. NBP 1.0 (Dungan and Fernandez 2020).

from the held-out validation set is utilized as the primary input when creating a new melody. The second bar is then generated by the trained model. The second bar will then serve as input to create the third bar. This iteration process is repeated until the desired number of bars is reached.

METHODOLOGY

The stages and the procedure in the creation of the NBP 2.0 are as follows: acquisition of melody, data transformation, model training, and evaluation. The output of NBP 2.0 was compared with the baseline models (NBP 1.0 and MidiNet) using objective and subjective metrics.

Acquisition of Melody and Chord Progression Dataset

The dataset, which contains 500 pop songs, was acquired from Theorytab[1]. The same dataset was used to train the two baseline models. Each song is made up of eight bars, and each bar has exactly one chord. As an example, a song excerpt from the official Game of Thrones soundtrack is presented in Figure 4. It consists of eight bars in sheet format, with the first bar highlighted. The first bar is made up of the following beats: two successive eighth notes (half beat), a quarter note (1 beat), two more eighth notes (half beat), and finally a quarter note (1 beat). For simpler and uniform implementation, we removed from the initial dataset all songs that contain 32nd notes in order to ensure that the smallest note is the 16th note. This way, we can use the same set of data previously used in our baseline studies. Excluding songs with 32nd notes, the dataset was reduced to only 460 songs, of which 410 were used for training purposes and 50 were held out for the generation phase. To handle the rest notes, each was eliminated by simply prolonging the previous note. This was also the procedure followed by the baseline studies when dealing with rest notes. It was also imposed in all instances of the dataset to have 1 chord per bar. As presented in Figure 5, the chord for the first bar and until the eighth bar is C# major.

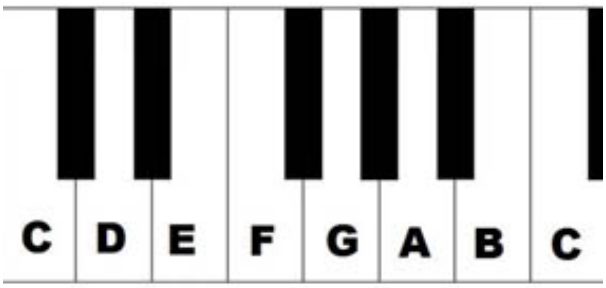

Figure 4. The first eight bars of the official soundtrack of Game of Thrones, as represented in lead sheets.

Data Representation

The data flow of NBP 2.0 is essentially the same as with NBP 1.0, which uses the previous bar as input and generates the next bar as output. In the new version, each instance of the dataset is represented to include information about the scale. Here is an example of the transformation: If a bar's pitch values are A4, A4, E5, C5, and C5 and it is played in the C5 Major scale, we assign C to zero, and each of the other notes a value based on their relative position (*i.e.* the number of half steps below or above). We utilize Figure 6 to show the pitch's relative location. Starting with the first note, we count the number of semitones from A4–C5. From B, A#, and A, that would be three half-steps downward, so our relative value is 3.

The numerical value along with the pitch represents the specific octave. An octave is a distance between one note (like C) and the next note bearing the same name (the next C that is either higher or lower). In our example, our note

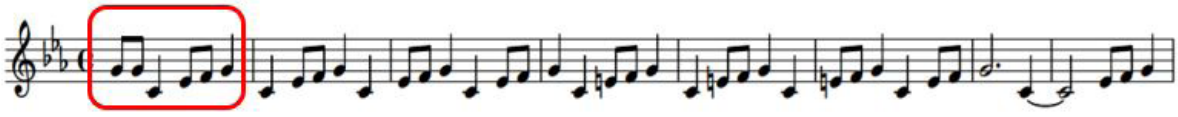

Figure 5. Corresponding chord sequence of the official soundtrack of Game of Thrones, as represented in lead sheets.

https://www.hooktheory.com/theorytab

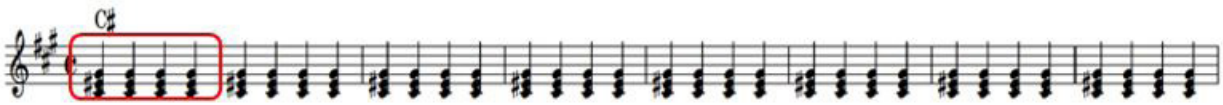

Figure 6. C Major scale formula as the basis in the relative position

is A4, and the base is C5. It means that we moved down an octave; and so, the sign is negative. Any pitch that uses C5 as its basis must employ this translation. As a result, the note sequence A4, A4, E5, C5, and C5 would be transformed as –3, –3, 4, 0, 0. This representation retains the property of being transposition invariant.

In representing duration, we use the symbol 1 when the pitch is played and 0 when the pitch is sustained. We assign a value to each time step, and because the smallest allowable note duration is the sixteenth note, then the duration representation is made up of 16 values. If the pitch and duration in our example are A4, A4, E5, C5, C5, and quarter, quarter, eighth, quarter, and eighth notes, we can describe the pitch and duration using a 16-element vector, as illustrated in Figure 7. If we are only referring to the pitch vector, it gives the impression that A4 is sustained. With the help of the duration vector, the model has the information that the said pitch was, indeed, hammered twice. Note that the current representation does not allow for rests to be handled. The duration representation may, in a further study, be enhanced to include –1 to indicate that a note is at rest. However, this option was excluded in the experiments to fairly compare the output of the NBP models against the MidiNet, which does not have rest notes.

Model Training and Generation of Melody

The NBP 2.0 is still a two-step process, where one trained model determines the pitch sequence, while another model determines the duration sequence of the next bar. A major change in the architecture was done to the previous model's input, as well as to the order of prediction. In the new version, the duration was first predicted using the pitch and the duration of the previous bar. This is to reflect the idea that a pitch sequence can affect the duration sequence (for example, if the note pitches low, then note duration tends to be long). To accomplish this, the value of the pitch vector is simply concatenated to the value of the duration vector. The new dimension of the input in predicting the next bar's duration, therefore, is a vector of size 32 consisting of the 16-element vector pitch (P_{prev}) and the 16-element vector duration (D_{prev}), both from the previous bar. This is used as input to the duration predictor in Figure 8. The output of the classification model is a 16-element vector (D_{next}) consisting of either 1 or 0 values, representing the duration vector of the next bar. Since only binary numbers were used to represent the duration, a classification model was used for the generator.

To predict the pitch of the next bar, a regression model was used. The detailed presentation of the model can be gleaned from Figure 9. The predicted duration (D_{next}) is concatenated to the previous 32-element vector, yielding a vector of size 48. The output of the regression model is a 16-element vector (P_{next}) with values that represent the pitch sequence of the next bar. Since the output may be real numbers, values are rounded off. We opted for a regression model instead of a multi-class model because first, the output is not restricted to a particular number of octaves, which gives the model liberty to jump from

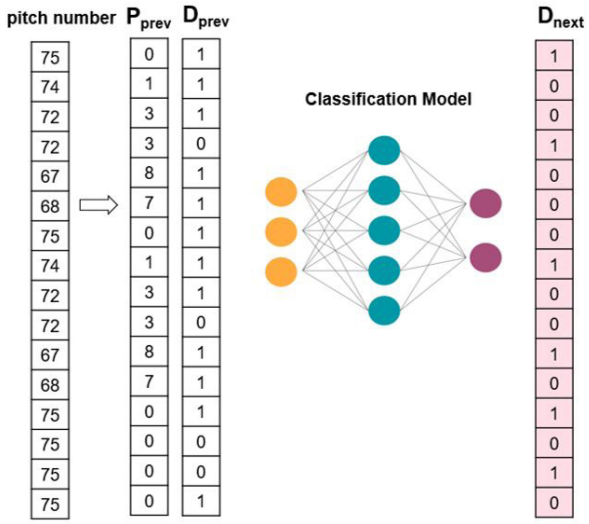

Figure 8. Detailed version of the duration predictor of NBP 2.0.

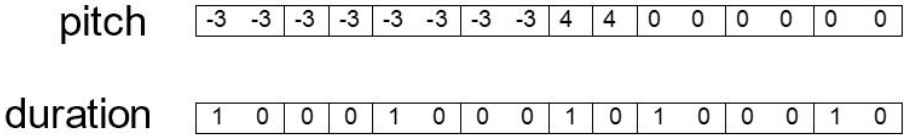

Figure 7. Input representation for pitch and duration.

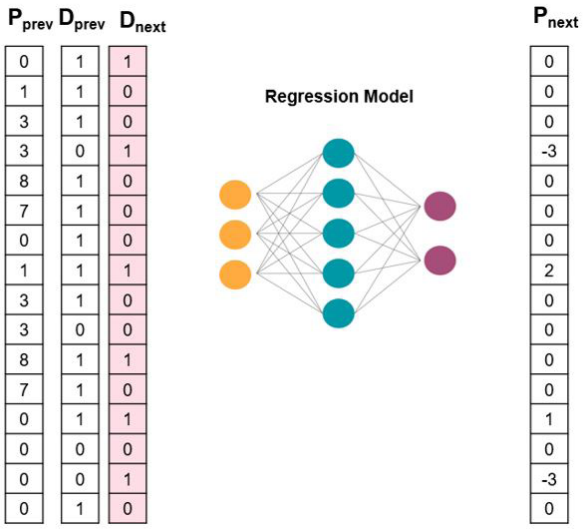

Figure 9. Detailed version of the pitch predictor of NBP 2.0.
*Legend:
[P*pev*] previous pitch
[D*prev*] previous duration
[P*next*] next pitch
[D*next*] next duration

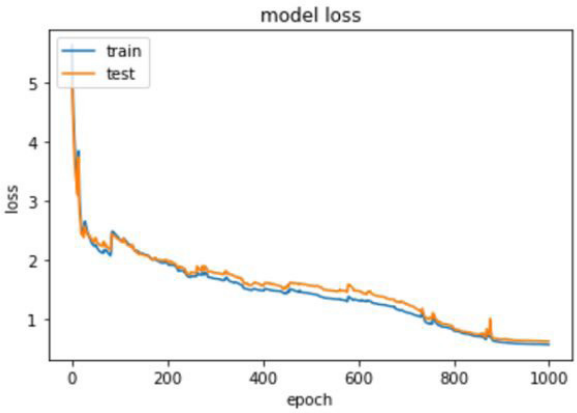

Figure 10. Training loss for the duration predictor.

one octave to another, and second, the encoding of the 16-element pitch vector output is more lightweight with the use of an integer instead of a one-hot encoding of the multiple classes for each element of the vector.

In the training phase, the pitch and duration predictor may be trained simultaneously since the requisite information are in the training dataset already. However, in the generation of new melodies, the duration, and pitch prediction was done in sequence because of the dependencies described earlier. In the production of a melody, a first bar acting as seed bar was randomly selected from the held-out set, and this bar was used as an initial previous bar to generate the second bar. The cycle in the generation was done seven times, making a song composed of seven bars. It is noteworthy to mention that the seed bar was discarded and was not included in the generated melody that was evaluated later.

For easier reference, Table 1 shows a comparison of NBP 2.0 against NBP 1.0 with respect to data encoding, the input dimension, the models used, as well as the output dimensions.

Model Specification

The duration predictor used a classification model, particularly a deep neural network. The input layer has 32 nodes and is followed by two hidden layers with 16 and 8 nodes respectively. An output layer with 16 notes completes the neural network architecture. L2 regularization was used in the first hidden layer, relu activation function for the hidden layers, and softmax activation function on the output layer. During training, batch sizes of 32, 64, and 128 were experimented on, but the final model used a batch size of 128 because it yielded the best results. The pitch predictor, on the other hand, is a decision tree regression model. For hyperparameter

Table 1. Comparison of NBP 1.0 and NBP 2.0.

	NBP 1.0	NBP 2.0
Pitch data encoding	First note of a bar is set to zero, and values thereafter is the relative distance (i.e., difference in the MIDI note numbers) of a pitch to its immediate predecessor pitch	All pitch values are anchored to the scale of the song, i.e., a pitch is represented by its relative distance to the scale note of the song.
Input	For Pitch Predictor: Previous Pitch (16-element vector) For Pitch Predictor: Previous Duration (16-element vector)	For Pitch Predictor: Previous Pitch + Previous Duration + Next Bar Duration (48-element vector) For Duration Predictor: Previous Pitch + Previous Duration (32-element vector)
Models used	Pitch Predictor: Decision tree regression model Duration Predictor: Decision tree classification model	Pitch Predictor: Decision tree regression model Duration Predictor: Deep neural network classification model
Output	Pitch Predictor: Next Bar Pitch (16-element vector) Duration Predictor: Next Bar Duration (16-element vector)	

tuning, we conducted a grid search to at least determine a very good set of hyperparameters for the decision tree.

Based on the results of the grid search, the maximum leaf node was set to 80, and the maximum depth of the tree is set to 3.

Evaluation

Objective and subjective analyses were used to investigate the improvements on the changes made in comparison with the baseline models. The objective analysis includes the comparison of the training time, dissimilarity score, weighted average of the out-of-scale notes, and MG evaluation (Yang and Lerch 2020). In the MG evaluation, we compared the different models using the pitch count, pitch range, note count, average pitch interval, and pitch histogram.

The ShapeH algorithm (Urbano *et al.* 2010) was utilized to compute the dissimilarity score. Using the ShapeH algorithm, a melody is compared to, and the similarity score computed against, each melody from the dataset. From the different scores, the candidate melody from the dataset having the highest similarity score would be the closest melody. Since we are interested in the dissimilarity score, we get the difference between the similarity score with 1. From the 50-generation set, we calculated the average dissimilarity score to gauge the overall performance of the different generative models.

Out-of-scale notes are pitches that fall outside of the collection of pitches traditionally associated with a certain scale. Such are often acceptable in music composition and serve as (what musicians refer to as) embellishments or accidentals that make the composition more interesting. However, in the case of pop songs, accidentals are rarely used. To analyze the model's output on how near the composition is based on the genre attribute, we computed the (duration-) weighted average of the out-of-scale notes. We tallied the number of out-of-scale note values in each of the model's generated melodies, divided it by the total number of notes with the same values, and multiplied it by the corresponding weight. We assigned the numbers 4, 2, 1, 0.5, and 0.25 for whole, half, quarter, eighth, and sixteenth notes, respectively, for the specified weight.

For the MG evaluation, pitch count and pitch range were selected to describe the generated pitch of the different models in comparison with that of the dataset, whereas note count and average pitch interval describe the produced note durations.

Subjective analysis was also conducted using human evaluation. The listening evaluation was conducted using three criteria: "how realistic," "how interesting," and "how pleasing," which are the same criteria used in MidiNet and NBP 1.0. A four-point Likert scale was used to grade the human survey. Thirty (30) persons were requested to participate in a survey to see if the three models (MidiNet, NBP 1.0, and NBP 2.0) can produce melodies that are pleasing, interesting, and realistic. Eleven (11) of the respondents are professionals who have a strong grasp of music theory and can play at least one musical instrument. Each is either taking up or has finished a master's degree in music education. The other 19 individuals are casual listeners. Each of the evaluators listened to a total of 30 different music files, comprising of 10 algorithmically generated songs from each of the three models. Further details about these files are discussed later. The hearing evaluation was conducted for four days of less than 15 min/d to avoid weariness among the participants.

For the generation of the melodies that underwent human evaluation, five seed bars were randomly selected from the held-out partition of the dataset. Each seed bar was then used to generate, for each of the three models, one 8-bar melody without chord progression, and another version of the melody – this time with chord progression included. Thus, there are 10 songs generated for each of the three models, for a total of 30 songs overall.

Since the songs generated by the three models have the same set of seed bars, we decided to remove the seed (first bar of each song) and only include in the evaluation test the actual seven bars produced by the respective models.

This will ensure that the evaluators are not influenced by the seed, which is a human-composed bar.

RESULTS AND DISCUSSION

Training Setup of the Different Models

The hyperparameters used by the different models are presented in Table 2. The hyper-parameters for MidiNet were based on the default values in the Github repository, whereas the hyperparameters for NBP 1.0 and NBP 2.0 were based on grid search and on our own initial listening evaluation.

In the duration predictor, we monitored the training and validation loss along with hyperparameter tuning. Based on the experiments conducted, training loss seems to stabilize at epoch 1,000 and tends to overfit if training is prolonged. We, therefore, decided to stop the training on the said epoch.

Training the pitch predictor is slightly different because we did not fully rely on the computed root mean squared error value (rsme). This is because intervals in pitch are crucial in music composition. Music theory in one hand

Table 2. Comparison of the hyperparameters used in training the different models.

MidiNet	NBP 1.0	NBP 2.0
Generator: Transposed CNN Discriminator: CNN Conditioner: CNN	Pitch Predictor: Decision Tree Regressor Maximum leaf Nodes: 50 Maximum depth of tree: 5	Pitch Predictor: Decision Tree Regressor Maximum leaf Nodes: 80 Maximum depth of tree: 3
Epoch: 20 Learning Rate: 0.0002 Beta 1:0.5 Optimizer: Adam Batch size: 72	Duration Predictor: Decision Tree Classifier Maximum leaf Nodes: 60 Maximum depth of tree: 3	Duration Predictor: Deep Neural Network Learning rate: 0.0001 Epoch: 1000 Batch size: 128 Optimizer: Adam

considers consecutive notes having an interval of 2 sounds better than an interval of 1 because a leap of 1 semitone would sound dissonant. rsme, on the other hand, does not work this way: the less the gap between the predicted and the ground truth, the better.

In order to resolve this issue, we did not solely rely on the computed rsme but we periodically listened to the output melody of the model. Though the rsme value was not totally discarded, we applied a broader tolerance in monitoring the value to give us a hint to whether the generated output is worth listening to.

Training Time

The training time was recorded to assess the efficiency of the various models in terms of resource use. In recording the training time, saving of the output and other unnecessary displays were disabled. Table 3 compares the time of execution recorded during the training of the three models. It is natural that NBP 2.0 consumes more time than NBP 1.0 due to the change in architecture, as well as the required input for the regression and classification models. Nonetheless, the training time of NBP 2.0 is only roughly 23% of the entire training time of the MidiNet.

Table 3. Training time is consumed by the different generative models.

Model	Training time (in s)
MidiNet	201.00
NBP 2.0	46.353
NBP 1.0	0.0679

Dissimilarity Score

The dissimilarity score was used to objectively measure if the models demonstrated some creativity. Comparing the three models shown in Table 4, MidiNet has the highest dissimilarity, whereas NBP 2.0 was a close second. NBP 2.0, if compared to NBP 1.0, has an average score that is about 0.230 higher (an improvement of about 35%).

Table 4. Comparison of the average dissimilarity scores of the output of the different models.

Model	Average dissimilarity score
MidiNet	0.960
NBP 1.0	0.685
NBP 2.0	0.923

Even if the three models' recorded dissimilarity scores are relatively high, indicating some good creativity, we cannot conclude that the high rating of dissimilarity could guarantee a pleasing melody, so we checked the performance of each model using other metrics.

Out-of-scale Notes

Inspecting further the quality of the generated melodies, we recorded how many from the generated set produced out-of-scale notes. As shown in Table 5, the NBP 2.0 recorded the least number of out-of-scale notes among the three models. This model generated unique melodies, as evident in the dissimilarity score but still maintained to produce notes that are mostly inside the musical scale. In contrast, MidiNet tends to generate notes that are outside the scale. This would partly explain why it garnered the highest dissimilarity score, since the songs in the dataset have very few notes that are outside the scale. The dataset, which involved pop songs, only contained a weighted average of 2.09% out-of-scale notes, leaving a significant gap from any of the three models. Thus, further investigation on reducing the number of out-

Table 5. The weighted average of out-of-scale notes based on the generation set of the different models.

Model	Average weighted out-of-scale
MidiNet	48.2%
NBP 1.0	65.9%
NBP 2.0	13.2%

of-scale notes was considered. Since out-of-scale notes are embellishments that would make the composition interesting, there is a risk that the output of NBP 2.0 is less interesting than those from the other two models, which can be confirmed by the human evaluation.

MG Evaluation (Yang and Lerch 2020)

Results of the MG Evaluation is presented in Table 6, which includes the pitch count, pitch range, note count, and average pitch interval. Pitch count and pitch range were selected to describe the generated pitch of the different models in comparison with that of the dataset, whereas note count and average pitch interval describe the produced note durations. The improvement in NBP 2.0 is evident from the results of the pitch count and pitch range. If we compare the scores of NBP 2.0 against the baselines, NBP 2.0's scores are closer to the mean and standard deviation of the dataset.

This may mean that predicting pitch after predicting duration indeed helped the model to predict better. Results on the note count and average pitch interval are not notable. However, the results of NBP 2.0 still outperformed MidiNet, along with the said criteria. The pitch histogram in Figure 11 further shows the mean of all the used notes. Visually, we can see those values of the dataset against the NBP 2.0 are nearly on the same level as compared with the other models. From the results, the improvement of the overall output may be attributed to the change in the architecture by predicting first the duration and increasing the input to the pitch predictor.

Human Evaluation

Novice. Combining all the produced output of the different models, we let the novice listeners decide which output they prefer on the different criteria. Listening plainly to the melodies, the respondents favored the output of the NBP 2.0 on the three criteria, as presented in Figure 12, against the MidiNet and NBP 1.0. An additional listening experiment by combining the predefined chord progression with the produced output was conducted. It is worthy to mention that in training MidiNet, the chord was introduced as a conditioner, whereas in the different versions of NBP, it was not used as an additional input.

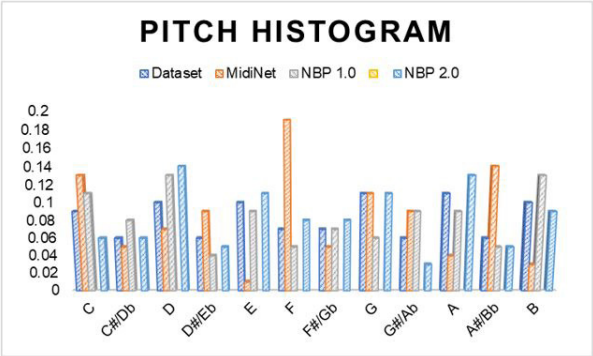

Figure 11. Pitch histogram using the 12-chromatic scale.

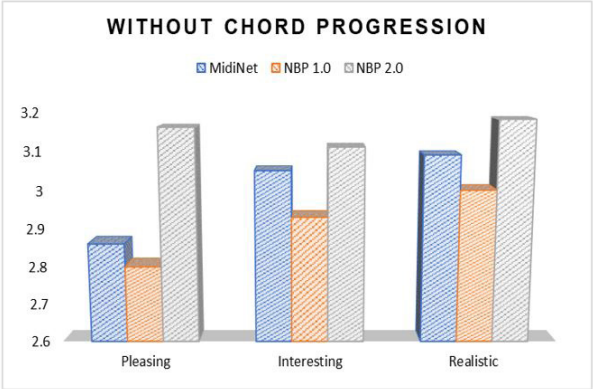

Figure 12. Ratings on the melody of the models, as rated by novice evaluators.

The evaluators showed more enjoyment when chord was included as compared to melodies only, as evidenced by the rise of all the scores shown in Figure 13. NBP 1.0 improved as it scored slightly higher than MidiNet in pleasantness and realistic criteria. The NBP 2.0 still was judged as the best among the three models. Table 7 shows the summary of the paired *t*-test *p*-values of the ratings on the different NBP models compared to the output of the MidiNet. On five out of the six criteria, the computed *p*-values suggested a significant difference, and only the interesting criteria under the case without chord progression displayed no statistical significance. This implies that, generally, novice users prefer the melodies

Table 6. Comparison of the dataset against the output of the different models using MG Evaluation.

Criteria	Dataset		MidiNet		NBP 1.0		NBP 2.0	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
Pitch Count	6.43	1.8	8.32	1.25	4.16	3.25	7.72	1.14
Pitch range	11.4	3.85	19.04	3.73	16.3	20.4	13.9	4.53
Note count	32.8	13.49	41.5	8.06	27.18	21.68	24.58	6.67
Average pitch interval	2.77	1.86	5.39	1.36	1.03	0.89	4.42	1.61

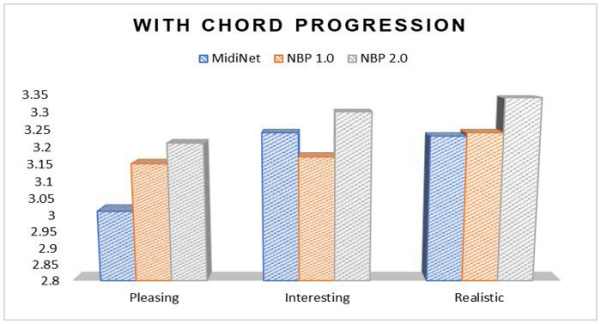

Figure 13. Ratings on the melody with the chord progression of the models, as rated by novice evaluators.

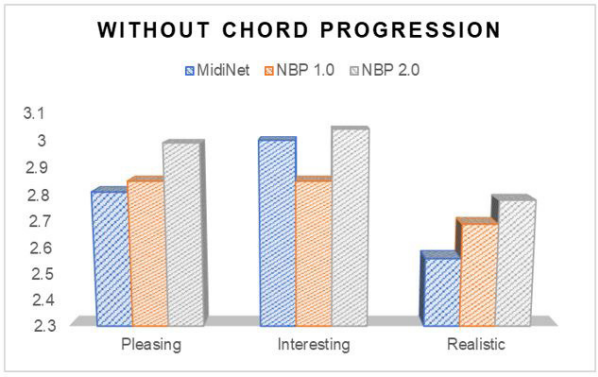

Figure 14. Ratings on the melody of the models, as rated by professional evaluators.

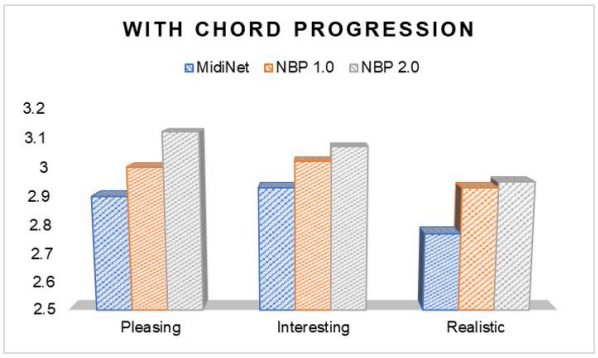

Figure 15. Ratings on the melody with chord progression of the models, as rated by Professional evaluators

produced by NBP 2.0 better than those of the MidiNet.

Professionals. Professional musicians were also asked to evaluate the melodies generated by the models. These evaluators preferred the audios produced by NBP 2.0 for both without (see Figure 14) and with (see Figure 15) chord progression as compared to NBP 1.0 and MidiNet.

Table 8 summarizes the paired *t*-test *p*-values of the ratings of the professional evaluators for the various models compared to the MidiNet. NBP 1.0, as expected, did not pose a viable contender because the values were not statistically significant even when the chord progression was added. The output of NBP 2.0 indicates that the expert evaluators' ratings are statistically significant in all the categories against the MidiNet.

CONCLUSION

In this paper, we presented NBP 2.0, a new variant of the next bar predictor (Dungan and Fernandez 2020) that is an algorithmic music generation model that is lightweight – as evidenced by the computed training time, and can generate pleasing, interesting, and realistic melodies. This improvement entails scale-based data transformation, wherein the pitch representation is relative to the value of the scale. The architecture was also improved, wherein the sizes of the input vectors in the duration and pitch predictor were adjusted to 32 and 48, respectively, from the original size of 16 for both. The duration predictor now requires the pitch and duration values of the previous bar, unlike in the previous version where only the duration of the current bar is required. Furthermore, the pitch predictor uses the pitch and duration of the previous bar concatenated with the duration of the current bar as its input, unlike the previous version that only required the pitch of the previous bar.

The evaluation metrics used in the study revealed general improvements on various metrics. NBP 2.0 outperforms the prior version because of several reasons. It generates more unique melodies, as evidenced by a higher dissimilarity score. When the new architecture was combined with the scale-based data transformation, the

Table 8. Paired t-test comparing the rating of the professional evaluators with and without application of the chord progression for the different output of the models against those from the MidiNet.

Models	Without Chord Progression			With Chord Progression		
	Pleasing	Interesting	Realistic	Pleasing	Interesting	Realistic
NBP 1.0	0.500	0.153	0.299	0.095	0.426	0.902
NBP 2.0	2.13E-10*	0.0009*	2.5E-05*	9.342E-09*	0.000252*	0.002*

number of out-of-scale notes dropped significantly. The two groups of respondents further confirm the NBP 2.0's adequacy as it was rated the highest amid the baseline models (MidiNet and NBP 1.0). This advantage was further demonstrated using a paired *t*-test, which yielded statistically significant findings. Ultimately, considering all the advantages of the NBP 2.0 over the baseline models, we can say that deterministic models – specifically the NBP 2.0 – can be an efficient way to generate unique, pleasant, interesting, and human-like composed melodies.

For further research exploration, it may be worthy to investigate the inclusion of musical rests and shorter duration notes to cover a wider range of possible musical compositions. Additionally, a study similar to this may be conducted, wherein the seed bar is not discarded prior to evaluation so that the models' ability to continue a human-composed seed bar can be assessed. An ablation study may also consider investigating the cause of the improvement from NBP 1.0 to NBP 2.0. Furthermore, other techniques can be explored to fully automate the music generation, such as the automatic generation of a seed bar and chord progression. Finally, the proposed NBP 2.0 or further improved versions can be compared with other recent models for the algorithmic generation of music.

ACKNOWLEDGMENTS

This paper is not possible without the help of: [a] the Commission on Higher Education for the graduate scholarship grant to B. Dungan; [b] the Ateneo community; [c] our respondents, especially the musicians; and [d] friends in the academe who served as our peer reviewers.

REFERENCES

BRIOT JP. 2021. From artificial neural networks to deep learning for music generation: history, concepts, and trends. Neural Computing and Applications 33(1): 39–65.

CHEN K, WANG C, BERG-KIRKPATRICK T, DUBNOV S. 2020. Music SketchNet: Controllable Music Generation *via* Factorized Representations of Pitch and Rhythm. In: Proc. of the 21st Int. Society for Music Information Retrieval Conf., Montréal, Canada.

CHOI K, PARK J, HEO W, JEON S, PARK J. 2021. Chord conditioned melody generation with transformer-based decoders. IEEE Access 9: 42071–42080.

COLOMBO F, SEEHOLZER A, GERSTNER W. 2017. Deep artificial composer: a creative neural network model for automated melody generation. In International Conference on Evolutionary and Biologically Inspired Music and Art. Cham: Springer. p. 81–96.

CZYŻ M, KĘDZIORA M. 2021. Automated music generation using recurrent neural networks. In: International Conference on Dependability and Complex Systems. p. 22–31. Cham: Springer.

DONG HW, HSIAO WY, YANG LC, YANG YH. 2018. Musegan: multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In: Thirty-Second AAAI Conference on Artificial Intelligence.

DUNGAN BM, FERNANDEZ PL. 2020. Next Bar Predictor: an Architecture in Automated Music Generation. In: 2020 International Conference on Communication and Signal Processing (ICCSP). p. 109–113.

GENCHEL B, PATI A, LERCH A 2019. Explicitly conditioned melody generation: a case study with interdependent rnns. In: Proceedings of the 7th International Workshop on Musical Meta-creation, MUME.

GONOG L, ZHOU Y. 2019. A review: generative adversarial networks. In: 2019 14th IEEE conference on industrial electronics and applications (ICIEA). p. 505–510.

GOREN O, NACHMANI E, WOLF L. 2022. A-Muze-Net: Music Generation by Composing the Harmony Based on the Generated Melody. In International Conference on Multimedia Modeling. Cham: Springer. p. 557–568.

GREKOW J, DIMITROVA-GREKOW T. 2021. Monophonic Music Generation with a Given Emotion Using Conditional Variational Autoencoder. IEEE Access 9: 129088–129101.

GUO ZX, MAKRIS D, HERREMANS D. 2021. Hierarchical recurrent neural networks for conditional melody generation with long-term structure. In: 2021 International Joint Conference on Neural Networks (IJCNN). p. 1–8.

HOSHI Y, ORIHARA R, SEI Y, TAHARA Y, OHSUGA A. 2022. Versatile Automatic Piano Reduction Generation System by Deep Learning. In: 2022 2nd International Conference on Advanced Research in Computing (ICARC). p. 66–71.

HUANG W, XUE Y, XU Z, PENG G, WU Y. 2022. Polyphonic music generation generative adversarial network with Markov decision process. Multimedia Tools and Applications. p. 1–21.

HUNG HT, CHING J, DOH S, KIM N, NAM J, YANG YH. 2021. EMOPIA: a Multi-Modal Pop Piano Dataset for Emotion Recognition and Emotion-based

Music Generation. In: International Society for Music Information Retrieval Conference, ISMIR 2021. International Society for Music Information Retrieval.

JIANG N, JIN S, DUAN Z, ZHANG C. 2020. Rl-duet: online music accompaniment generation using deep reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence 34(1): 710–718.

JIANG T, XIAO Q, YIN X. 2019. Music generation using bidirectional recurrent network. In: 2019 IEEE 2nd International Conference on Electronics Technology (ICET). p. 564–569.

JAQUES N, GU S, BAHDANAU D, HERNÁNDEZ-LOBATO JM, TURNER RE, ECK D. 2017. Sequence tutor: conservative fine-tuning of sequence generation models with kl-control. In: International Conference on Machine Learning. p. 1645–1654.

KEERTI G, VAISHNAVI AN, MUKHERJEE P, VIDYA AS, SREENITHYA GS, NAYAB D. 2022. Attentional networks for music generation. Multimedia Tools and Applications 81(4): 5179–5189.

KUMAR K, KUMAR R, DE BOISSIERE T, GESTIN L, TEOH WZ, SOTELO J, COURVILLE AC. 2019. Melgan: generative adversarial networks for conditional waveform synthesis. Advances in Neural Information Processing Systems, Vol. 32.

KURNIAWATI A, SUPRAPTO YK, YUNIARNO E. M. 2020. Multilayer Perceptron for Symbolic Indonesian Music Generation. In: 2020 International Seminar on Intelligent Technology and Its Applications (ISITIA). p. 228–233.

LEE J, PARK J, KIM T, NAM J. 2017. Raw Waveform-based Audio Classification Using Sample-level CNN Architectures. In: Machine Learning for Audio Signal Processing Workshop, Neural Information Processing Systems (NIPS).

LIANG F. 2016. BACHBOT: automatic composition in the style of Bach chorales–developing, analyzing, and evaluating a deep LSTM model for musical style [doctorate dissertation, master's thesis]. University of Cambridge, Cambridge, UK.

LI G, DING S, LI Y. 2021. Novel LSTM-GAN Based Music Generation. In: 2021 13th International Conference on Wireless Communications and Signal Processing (WCSP). p. 1–6.

LIU H, XIE X, RUZI R, WANG L, YAN N. 2021. RE-RLTuner: a topic-based music generation method. In: 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR). p. 1139–1142.

LYU Q, WU Z, ZHU J, MENG H. 2015. Modelling high-dimensional sequences with lstm-rtrbm: application to polyphonic music generation. In: Twenty-Fourth International Joint Conference on Artificial Intelligence.

MAKRIS D, KALIAKATSOS-PAPAKOSTAS M, KARYDIS I, KERMANIDIS KL. 2017. Combining LSTM and feed forward neural networks for conditional rhythm composition. In: International conference on engineering applications of neural networks. Cham: Springer. p. 570–582.

MAO HH, SHIN T, COTTRELL G. 2018. DeepJ: style-specific music generation. In: Semantic Computing (ICSC), 2018 IEEE 12th International Conference. p. 377–382.

MAJIDI M, TOROGHI RM. 2022. A combination of multi-objective genetic algorithm and deep learning for music harmony generation. Multimedia Tools and Applications. p. 1–17.

MCALLISTER T, GAMBÄCK B. 2022. Music Style Transfer Using Constant-Q Transform Spectrograms. In: International Conference on Computational Intelligence in Music, Sound, Art, and Design (Part of EvoStar). Cham: Springer. p. 195–211.

MINU RI, NAGARAJAN G, BORAH S, MISHRA D. 2022. LSTM-RNN-based Automatic Music Generation Algorithm. In: Intelligent and Cloud Computing. Singapore: Springer. p. 327–339.

NIERHAUS G. 2009. Algorithmic composition: paradigms of automated music generation. Springer Science & Business Media.

PACHET F, SUZDA J, MARTINEZ D. 2013. A Comprehensive Online Database of Machine-readable Lead-sheets for Jazz Standards. In: ISMIR. p. 275–280.

PATI A, LERCH A, HADJERES G. 2019. Learning to Traverse Latent Spaces for Musical Score Inpainting. In: 20th International Society for Music Information Retrieval Conference (ISMIR).

RADFORD A, METZ L, CHINTALA S. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:1511.06434.

REN Y, HE J, TAN X, QIN T, ZHAO Z, LIU TY. 2020. Popmag: pop music accompaniment generation. In: Proceedings of the 28th ACM International Conference on Multimedia. p. 1198–1206.

ROBERTS A, ENGEL J, RAFFEL C, HAWTHORNE C, ECK D. 2018. A hierarchical latent vector model for learning long-term structure in music. In: International conference on machine learning. p. 4364–4373.

SABITHA R, MAJJI S, KATHIRAVAN M, KUMAR SG, KHARADE KG, KARANAM SR. 2021. Artificial Intelligence Based Music Composition System–Multi Algorithmic Music Arranger (MAGMA). In: 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC). p. 1808–1813.

SAXENA D, CAO J. 2021. Generative adversarial networks (GANs) challenges, solutions, and future directions. ACM Computing Surveys (CSUR) 54(3): 1–42.

URBANO J, LLORÉNS J, MORATO J, SÁNCHEZ-CUADRADO S. 2010. Melodic similarity through shape similarity. In: International Symposium on Computer Music Modeling and Retrieval. Berlin, Heidelberg: Springer. p. 338–355.

WALTER S, MOUGEOT G, SUN Y, JIANG L, CHAO KM, CAI H. 2021. MidiPGAN: a Progressive GAN Approach to MIDI Generation. In: 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD). p. 1166–1171.

WANG K, GOU C, DUAN Y, LIN Y, ZHENG, X, WANG FY. 2017. Generative adversarial networks: introduction and outlook. IEEE/CAA Journal of Automatica Sinica 4(4): 588–598.

YAMSHCHIKOV IP, TIKHONOV A. 2020. Music generation with variational recurrent autoencoder supported by history. SN Applied Sciences 2(12): 1–7.

YANG J. 2021. Music generation with long short-term memory network. In: Second IYSF Academic Symposium on Artificial Intelligence and Computer Engineering, Vol. 12079. p. 111–115.

YANG LC, CHOU SY, YANG YH. 2017. MidiNet: a Convolutional Generative Adversarial Network for Symbolic-domain Music Generation. In: 18th International Society for Music Information Retrieval Conference, Suzhou, China.

YANG LC, LERCH A. 2020. On the evaluation of generative models in music. Neural Computing and Applications 32(9): 4773–4784.

YU Y, SRIVASTAVA A, CANALES S. 2021. Conditional lstm-gan for melody generation from lyrics. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) 17(1): 1–20.