mathematics

MDPI

Article

Automatic Melody Composition Using Enhanced GAN

Shuyu Li, Sejun Jang and Yunsick Sung *

Department of Multimedia Engineering, Dongguk University-Seoul, Seoul 04620, Korea; lishuyu@dongguk.edu (S.L.); sejun@dongguk.edu (S.J.)

* Correspondence: sung@dongguk.edu

Received: 30 August 2019; Accepted: 19 September 2019; Published: 23 September 2019

Abstract: In traditional music composition, the composer has a special knowledge of music and combines emotion and creative experience to create music. As computer technology has evolved, various music-related technologies have been developed. To create new music, a considerable amount of time is required. Therefore, a system is required that can automatically compose music from input music. This study proposes a novel melody composition method that enhanced the original generative adversarial network (GAN) model based on individual bars. Two discriminators were used to form the enhanced GAN model: one was a long short-term memory (LSTM) model that was used to ensure correlation between the bars, and the other was a convolutional neural network (CNN) model that was used to ensure rationality of the bar structure. Experiments were conducted using bar encoding and the enhanced GAN model to compose a new melody and evaluate the quality of the composition melody. In the evaluation method, the TFIDF algorithm was also used to calculate the structural differences between four types of musical instrument digital interface (MIDI) file (i.e., randomly composed melody, melody composed by the original GAN, melody composed by the proposed method, and the real melody). Using the TFIDF algorithm, the structures of the melody composed were compared by the proposed method with the real melody and the structure of the traditional melody was compared with the structure of the real melody. The experimental results showed that the melody composed by the proposed method had more similarity with real melody structure with a difference of only 8% than that of the traditional melody structure.

Keywords: convolutional neural network; deep learning; generative adversarial network; long short-term memory; melody composition

1. Introduction

In traditional music composition, the composer has a special knowledge of music and combines emotion and creative experience to create music. As computer technology has evolved, various music-related technologies have been developed, but most of them have been focused on music editing such as arrangement and mixing. In these applications, music can be divided into several parts that are recombined based on special knowledge of musical composition to create new music. Experiments in musical intelligence (EMI) have used a recombinant algorithm to analyze music that had already been created and changed the style to generate new music [1].

To create new music, a considerable amount of time is required. Initially, people studied the possibility of mathematical models in music composition. The ILLIAC I computer composed new music known as the Illiac Suite based on the Markov algorithm [2]. With the remarkable achievements of artificial intelligence in various domains, people have begun to use artificial intelligence neural networks to generate higher quality music. Recurrent neural network-based music composition systems are being developed to help composers to quickly compose music. These systems include

Magenta [3], DeepJazz [4], BachBot [5], FlowMachines [6], and WaveNet [7], however, most of these music composition methods have a problem with composed music based on notes. However, as melody changes frequently within bars, the bar is used as the basic unit of composition.

Recently, deep learning has been used to compose music based on in-depth musical features. For example, CONCERT generates melodies using a recurrent neural network (RNN) [8] and DeepBach composes melodies and harmonies using a long short-term memory (LSTM) neural network [9], which can identify the structure of melodies more accurately than the RNN. The Song from the PI model composed melodies, harmonies, and percussion simultaneously using a multi-layer LSTM [10]. A generative adversarial network (GAN) is a deep-running algorithm that exhibits excellent performance in image processing and has been applied to several GAN-based music composition research. Continuous recurrent neural networks based on the GAN (C-RNN-GAN) compose music by building a GAN model with two LSTMs, then composing the melodies [11]. As the C-RNN-GAN composes music based on notes, it is limited by the poor quality of the composed music. Therefore, this study proposes a method for composing melody based on an enhanced GAN model. By proposing a GAN model suitable for bar-based encoding, the problems of note-based composition experienced by C-RNN-GAN were solved. Moreover, in our previous research, the differences were compared between the high frequencies of the melody and the non-melody by using the term frequency–inverse document frequency (TFIDF) algorithm and extracted the melody in the music as the training data by using a filter based on the shallow structure description [12].

Our contributions are as follows:

- This research utilizes bars to compose melody instead of notes so that the system can more accurately learn the complex and varied melody features of music.
- To ensure the rationality of the internal construction of the high dimensional matrix that represented the features of the bar, a CNN-based discriminator is utilized.
- When evaluating the quality of the composed melody, pitch is selected as a feature; by using the TFIDF algorithm to compare the composed melody with the real melody, the melody composed with the proposed method was proven to be more similar to the real melody.

The rest of this paper is organized as follows. Section 2 introduces previous research on deep learning-based melody composition. Section 3 describes the method for composing melody based on the enhanced GAN. Section 4 describes the experimental methods and results, and Section 5 analyzes the experimental results. Finally, Section 6 concludes the study.

2. Related Literature

2.1. Deep-Learning-Based Music Composition Methods

CONCERT [8] was the first system to generate melodies based on a RNN. To improve the quality of the generated melodies, various unique musical knowledge is encoded in the form of features. The input and output are notes that comprise three features: pitch, duration, and harmonic chord accompaniment. The representation of pitch is inspired by the circle of chromas and circle of fifths. The duration is expressed in cycles of 1/3 or 1/4 beats. The chord representation is based not only on a set of pitches that comprise the chord, but also on the four harmonics of each pitch. When generating the music, the output is interpreted as a probability distribution over a set of possible notes. After the data are output, the input data for the next generation of the output note are re-entered into the neural network as input data. However, when RNNs are used, gradient vanishing and gradient exploding problems occur, yielding a final result that is acceptable, but not as complex as real music.

MiniBach [13] can generate accompaniments to counterpoint a given melody of Bach. The accompaniment generation system is based on a feed forward neural network that consists of input, hidden, and output layers. For the input and output data, a one-hot encoding-based piano roll is used. The rectified linear unit (ReLU) nonlinear function is used as the hidden layer's

activation function and the output layer uses the Softmax function. However, MiniBach has difficulties accurately predicting notes when only a feed forward neural network is used. To resolve this problem, the MiniBach-based DeepBach [9] system was created. DeepBach is a composite structure that combines two LSTM neural networks and two feed forward neural networks. Unlike traditional LSTM neural networks, the DeepBack structure considers the piano's roll forward and backward directions based on the input of the one-hot encoding scheme. To predict the piano roll, the forward direction LSTM neural network analyzes the forward direction piano roll data, while the reverse direction LSTM neural network analyzes reverse direction data. Moreover, the feed forward neural network generates a piano roll. The two LSTM neural networks and the output of the feed forward neural network are combined and processed by the nonlinear ReLU function, and the notes are finally output by the Softmax function. The results of an online survey of 1200 individual auditors from experts to novices in this study indicate that it is difficult to distinguish whether a melody was made by Bach or generated by DeepBach. However, this is limited to the generation of accompaniments for Bach's melodies.

In contrast, the Song from the PI model [10] uses a multilayer LSTM neural network structure to generate polyphonic music that consists of melody, percussion, and harmony. This model separates melody, chords, and drums to construct a network. This multi-layer recurrent neural network is used to generate pop music. Additionally, this network used some aspects of music theory such as the twelve-tone, the triad, and the circle of fifths. For each time step, the melody will be encoded into random vectors, the first representing which key was pressed and the second representing the length of time the key was pressed. Then, chords and beats are generated in each time step according to the melody generated by the networks. Finally, the output of all layers constitutes the entire song. The objective of most music generation methods including Song from PI, is to generate melody based on notes. However, as the use of notes is limited to local features, it is unlikely to be suitable for composing music. Therefore, musical features must be addressed more holistically.

A C-RNN–GAN [11] is a generative adversarial network (GAN) model that is configured based on an LSTM neural network for generating music. MidiNet [14] and MuseGAN [15] are examples of research on using GAN models to generate music. MidiNet is a conditional GAN model configured by a convolutional neural network (CNN) that generates a bar-based melody based on a given chord. The conditional GAN model can generate music that satisfies a variety of conditions. MuseGAN is similar in structure to MidiNet, as both use GAN and CNN to generate music. However, to compensate for the lack of continuity caused by using CNNs to generate bars, the generator section uses two subnetworks: a bar generator and a temporal structure generator. The time continuity of the bar sequence generated by the bar generator is handled by the temporal structure generator. The JazzGAN [16] was proposed to create a jazz teaching tool. JazzGAN improved the monophonic jazz melodies and to solve the several not addressed, proposed the use of harmonic bricks for phrase segmentation.

2.2. Comparison of Deep Learning-Based Music Generation Methods

Table 1 shows the difference between existing music generation systems and the system proposed in this study. The proposed method has the following advantages compared to existing music generation systems. Due to the limitations of RNN, gradient vanishing and gradient exploding problems will occur in CONCERT [8]. Conversely, DeepBach [9] and the Song from PI [10] use a LSTM neural network that has overcome the problems of gradient vanishing and gradient exploding, which constitute the primary disadvantages of the RNN; thus, the structure of the melody can be understood more precisely. However, as melodies are generated based on musical notes, the melody's musical meaning and quality are degraded. To overcome these disadvantages, this study proposes a matrix-based bar encoding method that generates melodies based on bars. C-RNN–GAN [11] is a GAN model that is configured as a LSTM neural network to generate note-based melodies. This study proposes an enhanced GAN model that employs two discriminators to generate bar-based melodies. The enhanced GAN model discriminators use a LSTM neural network and a CNN neural network to ensure correlation between the bars and rationality of the note arrangement within bars. MidiNet [13]

generates bar-based melodies, but cannot distinguish between long notes and continuous tones with the same pitch when encoding is performed; therefore, this study uses a matrix-based bar encoding method to resolve this problem. Given that multiple notes appearing simultaneously are encoded together, polyphonic music can be represented using the proposed method.

Table 1. Differences between existing models and the proposed method.

	CONCERT	Deep Bach	Song from PI	C-RNN-GAN	MidiNet	Muse GAN	Proposed Method
Model	-	-	-	GAN	Conditional GAN	GAN	GAN
Neural Network	RNN	LSTM	LSTM	LTSM; Bi-LSTM	CNN	CNN	LSTM; Bi-LSTM; CNN

3. Enhanced GAN-Based Melody Composition System

The proposed method is an enhanced GAN model that composes melody based on bars. Figure 1 depicts the entire process of the proposed melody composition system. First, the training data required for the model is generated in the pretreatment phase. The difference between the high frequencies of the melody and the non-melody is compared by using the TFIDF algorithm, and the melody in the music is extracted as the training data by using a filter based on the shallow structure description. The extracted melody is then divided into several bars by using the tempo information stored in the MIDI file. Then, all the notes in the bar, which contain four features of pitch, start time, duration, intensity, will be encoded into a high latitude matrix. Finally, all coded bar matrices are normalized to the same dimension size. Second, in the MIDI file generate step, new melody matrices are generated while training with the enhanced GAN model using the preprocessed training data. New melody matrices can be constructed into new MIDI files through transformation; melody is then created using the new MIDI files. The enhanced GAN model consists of one generator and two discriminators. The two discriminators include the RNN based discriminator and CNN based discriminator. RNN-based discriminators can consider the characteristics of melody, which is time-series data, and CNN-based discriminators consider the whole structure of the melody. The generator generates new melody matrices based on noise vectors and sends them to the two discriminators. The two discriminators determine whether the input melody matrices are the melody matrices generated from the generator or those extracted from MIDI files. The discriminators and the generator are trained using the results of the discriminators. Then, the two discriminators are used to train the generator of the proposed GAN model. The structure of the enhanced GAN model is shown in Figure 2.

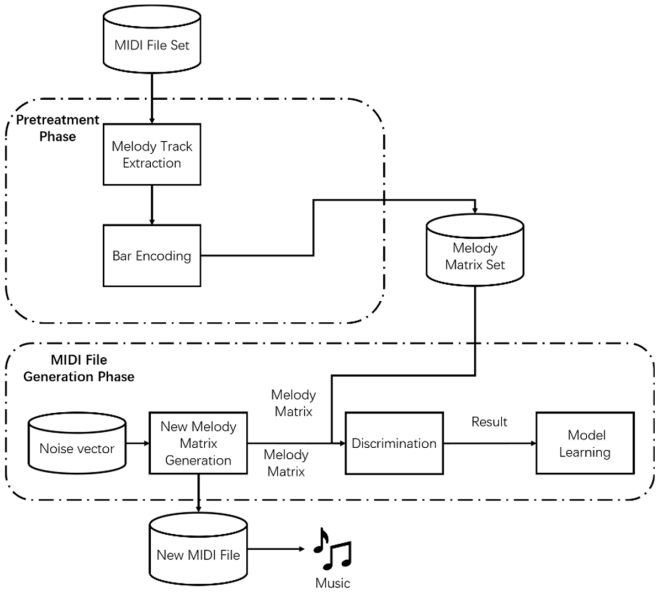

Figure 1. Process of the proposed melody composition system.

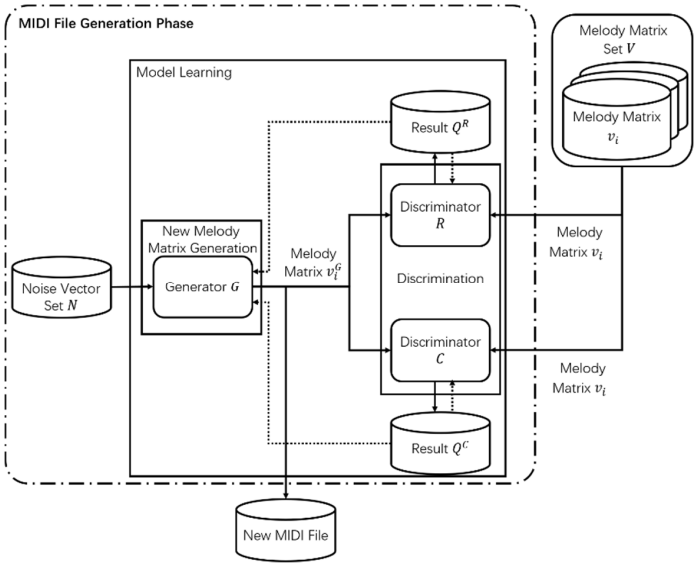

Figure 2. Structure of the enhanced Generative Adversarial Network model.

3.1. The Generator

The generator, G, consists of an LSTM layer and a fully connected (FC) layer, as shown in Figure 3. The RNN network is used to process sequence data with time attributes. LSTM adds the input gate, forget gate, output gates, and states, which are absent in RNN. LSTM solves the problem of gradient disappearance and gradient explosion of RNN by using these new structures. The amount of information passing through each gate is controlled by the sigmoid function. The input gate extracts information from the input bar in each time step and adds it to the state by using an add operation. However, the forget gate will make states selectively forget some of the information based on the input bar. These two gates will constantly add and forget information to keep updating the state. Eventually, the updated state in each time step will predict the high input bar. The predicted bar will be used as the input for the next time step after sampling. By repeating the above process, a sequence of bars with time attributes can be generated. These sequences are melodies. The data processing procedure is as follows:

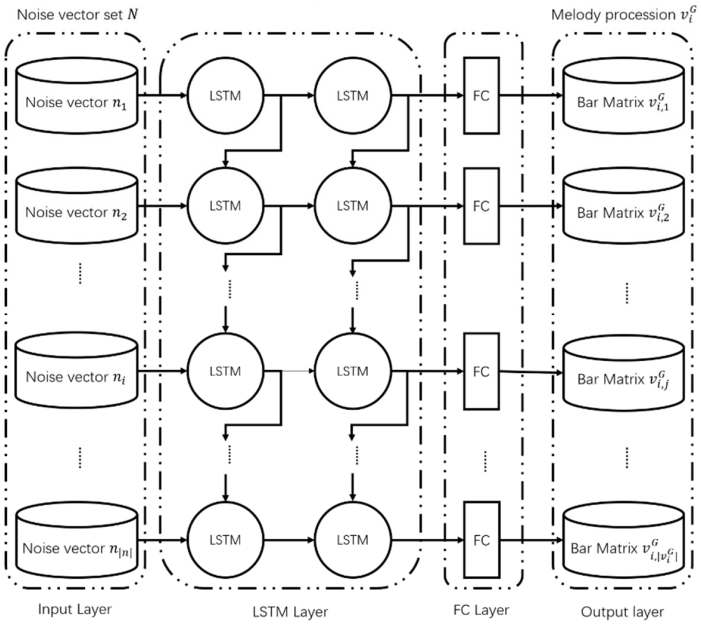

Figure 3. Structure of the generator (G).

The input to G is defined as a set of noise vectors, N, which consists of an equal number of noise vectors to the number of bars to be generated. The i^{th} melody matrix generated through the generator is defined as v_i^G. v_i^G consists of an equal number of bar matrices, $v_{i,j}^G$, to the number of bars to be generated. The LSTM layer consists of two hidden layers of the same size; the second hidden layer is connected to the FC layer. The FC layer receives each noise vector, n_i, that passes through the LSTM layer and outputs a j^{th} bar matrix, $v_{i,j}^G$. It then generates the i^{th} melody matrix, v_i^G, by integrating the generated bar matrices, $v_{i,j}^G$.

3.2. The Two Discriminators

The bidirectional-LSTM (Bi-LSTM) model is used in discriminator R. As shown in Figure 4, the Bi-LSTM layer consists of a forward LSTM layer and a backward LSTM layer. Unlike LSTM, Bi-LSTM can analyze melody simultaneously from both directions, thus ensuring the accuracy of the judgment. The inputs to R are the melody matrix, v_i^G, generated by the generator, G, and the melody matrix, v_i, extracted from a set of melody matrices, V. The forward LSTM layer receives the bar matrices, $v_{i,j}^G$ and $v_{i,j}$, in the forward direction and the backward LSTM layer receives the bar matrices, $v_{i,j}^G$ and $v_{i,j}$, in the backward direction. The two LSTM layers are connected to the FC layer and the final determination result, Q^R, is subsequently derived. The enhanced GAN model proposed in this study, unlike the traditional GAN model, uses two discriminators by adding discriminator C to ensure the rationality of the bars. Discriminator C derives the determination result using the CNN model shown in Figure 5 and C consists of two hidden CNN layers: a pooling layer and an FC layer. CNN demonstrates excellent performance when dealing with tasks such as object recognition and image classification. The purposed method is to compose melody based on the bar, rather than based on the note, so each bar is encoded as a high-dimensional matrix, which can be considered a picture. Through the convolution operation, the features of each bar are deeply extracted, and the internal structure of the generated bar analyzed. As a basic unit of melody composition, the rationality of the internal structure of the bar is very important. In the CNN-based discriminator C, the first hidden CNN layer receives the divided bar matrices, $v_{i,j}^G$ and $v_{i,j}$, from the melody matrices, v_i^G and v_i, then extracts features through a convolution process. The extracted feature distribution is inputted into the second CNN hidden layer in order to extract more in-depth features, which are then inputted to the pooling layer. The features extracted by the pooling layer are inputted into the FC layer and the determination result is outputted.

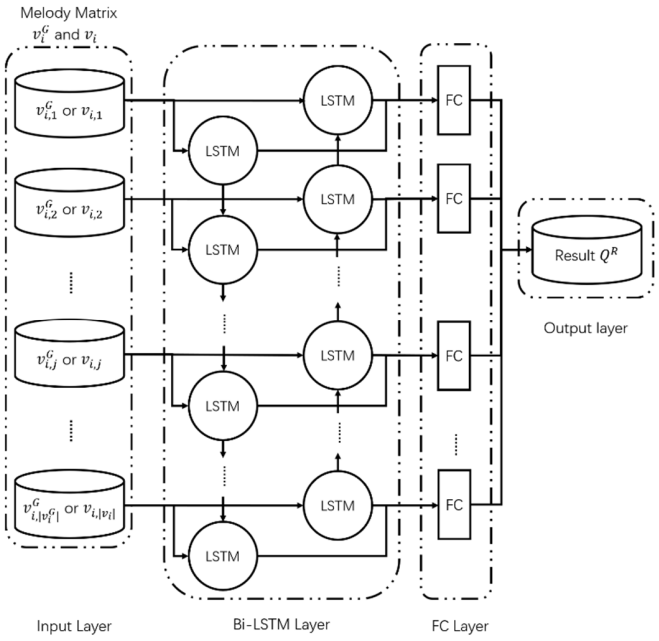

Figure 4. Structure of discriminator (R).

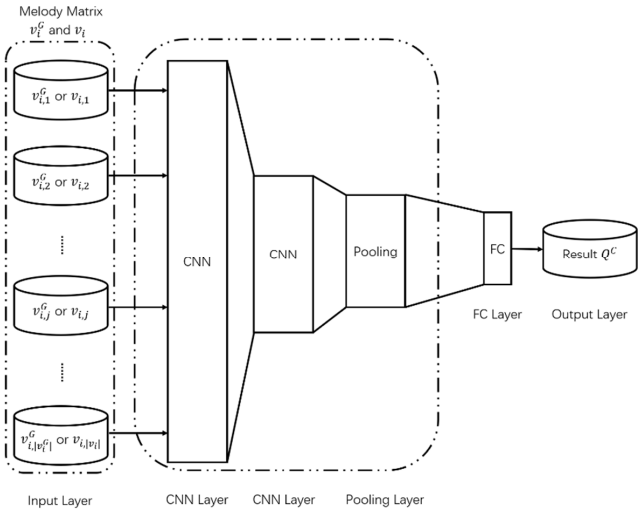

Figure 5. Structure of discriminator (C).

The discriminators R and C simultaneously process the same melody matrices and output the judgment result Q^R and Q^C by the sigmoid function. The determination result has a value between 0 and 1. If values inputted to the discriminators are melody matrices of the encoded melody tracks, the discriminators are trained to derive 1. Conversely, for melody matrices from the generator, the generator is trained to determine 0.

3.3. Model Training

Therefore, G is the generator, R and C are the discriminators, N is a set of noise vectors, V is a set of melody matrices encoded with a set of melody tracks, and v_i is a melody matrix extracted from V. The discriminators C and R derive the determination results according to the correlation between each bar and the rationality of the bars, respectively. Based on each determination result, they are trained by maximizing the loss, as shown in Equations (1) and (2) using $Loss_R()$ and $Loss_C$ (), respectively.

$$Loss_R(R, G) = E_{v_i}[logR(v_i)] + E_N[1 - logR(G(N))] \tag{1}$$

$$Loss_C(C, G) = E_{v_i}[logC(v_i)] + E_N[1 - logC(G(N))] \tag{2}$$

The generator is trained as in Algorithm 1 by optimizing the loss using $Loss_G()$ (Equation (3)) by comprehensively considering the determination results of the two discriminators.

$$Loss_G(R, C, G) = \frac{E_N[1 - \log(G(N))] + E_N[1 - logC(G(N))]}{2} \tag{3}$$

Algorithm 1. Enhanced GAN model training algorithm.

```
FUNCTION BackPropagation(N, V)
BEGIN
  Initialize generator G, discriminator R, discriminator C
  Initialize N
  FOR i←1 to SIZE(number of iterations)
    FOR j←1 to SIZE(number of batchsize)
      v_i from V
      update R by Loss_R(R, G)
      update C by Loss_C(C, G)
      update G by Loss_G(R, C, G)
    END FOR
  END FOR
END
```

4. Experiments and Results

4.1. Experimental Objectives

Experiments were conducted to verify the quality of the MIDI files generated using the proposed method.

4.2. Experimental Environment

The following quality measures were compared for MIDI files generated by the proposed bar-based GAN model and the note-based C-RNN-GAN model. First, the loss variation was compared. The loss was calculated based on the discrimination results of the discriminators, where the calculated loss was inversely proportional to the performance of the generator and the discriminators. Second, we compared the structure of MIDI files generated by the proposed method and the traditional method with the structure of a set of MIDI files for the melody composed by a real composer. In the comparison, the structural difference, D, was determined by utilizing the TFIDF algorithm (Equation (4)) based on the number of notes, s_i^G, contained in the MIDI file, M_i^G, among a set of MIDI files, M^G, the number of melody tracks, β, contained in a set of actual melody tracks, T, and the number of files, n_p^T, where a pitch, p, is present. When p is not present in the MIDI files, M^G, the calculation is not possible because $n_p^{M^G}$ becomes 0. Therefore, $n_p^{M^G}$ is not 0 and μ is the number of the pitch, p.

$$\mathrm{D} = \frac{1}{\mu} \times \sum_{\mathrm{p}=1}^{128} \frac{n_p^{M^G}}{s_i^G} \times \log\left(\frac{\beta}{n_p^T}\right) \tag{4}$$

The experimental parameters are shown in Table 2, where keep_prob is a dropout value, *batch_size* is a melody track inputted into the model at one time, *pretraining_epochs* is the number of pre-training epochs of the model, *learning_rate* is the amount of learning, and *songlength* is the length of the number of composed units. The basic unit for composing a melody in C-RNN-GAN is a note. The basic unit in the proposed method is the bar, so in the case of the same parameters, the melody composed by the proposed method will be longer. Therefore, in order to make the melody composed by the two methods in the same length, after repeated trials and comparisons, the parameters of C-RNN-GAN are finally adjusted to four times that of the proposed method. Furthermore, hidden_size_g is the size of the hidden layer of the proposed bar-based GAN model and the traditional note-based C-RNN-GAN model, hidden_size_d is the size of the hidden layer in traditional models, hidden_size_d_rnn is the size of the hidden layer of the proposed Bi-LSTM-based discriminator, and kernal_size_d_cnn is the size of the CNN-based discriminator filter added to the proposed bar-based GAN model.

Table 2. Values of experimental parameters used in the proposed bar-based GAN model and the note-based C-RNN-GAN model.

Parameter Name	Proposed	C-RNN-GAN
keep_prob	0.5	0.5
batch_size	20	20
pretraining_epochs	5	5
learning_rate	0.1	0.1
songlength	4	16
hidden_size_g	100	100
hidden_size_d	-	100
hidden_size_d_rnn	100	-
kernal_size_d_cnn	5	-

The experimental environment is composed of Windows 10, i5-6400, NVIDIA GeForce GTX 1050 2 GB, DDR4 8 GB. The proposed system was developed in Python and the GAN model was implemented using the deep learning library TensorFlow. MIDI files were processed by the python-midi library.

4.3. Experimental Data

For the quality verification of melody, the dataset used in experiments was the MIDI files that was configured with Korea pop songs and 2000 MIDI files. A set of melody tracks were derived from the dataset using a melody track method as input data for both the proposed model and the traditional model. In the experiments, it was encoded using a set of melody tracks extracted from MIDI files based on the individual bars. A set of extracted melody tracks was made up of 2000 melody tracks; each bar in a melody track was encoded into a matrix with a fixed size. The 2000 melody matrices encoded based on the individual bars were used as inputs to the proposed enhanced GAN model.

4.4. Experimental Results

This section describes the quality of MIDI files composed by the proposed melody composition system based on the enhanced GAN model. The loss variation of the proposed model is shown in Figure 6. Due to the high loss from Epoch 1 to 5, the model was trained in advance by pre-learning to reduce the loss below a certain level. The loss of the generator of the proposed model, indicated by the red solid line in Figure 6, started at 2663 at Epoch 6, reached 264.245 at Epoch 70, and ended at 2.699 at Epoch 150. The loss of the RNN-based discriminator of the proposed model is indicated by the blue solid line. The loss of the RNN-based discriminator started at 2659 at Epoch 6, reached 263.802 at Epoch 70, and training ended at 1.326 at the final Epoch. The loss of the CNN-based discriminator of the proposed model, indicated by the green solid line, started at 2670 at Epoch 6, reached 264.392 at Epoch 70, and training ended at 1.813 at the final Epoch. The loss of the proposed model decreased evenly, which will be discussed in Section 5. Figure 7 shows the loss variation of the traditional model, which was also pre-trained from Epoch 1 to 5 due to the high loss. Similar to the proposed model, the traditional model was limited to 150 Epochs. The loss of the generator and discriminator of the proposed model (red and green lines, respectively) started from 96.948 and 98.119 at Epoch 6, then reached 12.246 and 4.456 at Epoch 70, and training ended at 15.714 and 4.617 at the final Epoch, respectively.

During training of the GAN models, the loss change graphs of the two models were outputted. However, as the bar matrix contains a larger number of features than a single note, the loss of the proposed method at the beginning of training was significantly higher than the loss value of C-RNN-GAN. At the end of the training, it is worth noting that the loss value of the proposed method was lower than the loss value of C-RNN-GAN. This shows that the two discriminators can help the generator improve better than a single discriminator. Table 3 shows the difference in the number of Epochs, training time, and loss between the proposed model and the traditional model. The proposed and traditional methods are based on GAN models, so it is possible to compare the methods by loss values. The loss of the generator in the proposed model was 2.699, which was approximately 82.8% lower than the loss of the generator in the traditional model (15.714). The loss of the RNN-based discriminator in the proposed model was 1.570, which was approximately 66.0% lower than the loss of the discriminator in the traditional model (4.617).

Table 3. Results of the proposed model and C-RNN-GAN model.

	Proposed Model	**C-RNN-GAN**
Epoch	150	150
Training Time	45 min	270 min
Loss of Generator	2.699	15.714
Loss of Discriminator	1.570	4.617

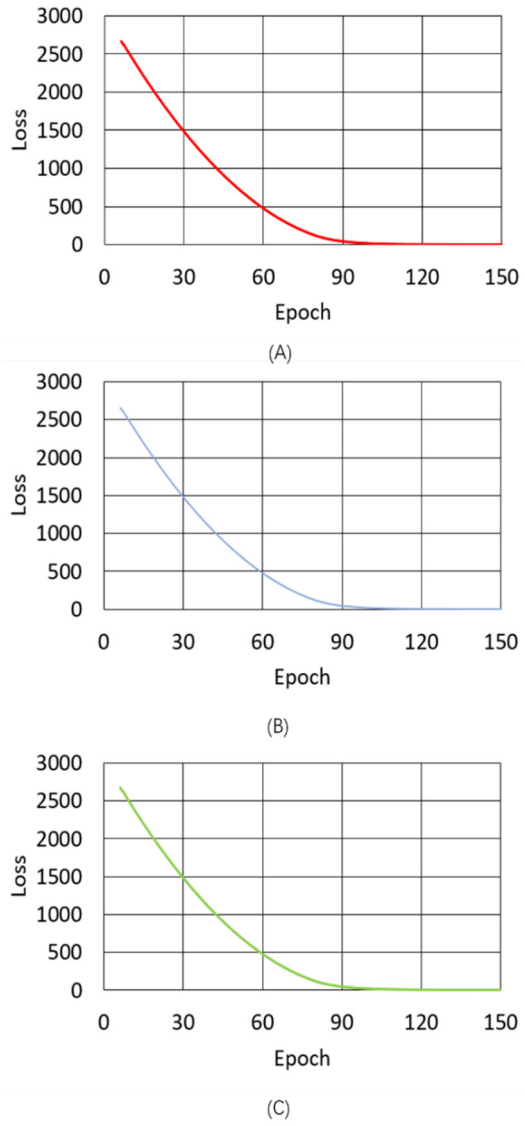

Figure 6. Loss variation of the proposed model for: (**A**) the generator-RNN; (**B**) the discriminator-RNN; and (**C**) the discriminator-CNN.

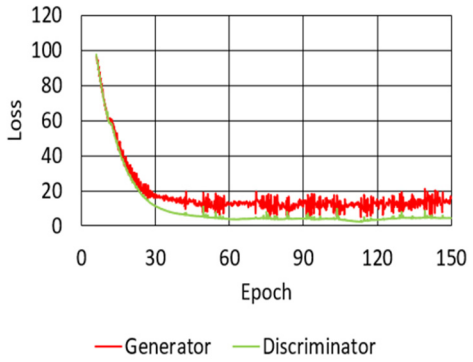

Figure 7. Loss variation of the C-RNN-GAN model for: the generator and the discriminator.

Figure 8 shows the TFIDF values derived from the TFIDF [17] algorithm in order to compare the structure of the MIDI files generated randomly by the proposed system and generated by the traditional system with the structure of MIDI files generated by a real composer. To verify the effectiveness of the training, 20 MIDI files were utilized that were generated by the model without training, 20 files

generated by the traditional system, 20 files generated by the proposed system, and 20 files composed by a real composer. The TFIDF values of the MIDI files generated by the proposed method (in blue) were similar to those of the actual MIDI files (in yellow). Orange indicates the TFIDF values of the MIDI files composed by the C-RNN-GAN model and gray indicates the TFIDF values of the randomly generated MIDI files, which are different from the TFIDF values of the actual MIDI files. Compared to the MIDI files generated by C-RNN-GAN, the MIDI files generated by the proposal method were more similar to the real melody in the pitch structure. After calculating the average value, this gap was 8%. Therefore, this verifies the effectiveness of the proposed method training.

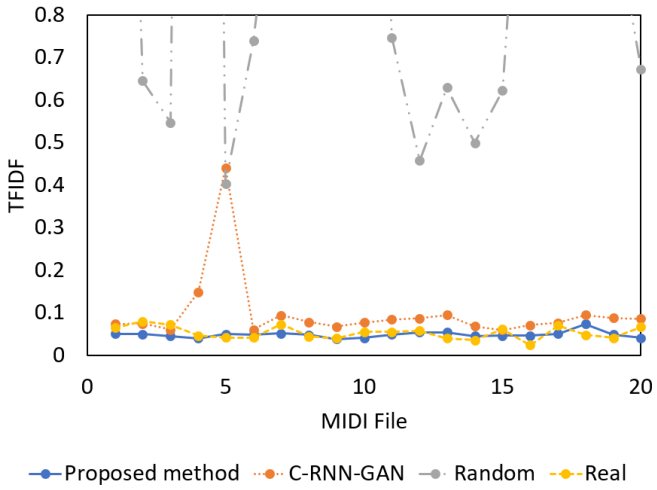

Figure 8. MIDI file quality verification results.

5. Discussion

5.1. Loss Analysis of the Enhanced GAN Model

The loss of the proposed model presented in Section 4 appeared to decrease evenly when the loss of all iterations is shown because of the substantial difference in loss between the start and end of training. Figure 9 shows the loss of each iteration from Epoch 85–125 for the proposed model, which indicates an uneven decrease in the loss.

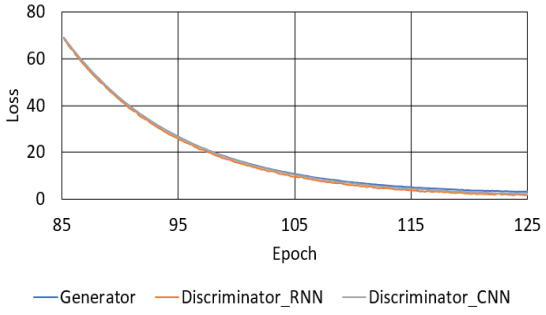

Figure 9. Variation of loss in the proposed model between Epoch 85 and 125.

5.2. Analysis of TFIDF-Based MIDI File Quality Verification

According to the verification results of melody quality, the average TFIDF value of the melody tracks generated by the proposed system and a set of real melody tracks was 0.049 and 0.053 (Table 4). When the TFIDF calculated their similarity with real melody, the proposed method obtained a lower score. Compared to the real melody, the composed fake melody was more similar to the real melody. By observing the TFIDF value, it was found that the TFIDF values of a set of real melody had a wider range of values and were evenly distributed in this range. However, the melody composed by the

proposed method mostly had the same TFIDF value, with most of it concentrated between 0.04 and 0.05. This demonstrates the great diversity of the real melody, and the proposed method will carefully select the most probable notes in the alternative notes to form the bar.

Table 4. TFIDF values of the MIDI file quality verification results.

Index	Random	C-RNN-GAN	Proposed System	Real
Average	1.569	0.099	0.049	0.053
STD	1.500	0.083	0.007	0.015

6. Conclusions

This study introduced a system that composed melody based on an enhanced GAN model. In the enhanced GAN model, two discriminators based on RNN and CNN were utilized to ensure the correlation between bars and the rationality of the node structure. In order to verify the proposed method, experiments on the quality of the MIDI files were conducted. In the experiment, a new melody was composed using a matrix-based bar encoding method and an enhanced GAN model, and evaluated the quality of the composed melody. The structure of the randomly composed melody was compared, the melody composed by the C-RNN-GAN model, and the melody composed by the proposed method with the structure of actual melody, then determined the differences. The structure of the melody composed by the proposed method was most similar to the structure of the actual melody, with a difference of only 8%. Therefore, the melody composed by the proposed method were closest to the actual melody.

Author Contributions: Conceptualization, S.L., S.J., and Y.S.; Methodology, S.L., J.S., and Y.S.; Software, S.L., S.J., and Y.S.; Validation, S.L., S.J., and Y.S.

Acknowledgments: This work was supported by the Dongguk University Research Fund of 2017.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cope, D. Experiments in musical intelligence (EMI): Non-Linear linguistic-based composition. *J. New Music Res.* **1989**, *18*, 117–139. [CrossRef]
2. Sandred, O.; Laurson, M.; Kuuskankare, M. Revisiting the Illiac Suite—A rule-Based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas* **2009**, *2*, 42–46.
3. Magenta. Available online: https://magenta.tensorflow.org (accessed on 22 July 2019).
4. DeepJazz. Available online: https://deepjazz.io (accessed on 22 July 2019).
5. BachBot. Available online: https://bachbot.com (accessed on 22 July 2019).
6. FlowMachines. Available online: https://www.flow-machines.com (accessed on 22 July 2019).
7. WaveNet. Available online: https://deepmind.com (accessed on 22 July 2019).
8. Mozer, M.C. Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-Scale processing. *Connect. Sci.* **1994**, *6*, 247–280. [CrossRef]
9. Hadjeres, G.; Pachet, F.; Nielsen, F. Deepbach: A steerable model for bach chorales generation. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1362–1371.
10. Chu, H.; Urtasun, R.; Fidler, S. Song from PI: A musically plausible network for pop music generation. *ICLR* **2017**. under review.
11. Mogren, O. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *Constructive Mach. Learn. Workshop* **2016**. accepted.
12. Pedro, J.; De León, P.; Pérez-Sancho, C.; Inesta, J.M. A shallow description framework for musical style recognition. In Proceedings of the Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Lisbon, Portugal, 18–20 August 2004; pp. 876–884.

13. Briot, J.P.; Hadjeres, G.; Pachet, F. Deep learning techniques for music generation-a survey. *arXiv* **2017**, arXiv:1709.01620.
14. Yang, L.C.; Chou, S.Y.; Yang, Y.H. MidiNet: A convolutional generative adversarial network for symbolic-Domain music generation. In Proceedings of the 2017 International Society of Music Information Retrieval Conference (ISMIR), Suzhou, China, 24–27 October 2017.
15. Dong, H.W.; Hsiao, W.Y.; Yang, L.C.; Yang, Y.H. MuseGAN: Multi-Track sequential generative adversarial networks for symbolic music generation and accompaniment. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 34–41.
16. Trieu, N.; Keller, R.M. JazzGAN: Improvising with Generative Adversarial Networks. In Proceedings of the 6th International Workshop on Musical Metacreation (MUME), Salamanca, Spain, 25–26 June 2018.
17. Trstenjak, B.; Mikac, S.; Donko, D. KNN with TF-IDF based framework for text categorization. *Procedia Eng.* **2014**, *69*, 1356–1364. [CrossRef]