Knowledge Engineering and Data Science (KEDS)
Vol 7, No 1, April 2024, pp. 101–116

pISSN 2597-4602
eISSN 2597-4637

Timbre Style Transfer for Musical Instruments Acoustic Guitar and Piano using the Generator-Discriminator Model

Widean Nagari [1], Joan Santoso [2,*], Esther Irawati Setiawan [3]

Institut Sains dan Teknologi Terpadu Surabaya
Jl. Ngagel Jaya Tengah No. 73-77, Surabaya 60284, Indonesia
[1]widean.n23@mhs.istts.ac.id; [2]joan@stts.edu; [3]esther@stts.edu*
**corresponding author*

ARTICLE INFO

Article history:
Received 27 June 2024
Revised 15 July 2024
Accepted 23 August 2024
Published online 05 September 2024

Keywords:
Discriminator
Generator
Music
Style Transfer
Timbre

ABSTRACT

Music style transfer is a technique for creating new music by combining the input song's content and the target song's style to have a sound that humans can enjoy. This research is related to timbre style transfer, a branch of music style transfer that focuses on using the generator-discriminator model. This exciting method has been used in various studies in the music style transfer domain to train a machine learning model to change the sound of instruments in a song with the sound of instruments from other songs. This work focuses on finding the best layer configuration in the generator-discriminator model for the timbre style transfer task. The dataset used for this research is the MAESTRO dataset. The metrics used in the testing phase are Contrastive Loss, Mean Squared Error, and Perceptual Evaluation of Speech Quality. Based on the results of the trials, it was concluded that the best model in this research was the model trained using column vectors from the mel-spectrogram. Some hyperparameters suitable in the training process are a learning rate 0.0005, batch size greater than or equal to 64, and dropout with a value of 0.1. The results of the ablation study show that the best layer configuration consists of 2 Bi-LSTM layers, 1 Attention layer, and 2 Dense layers.

I. Introduction

In an information-driven world, a company's technological infrastructure is the core of its daily operations and a fundamental pillar for its growth and adaptation to a constantly evolving business environment. The Company, aware of this reality, is embarking on a strategic initiative to modernize its virtual server infrastructure by implementing Docker technology, a leading solution in virtualization at the operating system level. The company's current infrastructure is heavily reliant on VMware virtualization technology. While VMware has provided a robust solution for server virtualization, several challenges have emerged over time that are affecting the scalability, operating costs, and overall business agility. Here is an overview of these challenges: scalability limitations, increasing operating costs, and complexities hindering business agility. Given these challenges, the company recognizes the need to transition to a more flexible, scalable, and cost-effective solution. Docker-based server virtualization offers a promising alternative, providing lightweight containers that can run across various environments with minimal overhead. This transition aims to address the limitations of the current VMware infrastructure by offering improved scalability, reduced operating costs, and enhanced business agility through streamlined management and deployment processes.

Neural style transfer techniques were initially developed for image processing, where models transfer artistic style from one image to another. Many works have been produced in this domain, including ones related to timbre conversion. Timbre or tone colour is the quality of the sound made by an object, which allows listeners to distinguish whether two sounds have similar or different sound qualities and characteristics. Only musical instruments that can produce harmonious combinations have timbre, while objects such as metal rods that only make one sound do not have timbre characteristics.

Music Style Transfer is creating creative music that resembles human work by combining different songs' musical content and musical styles. The image style transfer concept explained previously

https://doi.org/10.17977/um018v7i12024p101–116

makes the Music Style Transfer problem interesting to solve. Music Style Transfer focuses on creating new music that combines input song content and target song style with human sound characteristics so that humans can enjoy it.

This research uses the Generator-Discriminator model to solve the timbre style transfer, a branch of music style transfer. Timbre style transfer is an important topic in music style transfer because it transforms sound characteristics from one instrument to another without changing the melody or harmony being played. This technology enriches music production by offering an unprecedented variety of sounds, allowing producers and musicians to create more innovative and unique works. In addition, timbre style transfer also drives the development of machine learning technology in sound analysis and synthesis. With the ability to retain a song's essence while changing the sound's characteristics, timbre style transfer provides an exciting new dimension in music exploration and creation. In this problem, we train a Generator-Discriminator model to change the sound of a musical instrument from one song by using the sound of a musical instrument from another.

Research on neural style transfer was pioneered by Gatys et al. [1], who introduced the concept of combining the style and content of two images using a pre-trained CNN model. The emergence of the idea of neural style transfer makes similar research in the sound domain, especially music, interesting. Several methods have been used, such as GAN-based models. Brunner et al. [2] utilized the CycleGAN model to transfer genres to songs by training using a dataset containing songs from the Jazz, Classic, and Pop genres in MIDI form. Research utilizing the RaGAN model was conducted by Lu et al. [3] by using unsupervised learning techniques. The dataset in research conducted by Lu et al. utilizes piano and guitar solo performance videos, which are preprocessed to become mel-spectrograms. Dong et al. [4] developed the MuseGAN model, a sequential GAN model trained to generate multi-track musical compositions, including melody, harmony, and rhythm. This model can produce coherent and diverse musical compositions. Yang et al. [5] introduced the MidiNet, a Convolutional-GAN-based model designed for creating music with MIDI files. MidiNet models are trained using large datasets containing MIDI files to capture complex patterns and structures in music and produce enjoyable music. Zaoxu Ding et al., in their research [6], propose a method called SteelyGAN, a music genre transfer model that uses GANs to transfer musical style at both pixels on piano rolls and latent levels.

Besides GANs, autoencoder models are also effective for performing timbre style transfer. Brunner et al. [7] introduced the MIDI-VAE, a Variational Autoencoder-based model capable of handling polyphonic music. Brunner et al. show that the MIDI-VAE model can transfer styles to symbolic music and automatically change the pitch, dynamics, and instruments from classical music to jazz music. Cifka et al. [8] developed a VQ-VAE model trained with self-supervised learning techniques to obtain separate representations of timbre and pitch. Research by Cifka et al. using Lakh MIDI Dataset and RealTrack datasets, which are preprocessed in such a way that they are in STFT form. Wu et al. [9] utilise a VAE model combined with a Transformer called MuseMorphose. Wu's research focuses on transferring style to produce pop piano music. Cifka et al. [10] presented a one-shot style transfer method for accompaniment styles in the symbolic music domain called Groove2Groove, based on AutoEncoder. Cifka's research focuses on the case of accompaniment styles in popular music and jazz. Hung et al. [11] used an autoencoder-based model that can take a piece of music and make it sound like it was played in a different style, focusing on changing the instrument's sound quality without changing the core tone. Hung's research used the MedleyDB+Mixing Secret and MuseScore datasets.

This research has several similarities and differences with the previous studies that have been described. This research and the research conducted by Lu et al. [3] utilized the mel-spectrogram data representation and the GAN model to perform music style transfer. The mel-spectrogram is a form of spectrogram that uses mel-scale to represent the shape of the waveform of an audio signal, which can capture the time and frequency information of the signal in a concise and informative manner, making it suitable for music-style transfer tasks. Besides mel-spectrograms, many other studies use MIDI data representations, such as Brunner et al. [2], Yang et al. [5], Cifka et al. [8], etc. Lu et al. use the RaGAN (Relativistic average GAN) model, which differs from traditional GANs by making the discriminator predict relative authenticity, i.e., whether a given real data example is more realistic than artificial data. This relativistic approach aims to stabilize the training and produce higher-quality output. In contrast, this study uses the standard GAN model, which focuses on generating timbre-transferred songs by optimizing the generator to produce mel-spectrograms that capture the target style and retain

the original content. In terms of loss function, this study uses BCE Loss as adversarial loss and Contrastive Loss as reconstruction loss in the training phase. Lu et al. [3] use Chi-Square Loss as adversarial loss and to count the reconstruction loss as in (1).

$$L_{rec} = L_{rec}^{x} + L_{rec}^{y} = |x - \hat{x}|_1 + |y - \hat{y}|_1 \quad (1)$$

Where $\hat{x}$ and $\hat{y}$Are the reconstructed features of x and $y$respectively. This study focuses on musical timbre style transfer, where the main goal is to combine the musical content and musical style of the sounds of different instruments in 2 songs with the same melody by utilizing the standard GAN model. This research focuses on two musical instruments: piano and guitar.

This paper presents a novel Generator-Discriminator model for timbre-style transfer, advancing the current state of research in this domain. The model is rigorously validated through comprehensive testing methodologies, including ablation studies, hyperparameter tuning, input dimension assessments, evaluations using slower-tempo songs, and performance analysis of the discriminator using a threshold-based approach. Additionally, contrastive learning techniques are employed, specifically utilizing the Contrastive Loss Function, to enhance the training efficacy of the generator model. This multi-faceted approach demonstrates the robustness of the model and establishes its effectiveness in achieving superior timbre style transfer outcomes, underscoring the innovative contributions of this research.

II. Methods

This section will discuss the system created, starting from the dataset model architecture and the input and output.

A. Dataset

The dataset used in this research is the MAESTRO (MIDI and Audio Edited for Synchronous Tracks and Organization) dataset. This dataset is a collection of audio created by PG Music and available for download from Magenta. MAESTRO consists of approximately 200 hours of piano performance aligned with notation labels and audio waves. MAESTRO dataset has been used in several studies, such as Somrudee D. et al. [12], Xiaomei X. et al. [13], and Donelly P. et al. [14]. Two datasets are provided, namely datasets in .wav and. midi formats. However, because the size of the .wav dataset is large (120GB), we used a smaller one- the Midi dataset at 81MB. Datasets consist of a folder containing several MIDI files and a .csv file containing data from all MIDI files. There are 1276 MIDI files with Grand Piano sounds collected from 10 years of competition implementation International Piano-e-Competition (2004, 2006, 2008, 2009, 2011, 2013, 2014, 2015, 2017, 2018). Because using MIDI format, all songs in this dataset only have the sound of a piano musical instrument and no noise. A limitation of this dataset is that it includes only piano sounds. To meet the needs of this research, we had to convert the piano sounds to guitar using a MIDI editor application. In addition, adding other datasets to this research could help generalize the data further. However, this research focuses exclusively on using the MAESTRO dataset.

Pre-processing is an essential technique for preparing a dataset to ensure that incoming data is clean and usable. In this research, six pre-processing stages were conducted to ready the dataset for input into the model. The first stage, Song Selection, involved choosing songs from the MAESTRO dataset, which contains 1,276 songs. Due to the large number, a subset with a total duration of 445 minutes was selected based on criteria such as appropriate length, minimal staccato technique usage, and clear audibility. The second stage, Convert Piano Files to Guitar Files, used the MidiEditor application to change the sound from a grand piano to an acoustic guitar, retaining the same musical tones but altering the instrument's sound for the training process.

Next, Convert File Format was carried out to convert the MIDI files into .mp3 format, enabling them to be processed using the Librosa library. In the fourth stage, Loading MP3 into STFT, the .mp3 files were loaded into the Python program using the Librosa library's `librosa.load()` function, resulting in the song's Short-Time Fourier Transform (STFT) signal and sample rate. The fifth stage, STFT to Mel-Spectrogram, converted the STFT signal into a Mel-spectrogram using the `librosa.feature.melspectrogram()` function, creating a matrix with dimensions of 128 by the length of the song. Finally, in the Matrix to Vector stage, the Mel-spectrogram matrix (128 columns by song

length) was converted into a vector with a size of 1 by 128, simplifying the input for machine learning models by reducing it to a single dimension for easier processing.

B. Model Architecture

The primary model in this research is the Generator-Discriminator model [15]. The generator-discriminator architecture consists of several parts. The flow of this architecture begins with the dataset preprocessing stage, followed by data processing in the generator and discriminator models. Finally, the predicted data will be passed through the postprocessing stage until it becomes an output that humans can enjoy. An image of the generator-discriminator architecture can be seen in Figure 1.

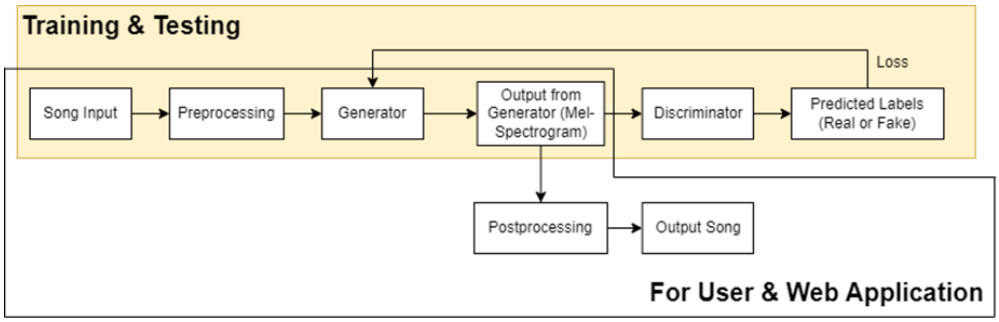

Fig. 1. Generator-discriminator architecture

The Generator Model creates a new version of an entered song by changing the musical instruments used in the song. For example, when a song is played on a piano, the generator will produce a version of the music played on a guitar and vice versa. The quality of the generator model can be measured from the output produced, which can deceive the discriminator model so that the discriminator model will assume that the output from the generator is original data. The generator architecture diagram can be seen in Figure 2.

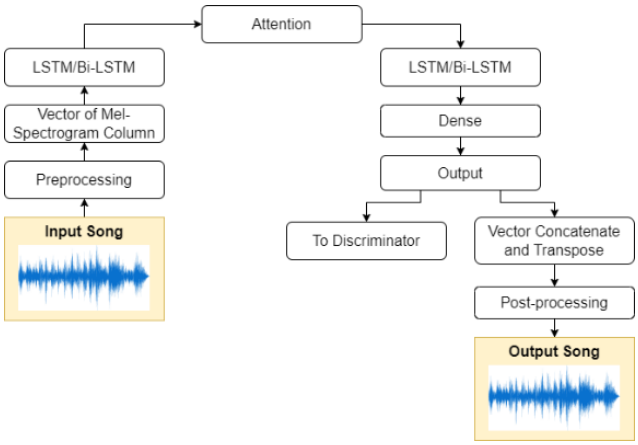

Fig. 2. Generator model architecture

The generator takes one type of input and produces one kind of output. The input is in the form of a .mp3 file preprocessed to produce mel-spectrogram column vectors. We use mel-spectrogram because mel-spectrogram is one of the most popular in sound representation, as seen in many studies such as Lu et al. [3] and Giorgi et al. [16]. This input goes through several machine-learning layers in the model. First, the input will be entered into the Bidirectional Long Short-Term Memory (Bi-LSTM) layer [17] to capture context information from both directions and understand the global context of the input data. The output from Bi-LSTM is then fed into the Attention model [18] for focusing on relevant context to improve model performance. This attention mechanism is also found in studies such as Cifka O. [10] and Guo Z. et al. [19]. The type of attention layer used is simple attention, also called global attention. Simple attention is a mechanism used in natural language processing to emphasize input elements based on their relevance to a particular target or focus element. Simple attention allows the model to calculate complex relationships between each pair of words in a sentence without regard to the order of the words. Attention weights are given to input elements based on their relevance to the target element being processed in simple attention. In simple attention, three

calculations are carried out, namely calculating the similarity score, attention score, and the new form/representation after passing through the attention layer as in (2) to (4).

$$weight * input^t = similarity_{score} \tag{2}$$

$$softmax(similarity_{score}) = attention_{score} \tag{3}$$

$$attention_{score} * input = representation_{new} \tag{4}$$

The Attention model's output is fed into another Bi-LSTM layer to enhance learning. The output from the second Bi-LSTM is then passed to a dense layer, producing a vector that has undergone timbre style transfer. For instance, if the input sounds like a piano, the output sounds like a guitar, and vice versa.

The generator's predicted vector output is evaluated using the Contrastive Loss function [20] and Binary Cross-Entropy Loss (BCE Loss). Contrastive Loss minimizes the distance between inputs that should be similar and maximizes the distance between inputs that should be different. There have been many studies that apply contrastive loss to perform contrastive learning in this domain, such as Dong et al. [21], Manco et al. [22], and Koo et al. [23]. The concept is based on calculating the distance between the model output and the target data using Euclidean distance. Contrastive Loss is calculated for each pair of input vectors x1 and x2 with a binary label y indicating whether the pair is similar or different. Contrastive loss is calculated as in (5).

$$Contranstive_{loss} = (1 - y) * d^2 + y * \max(\alpha - d, 0)^2 \tag{5}$$

BCE Loss is a loss function commonly used in binary classification problems, where the goal is to predict a binary output (0 or 1) based on input data. This function measures the difference between the label predicted by the discriminator model and the natural data labels. The smaller the BCE Loss value, the better the output produced by the generator because the discriminator assumes that the production is the actual data. Calculations for BCE Loss utilize the model output probability values and perform log-transformation of these probabilities to avoid infinite values. The smaller the value of BCE Loss, the closer the predicted results are to the actual label. BCE Loss can be calculated as in (6).

$$BCE_{loss} = y_i * log(y_{pred}) + (1 - y_i) * log(1 - y_{pred}) \tag{6}$$

The discriminator model is responsible for evaluating whether the output produced by the generator model is natural or synthetic data. If the generator model is good enough, the discriminator model will be fooled and incorrectly assume that the output produced by the generator model is accurate data. However, because the discriminator model is trained together with the generator model, the discriminator model is not that easy to fool. As the training progresses, the discriminator model will become more competent in distinguishing between natural and synthetic data. The discriminator model architecture diagram can be seen in Figure 3.

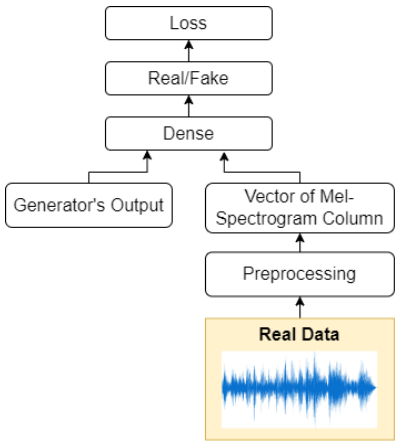

Fig. 3. Discriminator model architecture

The discriminator architecture has two types of input during the training process, which will be explained in the input and output sections. The discriminator model processes these two input types through a dense layer and a sigmoid activation function to produce output with a value range of 0 to 1. The output produced by this model is only one, the real/fake label, which has a value range of 0 to 1. The closer the value is to 0, the more confident the model is that the input provided is the output of the generator model or synthetic data, while the closer the value is to 1, the model is more confident that the feedback provided is the original data. The prediction results from the discriminator model were evaluated using the BCE Loss function described previously. The training process for the piano-to-guitar model proposed is in Pseudocode 1.

PSEUDOCODE 1. Training process for the piano-to-guitar model proposed

```
BEGIN
For each epoch DO
    For each batch(piano, guitar) DO
        SET real_data TO piano
        SET real_labels TO tensor of ones with shape (batch_size, 1)
        SET fake_data TO generator(guitar) // generate an initial fake data
        SET fake_labels TO tensor of zeros with shape (batch_size, 1)

        // Train discriminator
        CALL discriminator.zero_grad()

        SET real_outputs TO discriminator(real_data) // evaluate the real data
        SET real_loss TO criterion_real_D(real_outputs.to(device),
real_labels.to(device))
        CALL real_loss.backward()

        SET fake_outputs TO discriminator(fake_data.detach()) // evaluate the
fake data
        SET fake_loss TO criterion_fake_D(fake_outputs.to(device),
fake_labels.to(device))
        CALL fake_loss.backward()

        CALL optimizer_D.step()

        // Train generator
        CALL generator.zero_grad()

        SET g_outputs TO generator(piano) // generator model called to generate
new data
        SET d_outputs TO discriminator(g_outputs) // evaluate generator output
        SET g_loss_adv TO criterion_G_adv(d_outputs.to(device),
real_labels.to(device))
        SET g_loss_rec TO criterion_G_rec(g_outputs, guitar)
        SET g_loss TO g_loss_adv + g_loss_rec
        CALL g_loss.backward()
        CALL optimizer_G.step()
    END LOOP

    CALL scheduler_G.step() // learning rate scheduler works here
END LOOP
END
```

Pseudocode 1 is outlining the training process for the piano-to-guitar model proposed over a series of epochs, where each process processes multiple data batches. For each epoch, the code iterates through batches of data consisting of piano and guitar. The data is initially set to piano, and real labels are initialized to a tensor of ones. Fake data is generated by passing the guitar through the generator model, and fake labels are initialized to a tensor of zeros. The training process begins with the discriminator. The discriminator's gradients are reset to zero, and then it evaluates the actual data to produce real_outputs. The loss for the actual data is computed using the BCE Loss, which compares real_outputs to the actual labels. This loss is then backpropagated. Next, the discriminator evaluates the fake data (with gradients detached to prevent updating the generator during this step) to produce fake_outputs. The loss of the fake data is also computed using the BCE LOSS, which compares

fake_outputs to the fake labels. This loss is also backpropagated. The discriminator's parameters are then updated.

Following the discriminator training, the generator is trained. The generator's gradients are reset to zero, and it generates new data by processing the piano song. The discriminator evaluates this generated data. The adversarial loss is computed using BCE Loss, comparing d_outputs to the actual labels, encouraging the generator to produce more realistic data. Additionally, a reconstruction loss is calculated using Contrastive Loss, comparing g_outputs to the guitar, enabling the generator to produce data structurally like a guitar. The total generator loss (g_loss) is the sum of the adversarial and reconstruction losses and is backpropagated. The generator's parameters are then updated using Stochastic Gradient Descent (SGD). After processing all batches in an epoch, the learning rate scheduler updates the learning rate to ensure the training process adjusts dynamically. This entire process repeats for each epoch until the training is complete.

C. Input and Output

During training, the model receives piano and guitar songs as mel-spectrogram. The mel-spectrogram matrix is then transposed to produce a matrix with the size of (column, row). The generator model receives input as a mel-spectrogram matrix, cut into a vector measuring 1 x 128, with 128 columns. One song is input to the generator for timbre style transfer, while the other serves as a target for evaluation. The discriminator receives the generator's output during training, distinguishing between fake (generator output) and actual (original song) inputs. During testing, the generator processes the input song alone. For an entire song with a 100 x 128 mel-spectrogram, the model receives 100 input vectors of 1 x 128.

Because two models are used, namely the generator model and the discriminator model, the output of each model is different according to its role in the system. The generator model produces output as a vector of timbre-style transfer results. In contrast, the discriminator model produces output from real/fake labels to evaluate the generator results. All generator prediction results are formed into a list of vectors or matrices and then transposed to produce a mel-spectrogram with dimensions (rows, columns). Mel-spectrogram then goes through a post-processing stage to make a .mp3 file.

III. Results and Discussion

Optimizing Docker environments can significantly improve application performance, resource utilization, and operational efficiency. Below, we detail Docker optimization’s potential results and benefits across different dimensions.

Testing will be carried out using 6 test scenarios: ablation study, effect of hyperparameters, effect of input dimensions, test with slow tempo songs, and threshold test on the discriminator. Each experiment was carried out using ten epochs of the training process. Each trial result data will be equipped with a table to make it easier to understand the results. The information in the form of tables is hoped to increase understanding of the results of experiments on the trained, intelligent machine models.

As previously discussed in the Model Architecture section, the training loss and validation loss values in this test are calculated using the Contrastive Loss combined with the BCE Loss metrics. These two-loss metrics were selected for their specific benefits: we chose Contrastive Loss to focus on the similarity between the waveform of the output generated by the generator and the label/target data, ensuring that similar timbres are closely aligned while distinct timbres remain separated. Additionally, BCE Loss (Binary Cross-Entropy Loss) is used to evaluate whether the output of the generator resembles the original data or is still considered artificial by the discriminator, as BCE Loss is well-suited for optimizing the discriminator's performance in distinguishing between actual and generated samples.

The loss values used in the testing process, including hyperparameter tuning, the effect of the input dimension, and tests for songs with a slower tempo, are obtained using the PESQ and MSE metrics. The PESQ (Perceptual Evaluation of Speech Quality) metric was chosen for its ability to evaluate sound quality, allowing us to assess the clarity of the sound produced by the generator and identify any noise present. Meanwhile, we chose the MSE (Mean Squared Error) metric to evaluate the

accuracy of the waveform produced by the generator in comparison to the target data, providing a measure of the overall difference between the generated output and the desired result.

A. Ablation Study

The Ablation Study aims to identify each component's relative contribution and determine which elements most influence model performance. This research will test ten models and observe the resulting loss values. The model with the lowest loss value is used as the primary model. The value that will be observed is the loss value from each experiment. Detailed results of the ablation study trial are shown in Table 1.

Table 1. Test results on the ablation study

Model	Piano to Guitar		Guitar to Piano	
	Training Loss	Validation Loss	Training Loss	Validation Loss
Default	52.061	60.045	416.739	746.695
Dense Layer + ReLU	52.523	62.298	493.462	829.131
Bi-LSTM to LSTM	53.61	58.227	518.731	794.879
No Attention Layer	53.999	59.822	526.079	825.763
3 Dense Layer	54.836	62.28	569.896	854.622
1 Dense Layer	62.149	63.644	642.856	888.964
3 Bi-LSTM	64.059	69.955	556.587	885.357
Added Conv1D	66.139	69.041	1681.598	1992.078
Conv1D Without Attention	61.54	83.244	1766.178	2105.699
Conv1D + MaxPooling	72.019	78.74	2564.744	3008.396

In Table 1, experiments have been carried out with ten types of models. These ten models were trained using the same number of epochs, batch size, and learning rate. The default model is the model described in the Generator-Discriminator Architecture section. Meanwhile, other models are produced by adding or subtracting layers from the model. The following is a discussion of each model tested:

- Default Model: The best model tested is the default model because it has the lowest training loss and validation loss of the other nine models, the piano-to-guitar model obtained a training and validation loss of 52.061 and 6 0.045. In contrast, the guitar to the piano model resulted in a training and validation loss of training and validation loss of 416.739 and 746.695, respectively. The configuration for this model is Bi-LSTM->Attention->Bi-LSTM->Dense->Dense.
- Using ReLU: Adding the ReLU activation function to 2 dense layers in the model produces a loss value that is slightly larger than the default model. Because it produces unsatisfactory performance, the ReLU activation function is not used. The configuration for this model is Bi-LSTM->Attention->Bi-LSTM->ReLU(Dense)->ReLU(Dense).
- Replacing Bi-LSTM with LSTM: Replacing the Bi-LSTM layer with a regular LSTM aimed to test whether bidirectional processing was necessary. Results showed that Bi-LSTM produced smaller loss values, with training and validation losses for the guitar-to-piano model being much higher than those for the primary model. Therefore, the main model uses Bi-LSTM layers. The configuration for this model is LSTM->Attention->LSTM->Dense->Dense.
- Removal of the Attention Layer: The following test involved removing the attention layer. This test aimed to see the impact of using the attention layer. The increase in loss value from the previous model shows that the attention layer plays an essential role in the primary model, so this layer is used in the main model. The configuration for this model is Bi-LSTM->Bi-LSTM ->Dense->Dense.
- Using 3 Dense Layers: Adding one dense layer to the primary model gives it three thick layers. This layer's addition produces unsatisfactory results because it produces a higher loss value. Increasing the loss value of the default model concludes that too many dense layers are unsuitable for the model. The configuration for this model is Bi-LSTM->Attention->Bi-LSTM->Dense->Dense->Dense.
- Using 1 Dense Layer: After it was discovered that too many dense layers resulted in more significant loss, the next try was to reduce the number of dense layers by one so that the model only had one dense layer. Reducing the number of dense layers also does not provide better

results because the resulting loss value is more significant. Increasing training loss significantly means that too few dense layers are not good, so two dense layers are used. The configuration for this model is Bi-LSTM->Attention->Bi-LSTM->Dense.

- Using 3 Bi-LSTM Layers: Recognizing the importance of Bi-LSTM, an additional Bi-LSTM layer was tested after the second one. Despite decreased training loss for the guitar-to-piano model, both models showed increased validation loss, indicating overfitting. Thus, the primary model uses only 2 Bi-LSTM layers. The configuration for this model is Bi-LSTM->Bi-LSTM->Attention->Bi-LSTM->Dense->Dense.
- Addition of 1 Conv1D Layer: In this experiment, a Conv1D layer was added to the part before the first Bi-LSTM. The increase in loss value reached 2-3 times. In this way, it can be concluded that the Conv1D layer is unsuitable. The configuration for this model is Conv1D->Bi-LSTM->Attention->Bi-LSTM->Dense->Dense.
- Use of 1 Conv1D Layer and Removal of the Attention Layer: Curiosity about the previous model's significant loss value increase led to the next experiment, which involved removing the attention layer to see if it was incompatible with the Conv1D layer. While the guitar-to-piano model initially showed the increase, this time, the piano-to-guitar model did. This confirms that the attention layer is essential and cannot be removed. The configuration for this model is Conv1D->Bi-LSTM->Bi-LSTM->Dense->Dense.
- Use of 1 Conv1D Layer accompanied by MaxPooling: The final experiment used 1 Conv1D layer with a MaxPooling layer, resulting in a significant loss value increase for the guitar-to-piano model. This suggests that MaxPooling caused a loss of important information, leading to its exclusion. Since none of the Conv1D layer experiments were satisfactory, it was decided to discontinue using this layer type. The configuration for this model is MaxPooling(Conv1D)->Bi-LSTM->Attention->Bi-LSTM->Dense->Dense.

Of the ten model configurations tested, the default model, which is the primary model in this study, gave the best results. Full details about this default model configuration can be found in the model architecture section. The default model performed better than the other nine configurations. Regarding training loss, the default model outperformed all nine other model configurations. The default model outperformed the guitar-to-piano model for validation loss but was lost to the Bi-LSTM model configuration, which was transformed into LSTM on the piano-to-guitar model.

In this ablation study, several conclusions can be drawn. First, using Bi-LSTM results in a lower loss value than LSTM due to Bi-LSTM's ability to capture information from both directions of the sequence. Layer Attention is also needed to highlight the important part of the feature sequence generated by Bi-LSTM. The best number of layers for the Dense layer is 2, as too many/too few dense layers result in higher loss. Using too many Bi-LSTMs also causes a high loss value due to the overfitting factor. Lastly, the Conv1D layer and the MaxPooling and ReLU activation functions are unsuitable for this task.

From the mentioned conclusion, it is known that the configuration of 2 Bi-LSTM, 1 Attention, and 2 Dense Layer (Bi-LSTM->Attention->Bi-LSTM->Dense->Dense) is the best in this task. Each layer in the configuration has its own role, as follows:

- Bi-LSTM (First): This technique captures the temporal patterns and features of the input mel-spectrogram by accessing information from both directions of the sequence.
- Attention: Highlights the essential part of the feature sequence generated by the first Bi-LSTM, helping the model to focus on the most relevant information.
- Bi-LSTM (Second): This process further processes the information highlighted by Attention, improving the temporal context representation.
- Dense (First): Transforms the representation obtained from the second Bi-LSTM into a more compact form suitable for the next step.
- Dense (Second) refines the output of the first dense layer into a final representation best suited for the target timbre.

B. *Affect of Hyperparameter*

This subchapter discusses the experimental results by utilizing the best model from the experimental results in the ablation study to be trained using various hyperparameters. The goal is to find the best hyperparameters for the primary model. The hyperparameters that will be tested are learning rate, dropout, and batch size. The value that will be observed is the loss value from each experiment. In selecting values for the tested hyperparameters, we first tested various values randomly. Next, we selected some values with significant distances to observe the difference in the resulting loss. The trial compared four learning rates: 0.01, 0.005, 0.001, and 0.0005 as in Table 2.

Table 2. Test results on the effect of learning rate

Learning Rate	Piano to Guitar		Guitar to Piano	
	Training Loss	Validation Loss	Training Loss	Validation Loss
0.01	NaN	NaN	NaN	NaN
0.005	78.003	83.253	NaN	NaN
0.001	58.561	66.126	937.193	1303.798
0.0005	57.469	64.055	861.191	1159.181

From Table 2, the smaller the learning rate, the more stable the training process. This can be seen from the learning rate 0.01 caused by NaN (Not a Number) or infinity losses due to instability (losses fluctuate). Conversely, rates that were too small hindered learning progress. From the experiments that have been carried out, it can be concluded that the ideal learning rate is 0.0005. However, at some point, the model will usually experience convergence. Therefore, it is a good idea to implement a learning rate scheduler (e.g., reducing the rate by 0.5 every few epochs) to help manage convergence. Next is a test of the effect of dropouts shows in Table 3.

Table 3. Test results on the effect of dropout

Dropout	Piano to Guitar		Guitar to Piano	
	Training Loss	Validation Loss	Training Loss	Validation Loss
0.1	57.509	66.327	848.842	1110.385
0.3	60.131	63.589	919.171	1186.996
0.5	62.083	63.897	960.628	1182.089

In this trial, three dropout values were compared, namely 0.1, 0.3, and 0.5. From Table 3, the results shows that higher dropout values led to more significant losses due to fewer model parameters. This is caused by reducing the number of model parameters trained by the dropout value. Dropout is used to prevent overfitting. The best dropout value from these tests is 0.1, yielding the lowest loss (57.509 for training and 66.327 for validation) compared to 0.3 and 0.5. In the batch size trial, 16, 32, and 64 batch sizes were compared like in Table 4.

Table 4. Test results on the effect of batch size

Batch Size	Piano to Guitar		Guitar to Piano		Time Each Epoch
	Training Loss	Validation Loss	Training Loss	Validation Loss	
16	62.966	71.714	1094.137	1515.173	2:14
32	58.929	65.918	962.214	1345.849	1:22
64	56.394	63.31	844.255	1140.588	0:54

Table 4 shows that the larger batch sizes for this study result in smaller training and validation losses. Batch size 64 produces the lowest training and validation loss for the piano-to-guitar model: 56.394 (training) and 63.31 (validation), while the guitar-to-piano model: 844.255 (training) and 1140.588 (validation). However, substantial batch sizes can increase losses due to limited evaluation time per data batch. Meanwhile, the smaller the batch size, the longer it takes to complete one epoch. Smaller batch sizes lead to longer epochs but more frequent evaluations. Batch size 64 performed best in speed and losses, taking 54 seconds per epoch.

C. *Affect of Input Dimension*

In this experiment, the input consists of a 1-dimensional array (column vector) and a 2-dimensional array (matrix) representing the power mel-spectrogram. Both test scenarios will utilize the Contrastive

Loss, MSE Loss, and PESQ [24] metrics to evaluate model performance. The Contrastive Loss metric is used during training, while MSE Loss and PESQ are used to assess the model prediction results at the testing stage. Table 5 shows the test results using dimension one input (column vector).

Table 5. Test results on one dimension input (vector)

Model	Train Loss	Valid Loss	PESQ	MSE	
				With Target	With Input
Piano to Guitar	52.061	60.045	1.096	$3.201 * 10^{-3}$	$4.931 * 10^{-3}$
Guitar to Piano	416.739	746.695	1.1	$7.432 * 10^{-3}$	$5.713 * 10^{-3}$

From Table 5, the piano-to-guitar model had a training and validation loss of 51.061 and 60.045, while the guitar-to-piano model lost 416.739 and 746.695, respectively. The PESQ value of the guitar-to-piano model (1.1) is slightly higher than the piano-to-guitar model (1.096), indicating that the sound produced by the guitar-to-piano model is more precise. However, the Mean Squared Error (MSE) value of the piano-to-guitar model is lower than that of the guitar-to-piano model. The MSE value of the prediction results of the piano-to-guitar model is $3.201 * 10^{-3}$ with the target (guitar sound) and $4.931 * 10^{-3}$with the input (piano sound). For the prediction results of the guitar-to-piano model, the MSE value is $7.432 * 10^{-3}$ with the target (piano sound) and $5.713 * 10^{-3}$with the input (guitar sound). A low MSE value indicates that the prediction results are close to the expected results. However, for the guitar-to-piano model, the prediction results are still more like the input than the target. Therefore, a longer training process is required. The waveform predicted by the model with column vector input can be seen in Figure 4 and Figure 5.

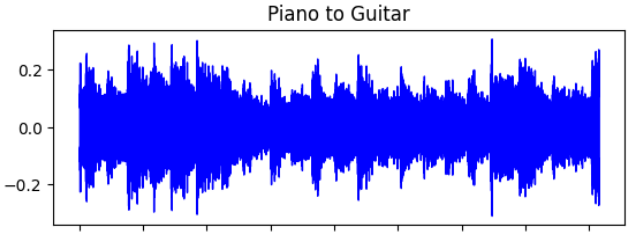

Fig. 4. Prediction results of piano to guitar model with vector input

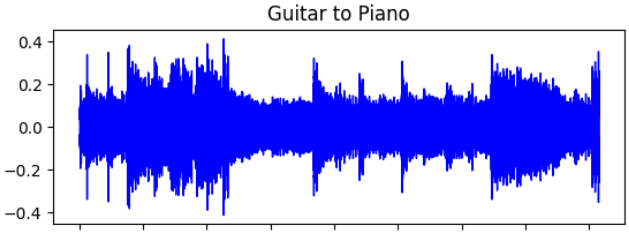

Fig. 5. Prediction results of guitar to piano model with vector input

Table 6. Test results on two dimension input (matrix)

Model	Train Loss	Valid Loss	PESQ	MSE	
				With Target	With Input
Piano to Guitar	512.168	425.389	1.112	$1.625 * 10^{-3}$	$3.347 * 10^{-3}$
Guitar to Piano	7338.361	8266.958	1.15	$4.813 * 10^{-3}$	$3.089 * 10^{-3}$

In the second trial using 2-dimensional input or a matrix, Table 6 indicates higher losses than column vector input. The piano-to-guitar model had training and validation losses of 512.168 and 425.389. Meanwhile, the guitar-to-piano model had losses of 7338.361 and 8266.958, respectively. This higher loss is due to processing more elements with the 2D matrix. The PESQ values for the piano-to-guitar and guitar-to-piano models are 1.112 and 1.15, respectively. The MSE value of the prediction results in the two models was also checked for the input and target. The prediction results of the piano-to-guitar model produce MSE values equal. $1.625 * 10^{-3}$ to the target (guitar sound) and $3.347 * 10^{-3}$To the input (piano sound). Meanwhile, the prediction results of the guitar-to-piano

model produce MSE values of $4.813 * 10^{-3}$ to the target (piano sound) and $3.089 * 10^{-3}$To the input (guitar sound). The waveform predicted by the model with matrix input can be seen in Figure 6 and Figure 7.

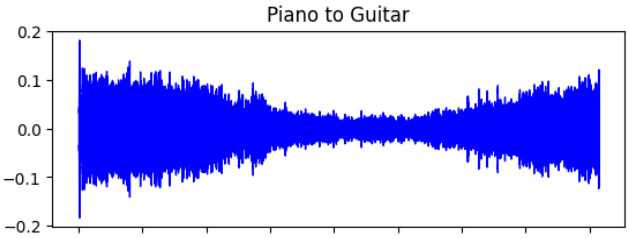

Fig. 6. Prediction results of piano to guitar model with matrix input

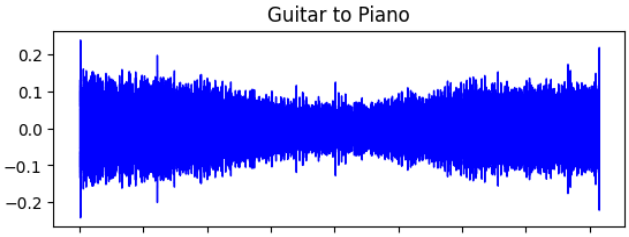

Fig. 7. Prediction results of guitar to piano model with matrix input

The MSE value of the model with matrix input is lower than the model with column vector input, presumably because the prediction results of the model with matrix input have more stable characteristics. In contrast, the prediction results of the model with vector input tend to be more varied with increasing and decreasing amplitudes. These amplitude variations can cause an increase in error values when the amplitude is too high or too low. For comparison, the wave shape of the original song can be seen in Figure 8 and Figure 9.

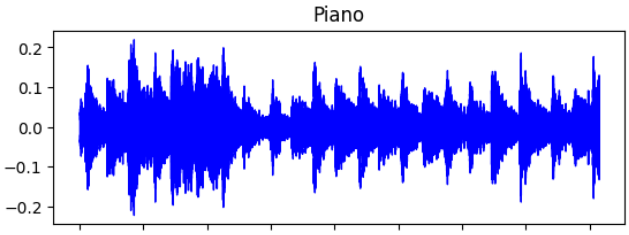

Fig. 8. Original song wave shape with piano sounds

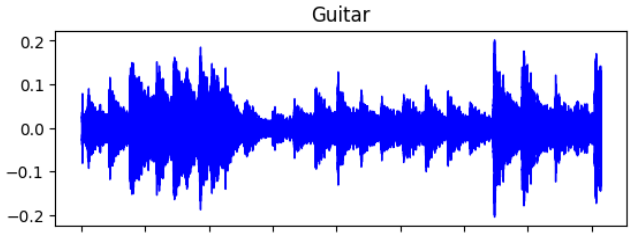

Fig. 9. Original song wave shape with guitar sounds

D. Test for Song with a Slower Tempo

This subchapter discusses the generator model's performance in handling the same song at a slower tempo. Songs with a slower tempo were changed to 0.75 times the speed of the original song. Five song samples were used in this trial for each model. Two metrics, PESQ and MSE, will be used to evaluate the results of the two songs. Table 7 shows the PESQ scores for songs tested with both trained models.

Table 7. Test results for song with a slower tempo (PESQ)

Model	Sample	PESQ	
		Sample Song	Slower Song
Piano to Guitar	1	1.155	1.075
	2	1.074	1.023
	3	1.128	1.034
	4	1.097	1.027
	5	1.086	1.094
Guitar to Piano	1	1.134	1.019
	2	1.1	1.036
	3	1.599	1.032
	4	1.199	1.025
	5	1.151	1.019

From Table 7, in the piano-to-guitar model, the original samples generally had higher PESQ scores, except for the 5th sample, where the slower song scored higher. The most considerable PESQ score difference was 0.094 in sample 3, and the smallest was 0.0085 in sample 5. Conversely, the guitar-to-piano model consistently favored original samples, with the most significant difference of 0.567 in sample 3 and the smallest of 0.064 in sample 2. Table 8 shows the test results for song with a slower tempo (MSE).

Table 8. Test results for song with a slower tempo (MSE)

Model	Sample	MSE	
		Sample Song	Slower Song
Piano to Guitar	1	$3.087 * 10^{-3}$	$5.333 * 10^{-3}$
	2	$3.256 * 10^{-3}$	$5.588 * 10^{-3}$
	3	$3.571 * 10^{-3}$	$5.669 * 10^{-3}$
	4	$3.977 * 10^{-3}$	$7.313 * 10^{-3}$
	5	$3.302 * 10^{-3}$	$4.899 * 10^{-3}$
Guitar to Piano	1	$6.825 * 10^{-3}$	$2.002 * 10^{-2}$
	2	$1.018 * 10^{-2}$	$1.532 * 10^{-2}$
	3	$1.469 * 10^{-2}$	$2.539 * 10^{-2}$
	4	$1.206 * 10^{-1}$	$2.955 * 10^{-2}$
	5	$1.553 * 10^{-2}$	$1,992 * 10^{-2}$

From Table 8, the MSE values obtained by the two piano-to-guitar and guitar-to-piano models for original and slow songs. This suggests that the results from the slower songs were more like the target. Overall, the MSE value of the original song is lower than the MSE value of the slower song. For the piano to guitar model, the range of MSE difference values ranges from $1.598 * 10^{-3}$ (sample 5) to $3.334 * 10^{-3}$ (sample 4). Meanwhile, in the guitar-to-piano model, the range of MSE difference values ranges from $4.388 * 10^{-3}$ (sample 5) to $9.104 * 10^{-2}$ (sample 4).

Overall, it can be concluded that the model can handle songs with both original and slower tempos. This is shown by the PESQ and MSE scores, which are not too far apart between the two types of songs. This shows the model's ability to maintain the quality and suitability of timbre in songs that have changed the tempo. Thus, the model generally produces quality output, regardless of the tempo changes applied.

E. Threshold Test on Discriminator

In this section, we explain the performance of the discriminator trained to evaluate the generator's output. The approach is to count the number of discriminator predictions that produce values below a threshold and divide these results by the total test data. Several thresholds have been tested, namely 0.5, 0.1, 0.05, 0.01, and 0.005.

Table 9 displays scores from the piano-to-guitar and guitar-to-piano models based on different threshold values. Scores at thresholds 0.5, 0.1, and 0.05 were not notably significant. For the piano-to-guitar model, scores of 0.5, 0.1, and 0.005 were 0.997, 0.985, and 0.971, respectively. The guitar-to-piano model scored 0.9994, 0.996, and 0.993 at these thresholds. A notable score difference appeared at a threshold of 0.01: the piano-to-guitar model scored 0.858, while the guitar-to-piano model scored 0.952. At 0.005, the piano-to-guitar model scored 0.782, and the guitar-to-piano model

scored 0.931. This indicates the discriminator's strong ability to evaluate the generator's results, with minimal confusion even at shallow threshold values, resulting in relatively high scores.

Table 9. Threshold test result on discriminator

Model	Threshold	Score
Piano to Guitar	0.5	0.997
	0.1	0.985
	0.05	0.971
	0.01	0.858
	0.005	0.782
Guitar to Piano	0.5	0.999
	0.1	0.996
	0.05	0.993
	0.01	0.952
	0.005	0.931

F. Implication and Contribution in Real Life

The timbre style transfer technique opens new opportunities for exploration and creativity in music production. With the development of this technique, music producers are expected to dig deeper into their creativity, producing unique songs with unusual instrument sounds. For example, a solo guitar performance played with piano sounds, or a classical piano composition played with guitar sounds. This technology enables a blend of styles and sounds that have never been heard before, giving a new dimension to music creation. In addition to contributing to the development of science in the music domain, this research is also expected to encourage the interest of researchers to develop it in other domains such as photography, design, etc.

In machine learning, this research can potentially encourage the development of more efficient and accurate models for handling sound data, especially in music. With this research, it is expected that there will be more and more exciting adoptions of similar techniques in the future, such as the creation of digital musical instruments that can mimic various timbres in real-time, the development of applications that can automatically change the style and characteristics of sound in musical compositions, and improvements in voice recognition technology and more natural and detailed voice synthesis.

Several real-life adaptations are based on neural style transfer techniques, such as the one used in this research. Zhao et al. [25] developed a model that focuses on creating musical accompaniments that match a given melody. This model helps musicians or music producers generate background music (such as chords, basslines, or drum patterns) that match the style and mood of a song. Radzikowski et al. [26] developed an autoencoder model and CNN to perform accent modification on speech audio. The dataset used contains English sentences spoken by Japanese people, namely the UME-ERJ dataset. Yuan et al. [27] utilized the VCTK dataset and autoencoder model to perform style transfer on human voices. This paper focuses on making it easier for computers to change how someone sounds (in terms of style), even if the computer has never heard that style before. It does this by teaching the computer to understand and separate the different parts of a voice to mix and match them more effectively. Zhang et al. [28] developed a model to generate high-quality singing voices with invisible styles, such as unique voice color, emotion, pronunciation, and articulation ability. Koutsogiannaki et al. [29] developed a way to make male voices sound more gender-ambiguous by borrowing certain speaking styles usually associated with female voices. The goal was to produce a voice that did not sound male or female but somewhere between. Marco Pasini [30] developed a MelGAN-VC model that focuses on converting one person's voice to sound like another person's voice and transferring audio styles (like making something sound from a different era or genre) over long audio samples.

However, remember that developments in this field can also have negative impacts. One of the possible adverse effects is copyright infringement, where a song can be plagiarized by simply changing the sound of the instruments. Also, using this technology can reduce creativity, as music producers may be tempted to rely on this technology to make music. Furthermore, the distinctiveness of a piece of music can be lost when musicians no longer use their full self-expression in the music creation process. Therefore, ethics and self-awareness are necessary for using technologies such as timbre-style transfer techniques. Music industry players must use these technologies responsibly and

respect the artistic values and originality in every work produced. This is important to maintain the integrity of music and ensure that technology remains a tool, not a substitute for human creativity.

IV. Conclusions

This research obtained several essential conclusions from the results of the trials that have been carried out. Through timbre style transfer, songs can transform significantly, creating a different sound from the original song. Songs that initially had a piano sound became guitar sounds and vice versa. This research utilizes the Generator-Discriminator model to perform timbre-style transfer tasks. This model was trained using several hyperparameter configurations, with an initial learning rate of 0.005 and a learning rate scheduler to adjust the training process, dropout of 0.1, and batch sizes of 64 and 128. The input type entered the model was a column vector mel-spectrogram from the song, which was converted. The trained model handled songs with the original tempo and songs with a slower tempo. The trained discriminator model can evaluate the generator results very well because the confusion level is minimal. This can be seen from the fact that even though the threshold value used is very low, the discriminator still gives a high score.

The main challenge of this research is the difficulty in obtaining an appropriate model, which makes it hard to reduce the loss values generated during the training process. This issue affects the quality of the generated waveforms, resulting in imperfect and noisy audio, particularly in the guitar-to-piano model. Choosing the proper loss function also presents a challenge, as multiple experiments are required to find a suitable one. This study's limitations include using only one type of dataset, specifically the MAESTRO dataset. The research does not utilize pre-trained models and focuses solely on developing a custom model.

For future research, it is recommended to try using an autoencoder-based model such as VQ-VAE. Using a more varied dataset is also recommended because it is hoped that it can overcome the problem of overfitting. It is also recommended to select a dataset to use a dataset that already has the sounds of all the musical instruments we want to convert, for example, in this study, the guitar and piano. Several different datasets are also highly recommended to make the dataset more generalized. Apart from that, further research can utilize pre-trained models because the large number of features that the selected pre-trained model has learned is expected to improve the prediction results of the model.

Declarations

Author contribution

All authors contributed equally as the main contributor of this paper. All authors read and approved the final paper.

Conflict of interest

The authors declare no known conflict of financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Additional information

Reprints and permission information are available at http://journal2.um.ac.id/index.php/keds.

Publisher's Note: Department of Electrical Engineering and Informatics - Universitas Negeri Malang remains neutral with regard to jurisdictional claims and institutional affiliations.

References

[1] L. Gatys, A. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," J. Vis., vol. 16, no. 12, p. 326, Sep. 2016.
[2] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, "Symbolic Music Genre Transfer with CycleGAN," in 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, Nov. 2018, pp. 786–793.
[3] C.-Y. Lu, M.-X. Xue, C.-C. Chang, C.-R. Lee, and L. Su, "Play as you like: Timbre-enhanced multi-modal music style transfer," in Proceedings of the aaai conference on artificial intelligence, 2019, pp. 1061–1068.
[4] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, "MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment," Proc. AAAI Conf. Artif. Intell., vol. 32, no. 1, Apr. 2018.

[5] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation," arXiv Prepr. arXiv1703.10847, 2017.
[6] Z. Ding, X. Liu, G. Zhong, and D. Wang, "SteelyGAN: Semantic Unsupervised Symbolic Music Genre Transfer," 2022, pp. 305–317.
[7] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "MIDI-VAE: Modeling Dynamics and Instrumentation of Music with Applications to Style Transfer." Sep. 20, 2018.
[8] O. Cifka, A. Ozerov, U. Simsekli, and G. Richard, "Self-Supervised VQ-VAE for One-Shot Music Style Transfer," in ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Jun. 2021, pp. 96–100.
[9] S.-L. Wu and Y.-H. Yang, "MuseMorphose: Full-Song and Fine-Grained Piano Music Style Transfer with One Transformer VAE." May 09, 2021.
[10] O. Cifka, U. Simsekli, and G. Richard, "Groove2Groove: One-Shot Music Style Transfer With Supervision From Synthetic Data," IEEE/ACM Trans. Audio, Speech, Lang. Process., vol. 28, pp. 2638–2650, 2020.
[11] Y.-N. Hung, I.-T. Chiang, Y.-A. Chen, and Y.-H. Yang, "Musical Composition Style Transfer via Disentangled Timbre Representations." May 30, 2019.
[12] S. Deepaisarn, S. Chokphantavee, S. Chokphantavee, P. Prathipasen, S. Buaruk, and V. Sornlertlamvanich, "NLP-based music processing for composer classification," Sci. Rep., vol. 13, no. 1, p. 13228, Aug. 2023.
[13] X. Xue and Z. Jia, "The Piano-Assisted Teaching System Based on an Artificial Intelligent Wireless Network," Wirel. Commun. Mob. Comput., vol. 2022, pp. 1–9, Jan. 2022.
[14] P. J. Donnelly and V. Ebert, "Transcription of audio to midi using deep learning," in 2022 7th International Conference on Frontiers of Signal Processing (ICFSP), IEEE, 2022, pp. 130–135.
[15] I. Goodfellow et al., "Generative adversarial networks," Commun. ACM, vol. 63, no. 11, pp. 139–144, Oct. 2020.
[16] B. Di Giorgi, M. Levy, and R. Sharp, "Mel Spectrogram Inversion with Stable Pitch." Aug. 26, 2022.
[17] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[18] A. Vaswani et al., "Attention Is All You Need." Jun. 12, 2017.
[19] Z. Guo, J. Kang, and D. Herremans, "A domain-knowledge-inspired music embedding space and a novel attention mechanism for symbolic music modelling," in Proceedings of the AAAI Conference on Artificial Intelligence, 2023, pp. 5070–5077.
[20] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06), IEEE, pp. 1735–1742.
[21] D. Yao et al., "Contrastive Learning with Positive-Negative Frame Mask for Music Representation," in Proceedings of the ACM Web Conference 2022, New York, NY, USA: ACM, Apr. 2022, pp. 2906–2915.
[22] I. Manco, E. Benetos, E. Quinton, and G. Fazekas, "Contrastive Audio-Language Learning for Music." Aug. 25, 2022.
[23] J. Koo, M. A. Martínez-Ramírez, W.-H. Liao, S. Uhlich, K. Lee, and Y. Mitsufuji, "Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects." Nov. 03, 2022.
[24] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221), IEEE, pp. 749–752.
[25] J. Zhao and G. Xia, "AccoMontage: Accompaniment Arrangement via Phrase Selection and Style Transfer." Aug. 25, 2021.
[26] K. Radzikowski, L. Wang, O. Yoshie, and R. Nowak, "Accent modification for speech recognition of non-native speakers using neural style transfer," EURASIP J. Audio, Speech, Music Process., vol. 2021, no. 1, p. 11, Dec. 2021.
[27] S. Yuan, P. Cheng, R. Zhang, W. Hao, Z. Gan, and L. Carin, "Improving Zero-shot Voice Style Transfer via Disentangled Representation Learning." Mar. 16, 2021
[28] Y. Zhang et al., "StyleSinger: Style Transfer for Out-of-Domain Singing Voice Synthesis," in Proceedings of the AAAI Conference on Artificial Intelligence, 2024, pp. 19597–19605.
[29] M. Koutsogiannaki, S. M. Dowall, and I. Agiomyrgiannakis, "Gender-ambiguous voice generation through feminine speaking style transfer in male voices." Mar. 12, 2024.
[30] M. Pasini, "MelGAN-VC: Voice Conversion and Audio Style Transfer on arbitrarily long samples using Spectrograms." Oct. 08, 2019.