Expert Systems With Applications 191 (2022) 116195

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

ComposeInStyle: Music composition with and without Style Transfer

Sreetama Mukherjee [a], Manjunath Mulimani [b,*]

[a] Microsoft R&D Pvt. Ltd., Hyderabad, 500 033, India
[b] Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, 576 104, India

ARTICLE INFO

Keywords:
Music composer classification
Style transfer
Generative Adversarial Networks (GAN)
Hybrid model
Musical Instrument Digital Interface (MIDI)
Piano rolls

ABSTRACT

Every music composition has a composer at the core of its building block, molding it into a style of their own. The creative compositional style of a composer varies dynamically with every composer which is perishable and inimitable but cannot be preserved. This paper proposes a hybrid style transfer model which is an end-to-end approach to transfer style, compose as well as evaluate the style transfer accuracy in the domain of the target music composer. The paper focuses on 3 composer maestros Liszt, Chopin and Schubert taken from the Maestro dataset. The main step of music composer style transfer is accomplished in 3 systematic steps, training composer classifiers according to feature sets, generating a model of a composition in a particular composer's style from noise, and finally style transfer and its evaluation. The 3 steps are interconnected in a way that each step is the building block for the next step. The GAN architecture of the style transfer step is built out of the GAN architecture of the second step. The compositions generated from each architecture is finally evaluated by the common pre-trained classifier of the first step. The highest accuracy obtained is 80% for composer classification using the maestro dataset, 77.27% for the classification of the generated style transferred version of a composition into the target composer class using the pre-trained classifiers.

1. Introduction

In today's world of technological advancement, machines are the most trusted assistants to humans. Adding on to the expertise of human beings, machines are giving us immense value add in every aspect of our lives with the help of Artificial Intelligence (AI) and Machine Learning (ML). Starting with analytics to generative networks (Goodfellow, 2016), machine learning has touched almost every domain be it information technology sector, medical sector or even artistic sectors like painting, music. With the recent boom of ML/AI, the impact has been so huge that the nature of jobs for human beings are about to change. Inspired from the wonders of ML/AI in the domain of paintings in which machines could generate paintings of its own, similar creative domains such as music have also started evolving.

Music is such a creative gift that has endless impact on human beings. At the center of every musical piece it is the music composer who primarily defines the styles of the musical piece. Although one composer can compose music of different genres, the style of every composer is unique. The machine learning models which focus on classifying music based on genre like classical, jazz, pop, rock and others, do not focus on the composer specific styles. Because a composer can compose music on any genre, studying on genre specific music does not capture the composer styles which is unique for every individual. The main challenge here is to transfer the compositional style of one composer to another composition which was earlier composed by another music composer. Our human brain is wired in such a way that we tend to listen more to the musical compositions of the specific composers who touches our soul. However, composition is such a creative talent that cannot be imitated or passed on. No two composers are alike in the styles of their composition. Thus, music composition is a talent that dies with the composer. In this way, machines can help in preserving the unique styles of each composer through their works thereby making them omnipresent. In this paper, an attempt has been made in a systematic manner to generate composer specific compositions with and without style transfer. The block diagram of the proposed method is shown in Fig. 1 which is described in detail in the 1.2 subsection. Initially the flow of work starts from data pre-processing, identifying the suitable features, to classifying the composers, generating compositions specific to a composer and finally performing style transfer between two composers. This is a wonderful journey that would definitely help

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: sritmukh@gmail.com (S. Mukherjee), manjunath.gec@gmail.com (M. Mulimani).

https://doi.org/10.1016/j.eswa.2021.116195
Received 28 September 2020; Received in revised form 2 November 2021; Accepted 2 November 2021
Available online 27 November 2021

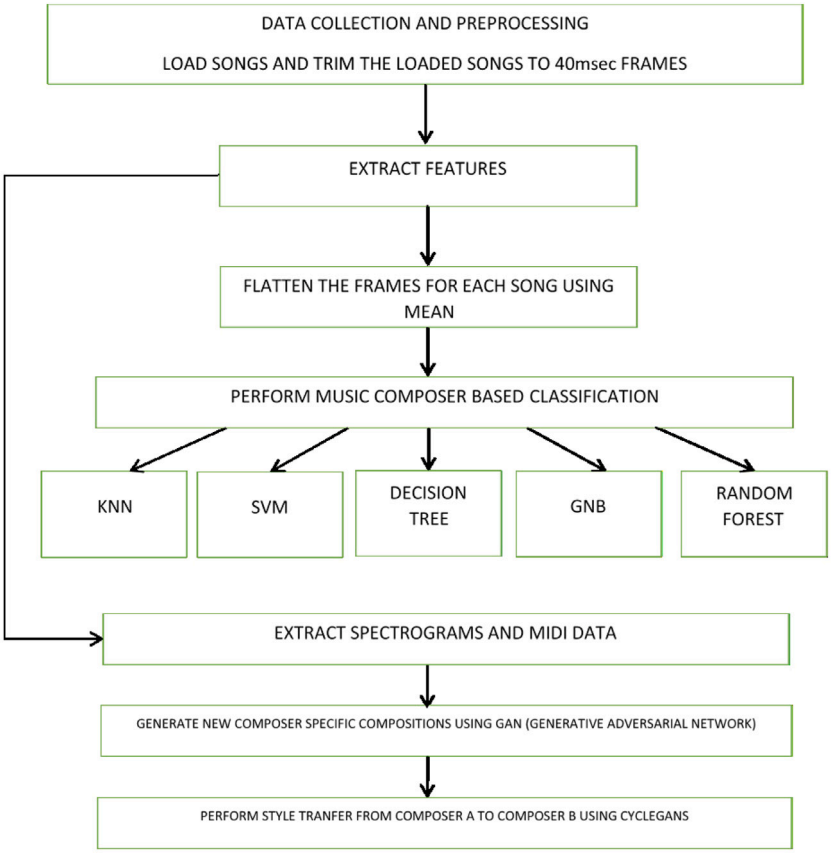

Fig. 1. Block diagram demonstrating the entire style transfer steps in detail.

the musical creativity evolve in unique ways with the help of machine learning worldwide.

Although recent literature reveals a few study on music genres (Chuan, 2013; Goulart, Maciel, Guido, Paulo, & da Silva, 2011; Panteli, Bittner, Bello, & Dixon, 2017), but music composer specific study is very few. Style transfer itself being a very recent concept, this paper explores more into the recent advancements and applicability of style transfer. Using the latest state of the art architectures, a hybrid model is developed which capture the entire objective of this paper in a nutshell. Although there remains significant room for improvement, however it lays the foundation for an end-to-end composer specific study. Since there is no exactly relevant work on music composer style transfer, comparison has been done by implementing the model architecture in a contextually similar work on genre style transfer. The entire setup is same except the model architecture of the comparison paper. This gives an opportunity to compare and contrast the advantages of our proposed hybrid style transfer architecture as compared to another style transfer model in a recent work by Brunner, Wang, Wattenhofer, and Zhao (2018b).

1.1. Motivation

The main motivation towards this work lies mainly due to the negligible attempts made in literature for exploring the music composer-based domain be it in classification, generation or even style transfer. The major distinction between our study and the ones that has already been done is that we focus on the composer domain and the recent work in this domain focus only on the genre style domain. The composer domain can be thought of as a super set of the genre style domain. The composer domain consists of the individual composer styles that can vary based on the requirement. The same composer can compose music in all styles like jazz, pop, blues, etc. depending on the requirement. Moreover, the styles are also unique in the sense that each composer can have their own compositional patterns, unique tempo, pitch styles. The most exciting part is that the composers play a central role in any music composition, exploring which will help us understand and preserve human creative thinking. Literature reveals a significant amount of work on various domains such as music genres, cultural music, song artist identification, etc. but very minimal research effort have been dedicated to the field of music composers. Composer specific style transfer is a new area of applying style transfer. Style transfer itself being a relatively new concept, not much contribution has been done in this field with respect to the musical composition. Most applications include images and in the field of music its mostly genres. Therefore, taking all other related works into consideration, this would be the first attempt towards performing style transfer among music composers leveraging the benefits of generating unique composer specific compositions.

1.2. The proposed strategy

In this section, Fig. 1 is described in detail. In the first step, the feature sets are prepared as follows:

- Each song is divided into 1000 frames of 40 ms duration with 50% overlap. This would ensure minimal information loss.
- Feature set is extracted from each of the frames consisting of 28 features comprising of the zero-crossing rate (Le, Ambikairajah, Epps, Sethu, & Choi, 2011), the spectral centroid (Giannakopoulos, 2009), the spectral rolloff (Bartsch & Wakefield, 2005; Zheng, Zhang, & Song, 2001), 13 Mel-Frequency Cepstral Coefficients (MFCCs) (Lederle & Wilhelm, 2018) and 12 chromagrams (Wakefield, 1999).
- After dividing each song into 1000 frames consisting of the above 28 features, each song is then flattened by taking their mean.

Now the dataset consisting of 100 songs of each of the 3 composers, are classified using various well-known ML models like:

- Basic Models: Decision Tree, K-Nearest Neighbors (KNN), Gaussian Naive Bayes (GNB), Support Vector Machine (SVM) using both linear and radial kernels, Random Forest.
- Ensemble Models: Bagging, Boosting and Stacking of few of the above models.

These classification models are then used to verify the effectiveness of the generated music in the later steps.

The second step involves the generation of composer specific music. User gives composer A of choice as input to the model. The model outputs the generated compositions in the style of preferred music composer. The generation of music is done from scratch taking random noise as input which is then trained to generate melodies.

The third step revolves around transferring the style of one composer to the other. Given an audio input of composer A, the model outputs the same audio in the style of preferred composer B. The model is developed as a hybrid of all of the 3 major steps. The user specifies the audio of composer A which is to be style transferred along with the label of the chosen composer B. The final hybrid model outputs the unique compositions of composer A (generated using the model in step 2), style transferred version of composer A→B (in step 3). The final hybrid model also combines the pre-trained classification models of step 1 which is then used for the purpose of evaluation. In this way, coupling of the individual steps promote sharing of models and parameter sharing which help in reducing training complexities. During the training phase of the proposed hybrid style transfer model, the training is done for both the composers A and B such that the model is capable of transferring style from A→B or B→A. The final model is thus capable of producing the following outputs as per the requirement:

- Style transferred compositions A→B.
- Style transferred compositions B→A.
- Unique compositions in the style of composer A from noise.
- Unique compositions in the style of composer B from noise.

1.3. Contributions

The major contributions of this paper is to empower the machines to generate composer specific style transferred compositions. The unique advantages of this are as follows:

- **Multi-styled composition:** In this way, the same composition can be created in the styles of various other composers of our choice.
- **Composer selection:** This can give the opportunity for comparing the various compositions as composed by different composers and choose the most suitable compositional style adhering to the requirement.
- **Recreating compositions:** This would also help to recreate the compositions in the style of talented composers who are no longer alive.
- **Listener specific generation:** This would also help in creating compositions corresponding to the individual tastes of the listener as per choice.

The paper is organized in a very systematic manner adopting a step by step approach towards accomplishing the desired research objectives. Initially, the compositions are classified into the different composer classes using various techniques. Then, the compositions specific to a particular composer of choice is generated using vanilla Generative Adversarial Network (GAN) (Goodfellow, 2016). Finally, style transfer is performed on the musical composition of composer A to generate compositions in the style of another target composer B of choice. The rest of the paper is organized as follows, Section 2 reviews the existing works on music composition. Section 3 contains explanation of the proposed methodology in brief. Experiments and results are given in Section 4. The discussions, limitations and conclusions are given in Section 5.

2. Related works

Music composition is a non-trivial artistic task which requires balanced combination of various parts such as pitch, rhythm, timbre, loudness, beats to mention few. Changing any such major components changes the composition altogether. Accordingly, each composer has their own way of combining these musical components which creates their own style of composing music. Although various attempts have been recorded in the literature to compose music using vanilla GAN such as (Abdulatif, Armanious, Guirguis, Sajeev, & Yang, 2019; Dong, Hsiao, & Yang, 2018; Dong & Yang, 2018; Kaneko, Takaki, Kameoka, & Yamagishi, 2017; Marafioti, Perraudin, Holighaus, & Majdak, 2019), they are not specifically for composers. In this paper therefore an attempt has been made at composing composer style specific compositions which has not been done before. The architecture has been described later in the paper in details.

Among many attempts at generating music at par with paintings, researchers have made several attempts starting from Variational Auto Encoder (VAE) (Brunner, Konrad, Wang, & Wattenhofer, 2018a; Luo, Yang, Ji, & Li, 2020) to modern GANs (Abdulatif et al., 2019; Dong et al., 2018; Dong & Yang, 2018; Kaneko et al., 2017; Marafioti et al., 2019). In this section, the most recent development in music composition using GAN in the (2016–2020) timeframe has been discussed in detail. GAN was introduced by Goodfellow (2016). It describes the advantages of using GAN over other generative networks. Another paper (Zhu, Park, Isola, & Efros, 2017) introduces the architecture for cycle GAN which is also a significant advancement in the field of generative networks. Cycle GANs were made keeping in mind the use case in which paired data is unavailable for training GANs. It introduces a cycle consistency loss in addition to the adversarial loss in order to evaluate the GAN performance in training with unpaired data. The context of application of this method is described in the domain of images. In this method, the source image is converted to the target image (A→B) and back to the source image (B→A), the cycled source image. This is performed using 2 generators (G and F) and 2 discriminators (D_x and D_Y) (Zhu et al., 2017). The cycled source image is compared with the real source image which in turn trains the target generator in turns. This concept which was initially applied to images, is now applied to the music domain as well. Now beginning with the various applications of GAN in practical use cases. (Zhang, Shu, Guo, Zhang, Xie, & Liu, 2020) proposes an automated oral evaluation technique suitable for language learning of children. Here, they propose 2 models Neural Audio Caption Model (NACM) and the Generative Adversarial Network-based Neural Audio Caption Model (GNACM). The first model uses Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM) to discover the mappings from the audio features and the corresponding text features (captions). The next model GNACM uses a GAN based framework with the output of NACM as input to the GAN in Zhang et al. (2020). This attempts to improve the quality of the generated comments. Other various applications of GANs have been explained in the survey paper by Yu (2020). Here, they give a detailed description of the pros and cons of using GANs over other generative networks.

In the field of music, GAN is found to be used only in the very recent works (Abdulatif et al., 2019; Dong et al., 2018; Dong & Yang, 2018; Kaneko et al., 2017; Marafioti et al., 2019). Although (Hantrakul, Engel, Roberts, & Gu, 2019) applies WaveNet (WaveRNN) for musical audio synthesis, (Engel, Agrawal, Chen, Gulrajani, Donahue, & Roberts, 2019) demonstrate the practical advantages of using GANs over WaveNets (Oord et al., 2016) by generating log magnitude spectrograms of musical data. Literature review shows various approaches

of music generation such as generation from spectrograms or generation of symbolic music. Spectrograms are the time frequency representation of musical audio. Although it is comparatively easy to generate spectrogram of a given audio but generating music from spectrogram data is non-trivial. (Kumar et al., 2019) takes an initial step at generating raw audio waveform through mel-spectrogram inversion. They have used a non-autoregressive CNN based GAN for this purpose. This finds application in many fields such as text-to-speech, music translation and music synthesis with slight quality degradation. On the other hand, (Chen, Wang, & Yang, 2019) describes the process of raw audio generation from symbolic music (piano rolls) using CNN models. (Marafioti et al., 2019) proposes GAN models to generate time frequency based raw audio waveforms. The GAN model is trained on time-frequency features which produce reliable and invertible Short Term Fourier Transform (STFT) representations which can be converted to raw audio waveforms. (Liu & Yang, 2018) focuses on producing lead sheets for musical audio. Lead sheets are musical notations for a particular song. Using Recurrent Convolutional Neural Network (RCNN) model, the authors propose better learning from Musical Instrument Digital Interface (MIDI) and unpaired lead sheets. (Abdulatif et al., 2019) have explored the domain of denoising audio tracks using GANs. Generating music from symbolic piano rolls result in real-valued piano rolls which must be post-processed in order to obtain the final binary valued results. (Dong & Yang, 2018) aims to solve this process of binarizing generated piano rolls using GANs. The network is trained in 2 stages. The generator and the discriminator are pretrained. Then the refiner binary neurons network is trained along with the discriminator in order to obtain better binarized values. It shows that it works better than Hard Thresholding (HT) and Bernoulli Sampling (BS) methods popularly used. In another work by Dong et al. (2018), 3 GAN models (jamming, composer and the hybrid model) are trained for music composition generation. The models generate piano rolls of 5 tracks namely, bass, drums, guitar, piano and strings. The models vary in architecture and the authors also extend it to generate accompanying tracks for human composed music. Another work by Kaneko et al. (2017), focusses on spectrograms. They have proposed a GAN based post filter to reconstruct high fidelity STFT spectrograms. The GAN is trained by divide and conquer by dividing the dataset with overlap and then concatenating the generated output again with overlap in order to minimize information loss. In the paper (Johnson, 2017), a set of parallel, tied weight Recurrent Neural Network (RNN) is used to predict and generate polyphonic music compositions which is transposition invariant. 2 models namely, Tied Parallel - Long Short Term Memory - Neural Autoregressive Distribution Estimator (TP-LSTM- NADE) and Biaxial Long Short Term Memory (BALSTM) are used for this purpose. Sheet music is explored by Agarwala, Inoue, and Sly (2017) in which they use RNN models to generate sheet music. The problem of generator differentiation which occurs when generator is supposed to generate discrete tokens instead of real valued tokens has been address by Yu, Zhang, Wang, and Yu (2017) In this, the authors attempt to solve the problem using a sequential framework called SeqGAN. Here, the generator is modeled as a stochastic policy in Reinforcement Learning (RL). The corresponding intermediate state–action steps get the rewards using Monte-Carlo search after the GAN discriminator judges on a complete generated sequence. Another work by Yang, Chou, and Yang (2017) demonstrates the generation of symbolic music using GAN either from scratch, followed by a chord sequence or a priming melody. RNN is used in 2 more recent works by Colombo, Muscinelli, Seeholzer, Brea, and Gerstner (2016), Mogren (2016) to generate music. (Mogren, 2016) proposes a Convolutional-RNN-GAN (C-RNN-GAN) to generate classical music trained n sequential data.

Style transfer is a new dimension in the field of machine learning and AI. The freshness of style transfer can be traced back to (Gatys, Ecker, & Bethge, 2016) who were the first to introduce this concept. Although the initial and many further attempts for style transfer have been incorporated to images mainly, other artistic fields such as music shows comparatively very few applications of style transfer. Moreover, different authors attempt to apply the concept of style transfer to the context of music in myriad ways using different network models. (Dai, Zhang, & Xia, 2018) presents a survey paper which provides the definition of music style transfer in general. It explains each of timbre style transfer, performance style transfer and compositional style transfer in details. Although, various attempts have been made to transfer style among various genres however, this would be the first work which combines the benefit of both creating unique composer specific compositions as well as transferring styles among music composers. As also mentioned in the introduction, this will not only foster more creative dimensions of machines in music, but also would help preserve the unique composer specific styles for days to come. Some of the latest research developments (2016–2020) in the field of style transfer as applied to music is described in detail below.

The Luo et al. (2020) explored the field of Chinese folk song generation by capturing specific music styles using VAE. In this paper, authors have used 2 classifiers to separate style and content. Finally, evaluation is done mainly using human evaluation, reconstruction accuracy and style recognition accuracy. In the field of hybrid machine learning models, some of the recent works like Zhou et al. (2020) have also demonstrated the power of hybrid models for achieving complex tasks like predicting tunnel boring machine performances. In the field of classification also, some of the recent works like Zhou, Li, and Mitri (2016) compares and contrasts the effectiveness of various classification models using classification rate and Cohen's kappa as the metrices. In the field of genre rearrangement, (Hung, Chiang, Chen, Yang, et al., 2019) has made an effort for arbitrary genre rearrangement using instrumental real-world music. Encoder/Decoder architecture has been used along with adversarial training to learn music representations and skip connections have been used for pitch information. Human evaluation along with instrument activity detection has been used as the evaluation metrics for this step. WeChat based application, authored by Liu and Yang (2018) gives the users an opportunity to generate random audio based on the recording provided by the user. Expectation analysis based on the user recording as motif is used to convert it into piano music which the user can also rearrange. Human evaluation is used as the only metric for evaluation. Bao et al. (2019) composes music from lyrics. Using 2 neural encoders, the lyrics and the context melody are encoded and one hierarchical decoder jointly produces musical notes and corresponding lyrics alignment. Human evaluation along with automatic evaluation are used for the purpose of evaluating the generated music. Weiß, Brand, and Müller (2019) exploits the Baum–Welch algorithm to generate genre based western music which is in turn evaluated using Gaussian Mixture Model (GMM) classifiers. Accuracy is chosen as the metric for evaluation in this case. In this paper, the authors extract soft mid-level features capturing the chord transitions. Nakamura, Shibata, Nishikimi, and Yoshii (2019) performs genre based style conversion (say classical to pop) using unsupervised models. Pitch and rhythm organization is captured using unsupervised models. Edit model extracts syntactic function of notes in an unsupervised manner. Yet another work on genre style transfer by Brunner et al. (2018a) is performed using cycle GAN. The authors convert MIDI music from one genre to another (say jazz to pop) which is evaluated using neural style classifiers. Colombo and Gerstner (2018) focuses on capturing the note transition probabilities of songs using a deep LSTM network. The network has been trained on British and American folk songs. Evaluation is done on CrowdAI platform along with human evaluation. Mao, Shin, and Cottrell (2018) generates music by improved version of biaxial LSTM which presents an augmented representation of music dynamics and beats. Human evaluation has been used for evaluating the performance. Wang and Tzanetakis (2018) performs a singing style investigation on pop music using variants of neural network. CNN in Siamese architecture have been used as the model which has been evaluated using KNN classifiers. Another recent work by Lu, Xue, Chang, Lee, and Su (2019) focuses on genre specific

music style transfer (piano, guitar, string quartet). Unsupervised multimodal music style transfer for one-to-one generation is done using Relativistic average Generative Adversarial Networks (RaGAN). Multichannel timbre features are used to improve sound quality. Human evaluation is performed using mean opinion scores. Brunner et al. (2018b) also performs genre style transfer and interpolation between medleys and song mixtures using neural network based VAE. Style validation classifiers are used for evaluating the generated music. Anantha bhotla and Paradiso (2017) presents prototyping of musical ideas by style transfer between the existing soundtracks. Various pipelines including training of Markov models are performed here. For the purpose of evaluation, human evaluation is considered here as well. Another work by Choksi, Sawant, and Mali (2017) also separates style and content from music and blends them using CNN models. Here also evaluation is performed by humans.

Among some of the very recent works in the music composition domain, De Prisco, Zaccagnino, and Zaccagnino (2020) focusses on solving the problem of unfigured harmonization using an evolutionary approach. It deals with 4-voice harmonization problem trained on a single composer. Composition in this paper refers to the creation of other 3 voice harmonizations when supplied with one harmonization. There is no style transfer involved among different music composers which is the main difference between our work and this recent work. Another recent work De Prisco, Malandrino, Zaccagnino, Zaccagnino, and Zizza (2017) shows a bio inspired way of composing music in a certain style. This work is thus a subset of our proposed approach. Moreover, in our approach we propose a cycle GAN way of training in presence of unpaired data.

3. Hybrid model for composer style transfer

The main aim of this paper is to perform style transfer among music composers and get an idea of how a certain music compositions would sound like if it was composed by different music composers in their own unique compositional styles.

The steps involved in composer style transfer using proposed hybrid model are given below.

1. Perform music composer classification using the extracted feature set by basic and ensemble models.
2. Generate music compositions from noise.
3. Generate style transferred version of composer A's music in the style of target composer B.
4. Evaluate the style transfer accuracy using the pre-trained classifiers in step 1.

Input to the proposed hybrid model is a composition composed by composer A. The corresponding output is the style transferred composition in the style of composer B. The evaluation of the generated style transferred composition is performed using the pre-trained classifiers which are a part of step 1 of the proposed hybrid algorithm. The entire procedure is performed in a step by step manner by training the classifiers first, followed by the training of the GAN models for generating random compositions in the style of a composer A from noise and then performing style transfer using the final style transfer model to generate composer A's compositions in the style of the target composer B. The corresponding models are described below in detail.

3.1. Architecture of the hybrid model

The hybrid model in Fig. 2 visualizes the architectures of the research objectives in a nutshell. There are 2 GANs (each with 2 generators) along with 2 common discriminators. The 2 GANs have different functionalities. The vanilla GAN (in step 2) as shown in Fig. 3 generates compositions from scratch. Given noise as input, the vanilla GAN can generate compositions in the style of the preferred composer of choice. The style transfer GAN architecture focuses on keeping the content of composition by composer A and outputs the melody in the style of the preferred composer B of choice. Finally, the performance of the individual models have been evaluated on the basis of the accuracy that is achieved while classifying the generated compositions by the basic and ensemble pre-trained classifiers of step 1 of the proposed hybrid algorithm. The classifiers in step 1 were trained using raw audio of the 3 composers and then tested on the test set.

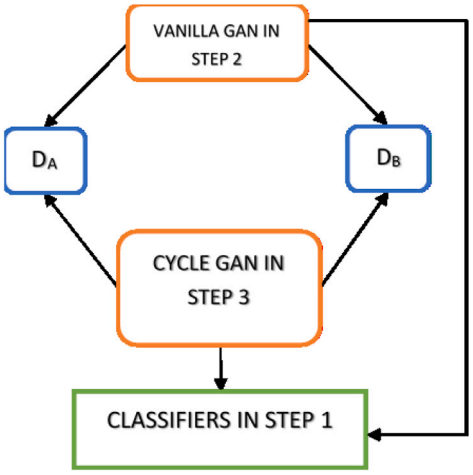

Fig. 2. Proposed hybrid style transfer model.

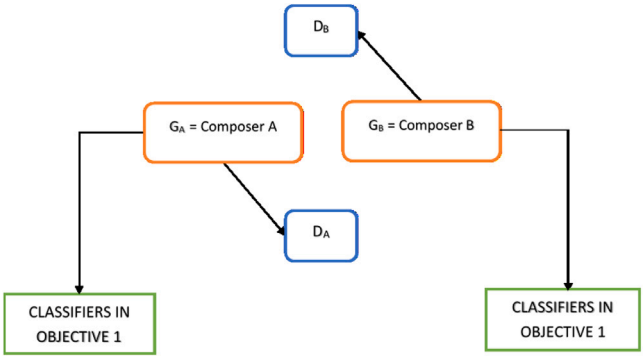

Fig. 3. Architecture of using Vanilla GAN for generating compositions from noise in Step 2.

3.2. Architecture of the vanilla GAN

The audio melodies thus generated are in MIDI format which is then converted into raw audio format (wav). Features are then extracted from them in a way similar to the process employed in step 1. The corresponding feature vector is constructed comprising of 1000 frames per song and 28 features per song which consists of (12 chromagram features, 13 MFCC features, 1 zero-crossing rate feature, 1 spectral centroid and 1 spectral roll off feature). The feature set is then flattened using simple mean resulting in 1 frame per song. This (n x m) feature vector is then used fully for the purpose of testing only. The models above were saved after being trained and tested in step 1. These models are now used to evaluate the performance of the generated compositions by the corresponding prediction accuracies. The better the accuracies of a generated composition being predicted to be in the desired label of the composer the more successful is the generation.

3.2.1. Description of GAN architectures G_A, G_B, D_A and D_B

In step 2, paired vanilla GAN has been trained to generate multitrack polyphonic music. In this step, composer specific melody is

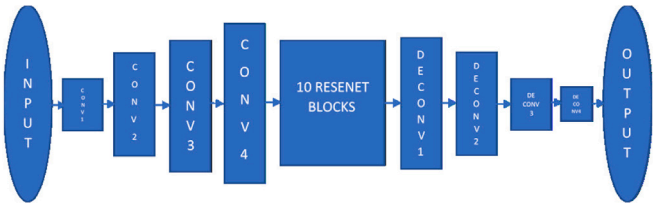

Fig. 4. High level generator architecture as used in the proposed hybrid model.

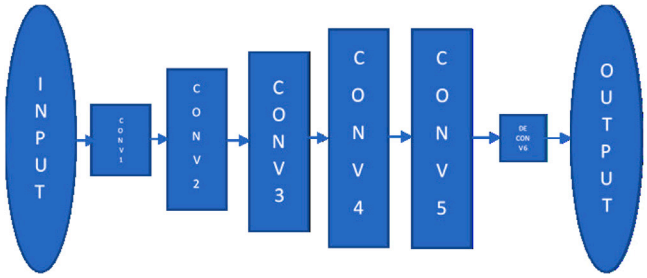

Fig. 5. High level discriminator architecture as used in the proposed hybrid model.

generated from noise. The overall architecture of a single GAN consists mainly of 2 parts: Generator and Discriminator 4. They are the 2 players in the minimax game of the GAN. The architecture of each of the individual components is described in detail below.

Individual Architecture of a single GAN:

1. **Generator:** The generator which is used here for step 2 comprises of the following high-level components:
2. **Discriminator:** The discriminator used in step 2 consists of the high-level components as shown in Fig. 5.

The main parts of the architecture are detailed below.

(1) **Generator:** Generator works similar to the encoding–decoding architecture. The model as shown in Fig. 6 contains 4 Conv layers and 4 DeConv layers. The convolutional layers down sample (encode) the input audio matrix in other to extract high level to low level information. This is then fed to the 10 ResNet blocks which have skip connections to produce improved interpretations and reduce the problems of vanishing and exploding gradients. The output from the ResNet block is then fed to the deconvolutional layers which perform the up sampling (decoding) to regenerate the audio in original desired structure. Instance normalization has been used in each of the layers instead of batch normalization as it is more capable of applying the style of composer B during style transfer from composer A to B. Also here, experimentation has been done to test the performance with batch normalization which yields very poor compositions. The generator is not trained directly but it tries to minimize the discriminator adversarial loss for marking the fake generated audio as real audio. That is, the generator always tries to fool the discriminator with fake data as real data. Also, the generator is updated based on the effectiveness of regeneration of the source audio across 2 hops (A → B → A).

(2) **Discriminator:** The discriminator consists of 5 Conv layers with various types of activations as shown in Fig. 8 like leaky RectiLinear Unit (ReLU) and Parametric RectiLinear Unit (PReLU). The discriminator is trained using mean squared loss with beta set to 0.5. This ensures that updating of the model is done as half the usual effect. This ensures slow and stable updating of the discriminator to avoid unstable training.

(3) **ResNet block:** The ResNet block as shown in Fig. 7 is comprised of 2 Conv layers with instance normalizations. Finally, the generated output and the input audio matrix are concatenated channel wise which brings the effect of skip connections. This output is then given to the de-conv layers of the generator. The effectiveness of the ResNet blocks is to have skip connections in order to reduce the problems of vanishing and exploding gradients.

3.3. Style transfer

The basic concept of style transfer in the context of music is to amalgamate the style of one audio with the content of another audio 9. The main components that constitute the artistic style of a musical composition are its patterns, timbre, loudness, use of silences, etc.

Input: The input to the style transfer model are 2 audios (real content audio and real style audio). Content audio is the audio which needs to be transferred to another style. Style audio is the audio in which style the content audio is to be transferred to. The content audio is the composition by composer A and style audio is the composition by composer B.

Output: The output of the style transfer model are 4 audio samples: real composition of composer A, generated composition of composer A, cycled composition of composer A and finally the style transferred composition of composer A in the style of composer B. Cycled composition is the one which comes back after 2 hops of style transfer (A → B → A). Generator G_{AB} converts the input audio A1 to the style transferred version A3 which is in turn converted back to A4 by the generator G_{BA} which is the cycled version of real input audio A1.

The associated losses are described as follows:

Adversarial loss: Adversarial loss refers to the discriminator loss for predicting a generated composition as "real". The goal of the generator is to minimize this adversarial loss. The generator is trained/updated accordingly such that it generates realistic compositions in the target composer domain.

Identity loss: Identity loss refers to the loss resulting from regenerating an input composition in the style of the same composer by the generator of the same composer domain. Goal is to minimize this loss such that there is minimal discrepancy between the real audio and the generated audio.

Forward cycle loss: Forward cycle loss refers to the regeneration of the source audio after completing a cycle starting from generator AB producing audio in the style of composer B and generator BA reproducing the same song after converting the generated audio in the last hop to the source domain of composer A. This completes a cycle as shown in Fig. 10.

Backward cycle loss: Backward cycle loss refers to the regeneration of a target audio when completing a cycle starting from generator BA. The generator BA transfers the input audio of composer B in the style of composer A which in turn is re-transferred into the audio domain in the style of composer B. This is shown as in Fig. 11.

3.4. Full style transfer architecture

The style transfer model consists of 4 generators and 2 discriminators. The generators and discriminators play a zero sum game in which the generator always try to produce better and realistic compositions such that the discriminator is not able to discriminate between the real and the fake compositions. The hybrid composite model in Fig. 12 is comprised of 2 sub models each performing their unique responsibility. The detailed description of the composite models are shown in Fig. 13 and described below.

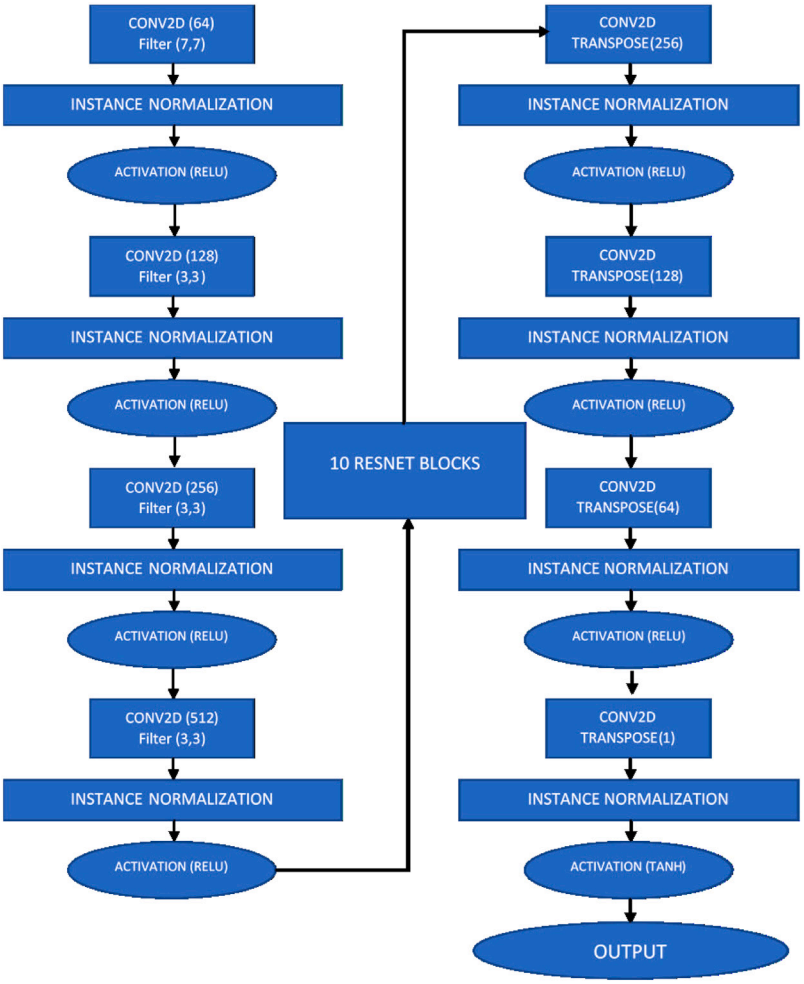

Fig. 6. Low level generator architecture as used in the proposed hybrid model.

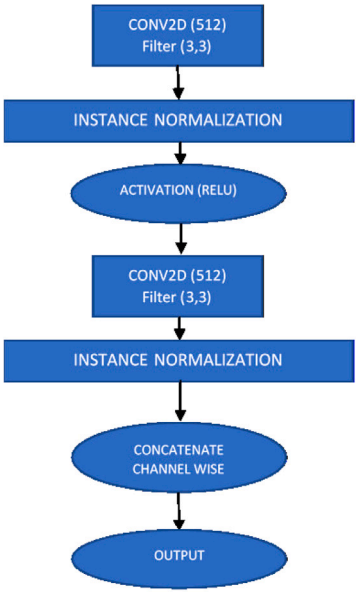

Fig. 7. Residual network architecture as used in the proposed hybrid model.

3.4.1. Single composite model

Input: Composer type of choice **(composer A/B)**

Output: Unique Compositions of the entered composer type **(audio melodies of composer A/B)**

The single composite model consists of 2 generators and 2 discriminators. The generators generate compositions from noise as input which is then sent to the respective discriminator to identify as real/fake. For example, the single composite models consist of generators G_A, G_B and discriminators D_A, D_B.

Noise → GA → FakeA → DA [Real/Fake]

Noise → GB → FakeB → DB [Real/Fake]

In single composite model, only adversarial loss is considered during training of the model.

3.4.2. Style transfer composite model

Input:

1. Real audio of a composer **(source audio: composer A)**
2. Composer label of preferred composer style **(target style: composer B)**

Output:

1. Unique compositions of composers A and B
2. Style transferred composition of the entered composer type (audio melodies of composer A in the style of composer B) that is, Source audio A → audio A in the style of B

One can note here that during the training phase of the proposed hybrid style transfer composite model, the training is done for both

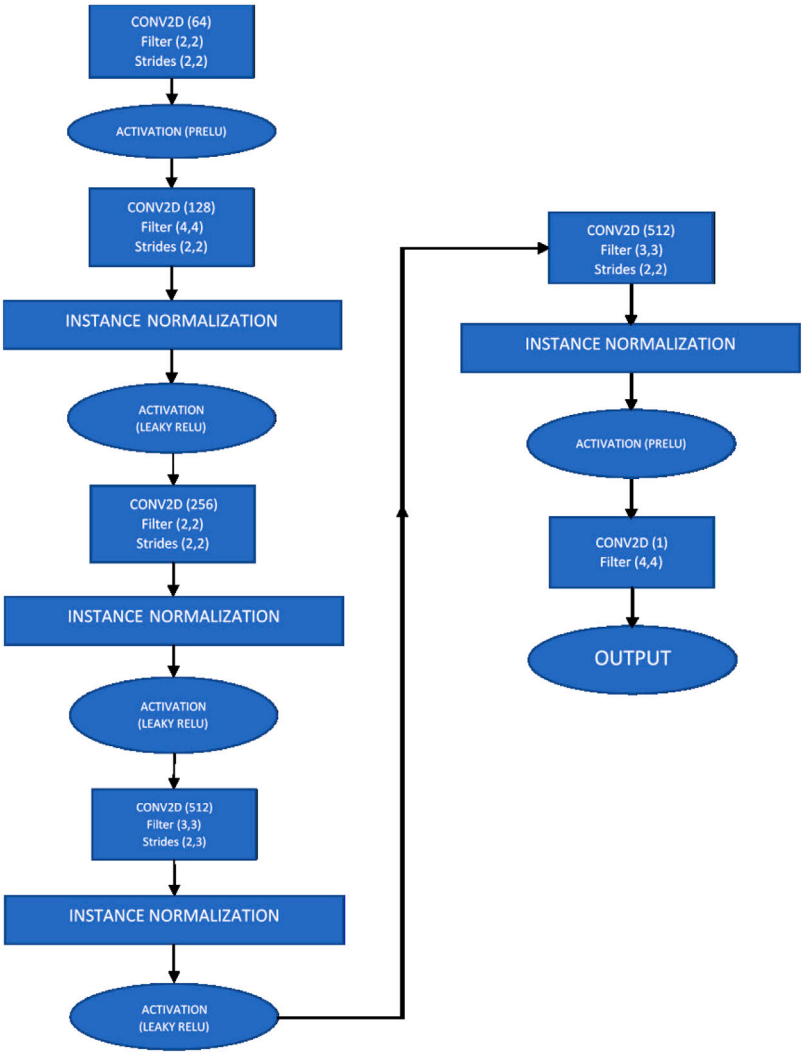

Fig. 8. Low level discriminator architecture as used in the proposed hybrid model.

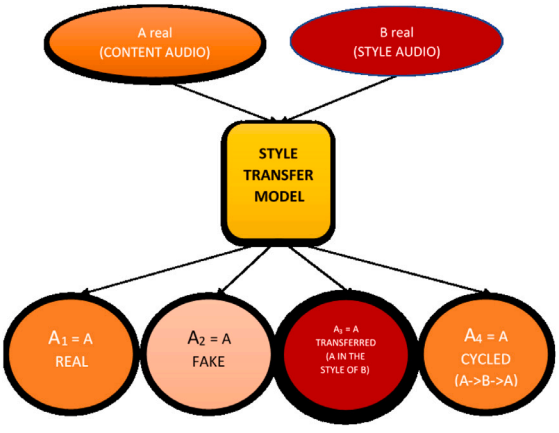

Fig. 9. Style transfer main components diagrammatic representation.

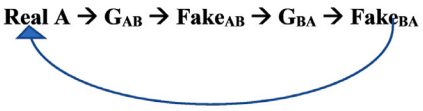

Fig. 10. Forward cycle loss.

Fig. 11. Backward cycle loss.

the composers A and B with real audio of both the composers in a cyclic way as shown in Fig. 12. This ensures that paired audio samples for both the composer domains are not required and that the model is capable of transferring style from A→B or B→A both. Output can thus also include style transfer to and from A and B as per requirement.

The style transfer composite model consists of 4 generators and 2 discriminators forming a cycleGAN. This model is a super set of the single composite model with addition 2 generators. In this model, the responsibilities of the generators are different. While one pair of generators (which is a part of single composite model) is responsible for unique composition generation of the composer of choice from noise, the other pair of generators have the responsibility to transfer the style of one composition to the style of another composer of choice. The discriminator pair is constant as present in the single composite model. The discriminators are trained on the respective composer classes based on the real data of the classes. For example, the style

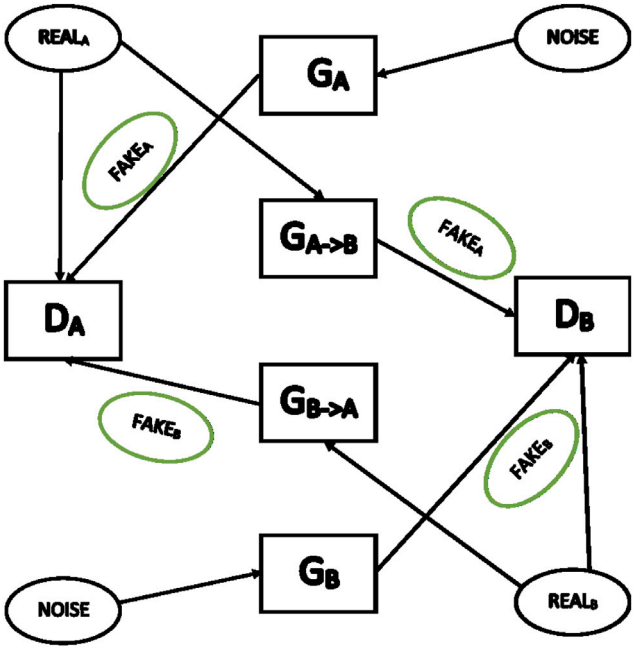

Fig. 12. Complete style transfer model architecture.

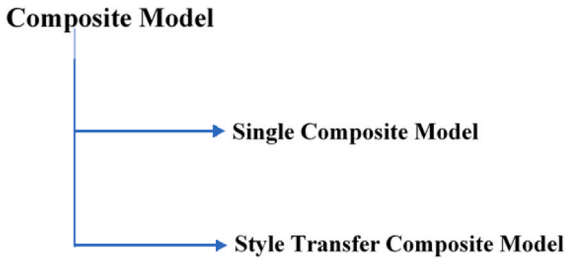

Fig. 13. Composite Model.

transfer composite model consists of the generators G_A, G_B, G_{AB} and G_{BA}. The discriminators are D_A and D_B respectively.

Noise → G_A → $Fake_A$ → D_A [Real/Fake]
Noise → G_B → $Fake_B$ → D_B [Real/Fake]
Real A → G_{AB} → $Fake_{AB}$ → D_B [Real/Fake]
Real B → G_{BA} → $Fake_{BA}$ → D_A [Real/Fake]

The output is $Fake_A$, $Fake_B$, $Fake_{AB}$ and $Fake_{BA}$ provided compositions of both composers A and B need to be converted to each other.

Associated losses with style transfer cycle composite model are:

1. Adversarial loss: A → [real/fake], B → [real/fake]
2. Identity loss: A → A, B → B
3. Forward loss: A → B → A / B → A → B
4. Backward loss: B → A → B / A → B → A

The 3 steps of this paper are thus intertwined as follows:

Step 1: Prepare the composer classification models by proper training and testing.
Step 2: Generate composer specific melody using a single composite GAN model. Test the performance of the generated melody using the composer classification models prepared in step 1.
Step 3: Generate Style transferred version of melody from composer A → composer B by combining the single composite model into the style transfer composite model as described earlier. Finally, test the generated style transferred melodies for their achieved accuracies using the pre-trained composer classification models.

3.5. Feature extraction

The feature set used in this work is discussed below along with diagrams which is then used for the training of the classification models in step 1.

The following Table 1 shows the 28 features used in feature set 1. This feature set now consists of 1000 frames per song which will later be flattened to 1 frame per song using simple mean.

The description of each of the features used in this set is given as follows:

1. **Zero Crossing Rate:** Zero crossing rate is the rate at which the audio signal changes sign from positive to negative and vice-versa. Zero crossing rate is used heavily in various domains like music information retrieval and speech recognition. It is defined in Eqs. (1), (2) and (3) (Bachu, Kopparthi, Adapa, & Barkana, 2010).

$$\sum_{n=-\infty}^{\infty} |sgn[x(m)] - sgn[x(m-1)]|w(n-m) \tag{1}$$

where, x(n) = nth sample of the signal x,

$$sgn[x(n)] = \begin{cases} 1, x(n) \geq 0 \\ -1, x(n) < 0 \end{cases} \tag{2}$$

and

$$w(n) = \begin{cases} \frac{1}{2N}, for 0 \leq n \leq N-1 \\ 0, otherwise \end{cases} \tag{3}$$

where w(n) = rectangular window for the nth sample, N = window size
Figs. 14 and 15 represent the zero crossing rates of an audio as shown below. In both Figs. 14 and 15, the y-axis represents [-abs(audio_time_series), abs(audio_time_series)] which is unitless.
2. **Spectral Centroid:** Spectral centroid indicates the "center of mass" for the audio signal. It is calculated by the weighted mean of the frequencies in the audio signal. It indicates where the audio signal is mostly centered (Giannakopoulos, 2009). Fig. 16 shows the spectral centroid of an audio. In Fig. 16, the y-axis represents [-abs(audio_time_series), abs(audio_time_series)] which is unitless.
3. **Spectral Rolloff:** Spectral rolloff indicates the frequency below which the majority of the frequencies of the audio signal lies. Fig. 17 shows the spectral rolloff for a particular song. Rolloff is given by the frequency Rt below which majority (85%) of the magnitude distribution lies as given in Eq. (4). In Fig. 17, the y-axis represents [-abs(audio_time_series), abs(audio_time_series)] which is unitless.

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n] \tag{4}$$

where, M[n] = FFT of the frequency bin n and N = number of frequency bins (Bartsch & Wakefield, 2005; Zheng et al., 2001). As seen in Fig. 17 when time is 2 : 30 the normalized spectral rolloff is 0.2662037037037037 Hz.
4. **Mel-Frequency Cepstral Coefficient (MFCC):** MFCCs (Lederle & Wilhelm, 2018) represent the small set of frequencies (usually 10–20) which describe the overall shape of the spectral envelope. It models the characteristics of human voice. Fig. 18 shows the MFCCs of an audio song. The shape of the audio song can be identified by taking a look at the color bar describing the shape of the spectral envelop in terms of the MFCC coefficients.
5. **Chroma Frequencies:** Chroma frequencies project the entire spectrum of the audio signal into 12 bins representing the 12 distinct semitones (chroma) of the musical octave ($C, C\#, D, D\#, E, F, F\#, G, G\#, A, A\#, B$) (Wakefield, 1999). Fig. 19 shows the

Table 1
Feature set consisting of 28 features for classification.

Features	Description
1 Zero Crossing	Rate of change of sign from positive to negative.
13 MFCCs	Small set of frequencies defining the overall shape of the spectral envelope.
12 Chromagrams	Representation of the 12 semitones (chroma) of the musical octave. (C, C#, D, D#, E, F, F#, G, G#, A, A#, B)
1 Spectral Rolloff	The frequency below which majority of the frequencies lie.
1 Spectral Centroid	Center of mass for the audio signal.

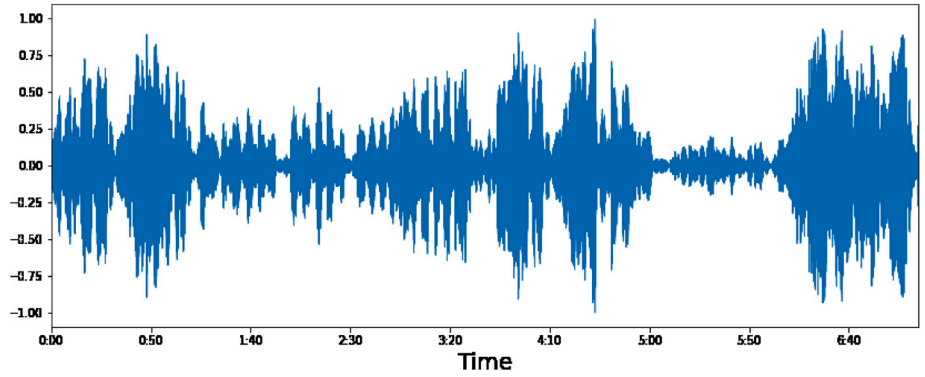

Fig. 14. Zero Crossing Rate.

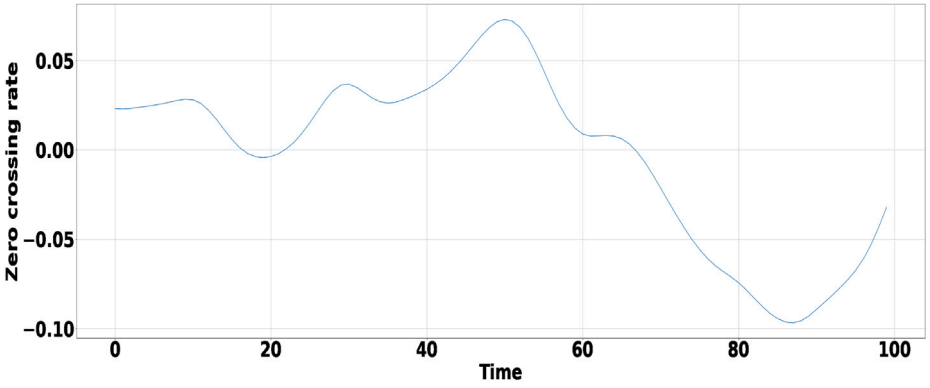

Fig. 15. Zero Crossing Rate Zoomed in.

chroma frequencies for an audio song. The colorbar shows the strength of the amplitudes in the different chroma frequencies at any given point in time. Although the *y*-axis shows labels for only the major notes like C, D, E, F, G, A and B, the other notes lie just after them such as C# is the note just after C in the *y*-axis of the image. For example it can be seen that certain chroma frequencies like F#, G# are having relatively stronger amplitudes than the rest of the chroma frequencies like C, C#, D, D#, E, F, G, A, A# and B.

4. Evaluation and results

4.1. Dataset

The dataset used here consists of 3 composers named Liszt, Chopin and Schubert from the Maestro dataset (Hawthorne et al., 2019). The MIDI files from Maestro v2 dataset is taken, converted to wav format, preprocessed and features are extracted from them. Total of 300 songs with 100 songs taken from each of the 3 composers are used for the feature extraction. The extracted features are then used to train the classifiers in Step 1. After performing the style transfer among composer styles in Steps 2 & 3, the generated MIDI compositions in the style of target composers are converted to wav format. The generated compositions are then evaluated using the pre-trained classifiers of Step 1 after extracting the features from the wav files of the generated style transferred compositions. The conversions involved in each step are as follows:

Step 1: MIDI to WAV format
Steps 2 & 3: MIDI to Piano rolls
Evaluation: Generated style transferred MIDI to WAV

The main reason for converting MIDI to WAV and vice-versa is to leverage the well known libraries for extracting the features from the audio data.

4.2. Steps of preprocessing the data

Before the data can be used to train the vanilla GAN and subsequently the hybrid style transfer model, the data must be preprocessed in the appropriate format.

Step 1: Audio data in the MIDI format is taken from the Maestro dataset (Hawthorne et al., 2019) and features are extracted from them.

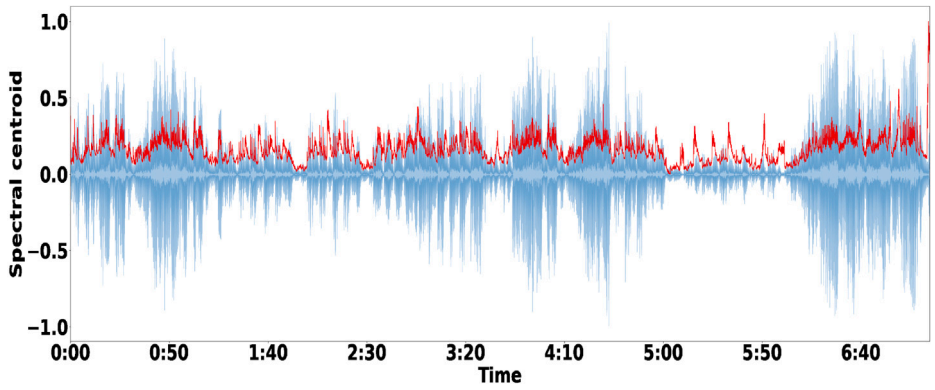

Fig. 16. Spectral centroid of the audio signal capturing its center of mass.

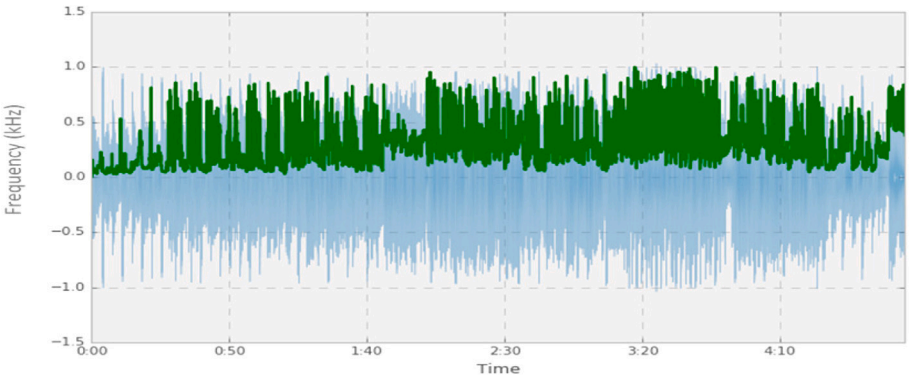

Fig. 17. Spectral Roll-off indicating the frequency below which majority of the frequencies of the audio signal lie.

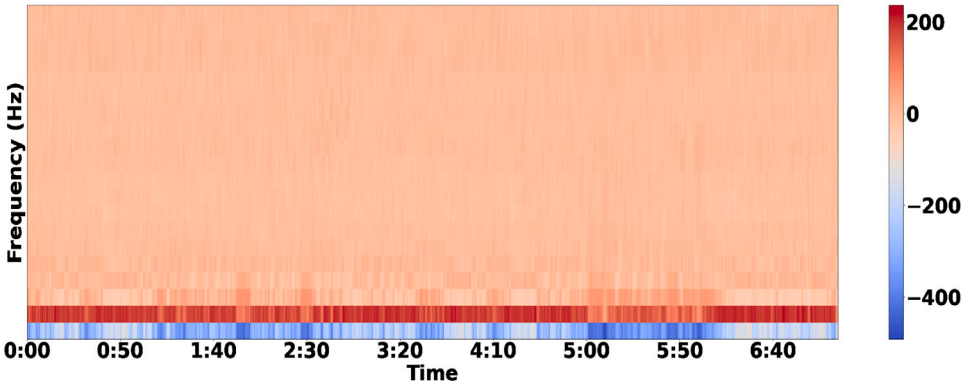

Fig. 18. MFCC showing the overall shape of the spectral envelope of the audio.

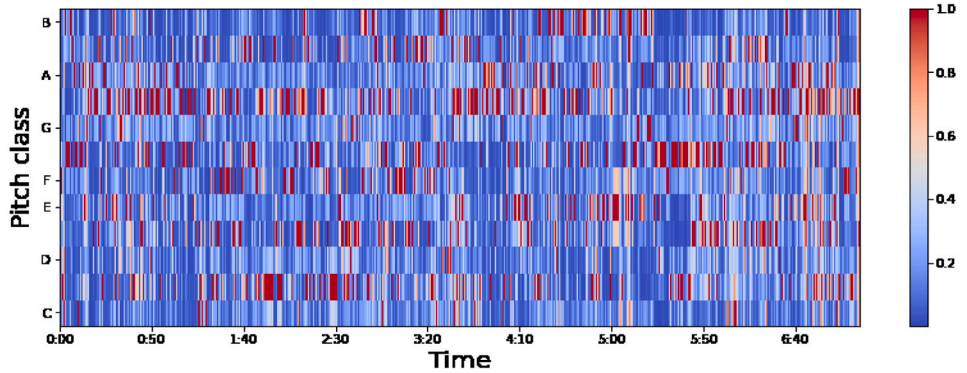

Fig. 19. Chroma Frequencies projecting the entire audio signal spectrum into 12 distinct chroma of the musical octave.

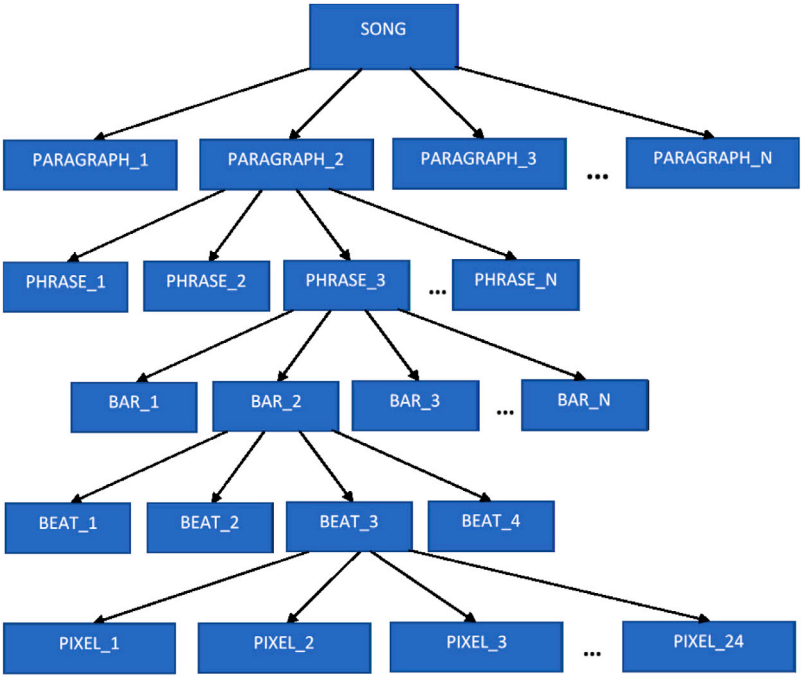

Fig. 20. Structure of a song.

1000 frames with 50% overlap are taken from which the 28 features are extracted from each of the audio file. The frames are then flattened using mean for each of the audio data.

Step2: Convert MIDI to piano rolls. Libraries used: pypianoroll, *pretty_midi* Steps for converting MIDI to piano rolls:

1. Load the MIDI file using the *pretty_midi* library
2. Create a multitrack object and parse the loaded MIDI file
3. Merge the piano rolls using 'sum' mode in which the merged piano roll contains the sum of all the piano rolls.
4. Ensure the times_steps are multiples of 64 as discussed above.
5. Clip the piano rolls to contain pitch range (24 to 108), that is, having 84 pitches in total (C1 to C8).
6. Reshape the final piano rolls to (−1, 64, 84, 1) and save it in .npy format.

These piano rolls are used in the next step for training the vanilla GAN and the hybrid style transfer GAN described in the next sections.

4.3. Structure of a song

This part gives the detailed structure of a song and how it is utilized in preprocessing the audio. The steps in preprocessing of the audio in order to feed to the style transfer network explained in detail as follows. These steps are necessary to produce the required composition structures which can lead to melodious generations. Understanding the structure of a song (Dong & Yang, 2018) will help in explaining the other components of the model like significance of the input shape, minimum threshold time, start time and end time of a pixel, duration of a phrase, etc. which are described below. The structure of a song in shown in the Fig. 20.

Each song can be thought of having a hierarchical structure as shown in Fig. 20. Starting with song as the root, there are multiple paragraphs within. Each paragraph consists of a number of phrases and each phrase in turn consists of multiple bars. In a 4/4 time signature, each bar consists of 4 beats and each beat can play a quarter note (1/4, 2/8 or 4/16 note). Each beat in MIDI can have a resolution of 24 to 960 pixels. Higher the resolution of a beat, more is the precision. Here, a beat resolution of 24 has been considered. For determining the steps which are necessary to be taken for the proper processing of the input audio data, reference has been taken from Brunner et al. (2018b).

Input shape: (64,84,1) = ($time\_steps, pitch\_range, output\_channel$)

Reason behind the input shape are described below in brief.

MIDI to piano rolls: We have followed the same methodology for converting MIDI to piano rolls as in Brunner et al. (2018b). Goal is to construct a t*p matrix where t=time steps and p=pitches. Using a sampling rate of 16 time steps per bar (Brunner et al., 2018b) which means keeping 16th note as the smallest note and considering 4 consecutive bars, we get (16*4) = 64 as the shape[0]. We extract only 84 pitches (C1 to C8) and discard the rest (Brunner et al., 2018b). The pitches form the 84 as shape[1].

Libraries used: pypianoroll (Dong et al., 2018), *pretty_midi* (Raffel & Ellis, 2014)

Steps for converting MIDI to piano rolls:

1. Load the MIDI file using the *pretty_midi* library
2. Create a multitrack object and parse the loaded MIDI file
3. Merge the piano rolls using 'sum' mode in which the merged piano roll contains the sum of all the piano rolls.
4. Ensure the *times_steps* are multiples of 64 as discussed above.
5. Clip the piano rolls to contain pitch range (24 to 108), that is, having 84 pitches in total (C1 to C8).
6. Reshape the final piano rolls to (−1, 64, 84, 1) and save it in .npy format.

The GAN generates piano rolls as the output. The next step is to convert the generated piano rolls into instrument.

The steps for setting piano rolls to instrument:

1. Calculate the beat resolution time.
2. Calculate minimum note duration threshold
3. Calculate maximum phrase end time
4. Pad piano rolls with zeros on both ends resulting in (n+2, 128) shape. 128 signifies the MIDI notes which can have a range of 0 to 127.
5. For each note (0 to 127):
    a. Calculate the start time

b. Calculate the end time
c. Calculate the duration of each note

6. Calculate each MIDI note by *pretty_midi* library.
7. Append notes to instrument object.
8. Sort the notes of instrument by their start time.

Calculations for Steps 2, 3, 5a, 5b, 5c are given below. MIDI pulses per quarter (ppq): In general, for MIDI, 120 is the beats per minute (BPM) which is the tempo. That is, 120 BPM = 120 quarter notes/min. Since 1 beat represents a quarter note, 120 beats in 60 s $\implies$ 1 beat takes 0.5 s. If beat resolution is 24, time_per_pixel = time_for_1_beat/beat_resolution = (0.5/24) = 0.02 s. Threshold = time_for_1_beat /notes per beat = (0.5/(1/4)) = 2. One phrase consists of n number of notes. Phrase end time = Number of notes * threshold = (n * 2) s. Start_time = start_idx * time_per_pixel, End_time = end_idx * time_per_pixel, Duration = end_time – start_time.

4.4. Data types used

The types of data involved in this work are described as follows. MIDI is the source and target format of both the source audio and the generated style transferred compositions. However, for training the hybrid style transferred GANs, the data type used is piano rolls. Piano rolls capture the time signature representation of the audio segments thereby creating a mathematical representation of a composition.

Step 1: MIDI to WAV format
Steps 2 & 3: MIDI to Piano rolls
Evaluation: Generated style transferred MIDI to WAV

The main reason for converting MIDI to WAV and vice-versa is to leverage the well known libraries for extracting the features from the audio data.

4.4.1. MIDI audio data

MIDI stands for Musical Instrument Digital Interface. MIDI is a structured symbolic form of representing musical data. MIDI data can only capture the instrumental information of musical data. As a result, MIDI format is most suitable for structured music composition generation by GANs as described in the later sections. MIDI has the following important segments:

1. NOTE ON/OFF: Indicates the beginning and the ending of a note being played.
2. NOTE PLAYED: Indicates the pitch information of the note being played.
3. VELOCITY: The loudness with which a note is played.
4. BPM (TEMPO): It indicates the beats/minute. The default MIDI tempo is 120 BPM and the default time signature is 4/4. This means a time segment can have 4 beats with each beat having 1/4 note or 2/8 note or 4/16 note played.
5. AFTERTOUCH: Pressure information after a key is pressed.

4.4.2. Piano rolls

Piano rolls represent the mathematical matrix representation of the structured audio segment. Piano rolls capture the time-pitch representation for each bar. A bar is segment of time consisting of a number of beats. Popular 4/4 time signature signifies 1 bar, contains 4 beats and each beat can contain 1/4 note, 2/8 note or 4/16 note (Brunner et al., 2018b). This requires each bar to be represented by a matrix. Each bar is thus represented by t time steps and p pitches. In this paper,$t = 16$ and p = 84. Time steps of 16 is a choice which is also the choice by many other works like in Brunner et al. (2018a) and Yang et al. (2017). The number of pitches p is kept to be between C1 to C8 octaves discarding lower and upper pitches. C1 to C8 contains 84 pitches. This is because each octave contains 12 pitches as follows:

$C, C\#, D, D\#, E, F, F\#, G, G\#, A, A\#, B$

Table 2
Diagrammatic representation of piano rolls.

	P1	P2	P3	P4	...	P81	P82	P83	P84
T1	0	1	1	1	...	0	1	0	1
T2									
T3									
T4	1	0	0	1	...	1	1	0	1
T5									
...									
T12									
T13	1	1	0	1	...	0	1	0	1
T14									
T15									
T16	1	0	1	1	...	1	0	0	1

Thus, there are in total (12*7) = 84 pitches.

The piano roll can be shown as follows in Table 2 which represents 1 bar:

The highlighted row in Table 2 shows the number of pitches played per time step. Moreover, here the velocity has been fixed to 100 so that each pitch can have the same loudness and can be represented as on/off only and not variable velocity values (velocity values can range from 0 to 127). “Note on” indicates the beginning of a note and “Note off” indicates the ending of that note. Since a single time step can have multiple notes being played simultaneously, we can thus represent the polyphonic music as a matrix representation without information loss. Similar to Dong and Yang (2018), 4 bar per phrase has been considered resulting in 64 x 84 piano roll matrix.

4.5. Algorithm configurations

The configurations used for the training of the style transfer model architecture are:

1. Number of training samples: 100 songs per composer (300 songs in total) each of Liszt, Schubert and Chopin
2. Number of epochs: 300
3. Batch size: 4

4.6. Network configuration

The proposed hybrid model comprises of primarily the generator and discriminator whose configurations are discussed in this section. There are 4 Conv2D layers in the generator with 64, 128, 256 and 512 filters each of dimension (3,3) followed by 10 residual network blocks followed by 4 more Conv2D transpose layers with 256, 128, 64 and 1 filters. Instance normalization has been used everywhere and activations used are mostly ReLU except the last activation used which is Tanh. The discriminator comprises of 4 Conv2D layers with 64, 128, 256 and 512 filters followed by the second last output layer with 512 filters and the output Conv2D layer with 1 filter. In this case also instance normalization has been used. Weight initialization was been done as random normal with standard deviation of 0.02 and learning rate of 0.002 has been used during training of the model.

Table 3
Results of training the classifiers in Step 1 on basic models.

Methods used	KFold CV	Stratified KFold CV	Leave One Out CV	Test	Cohen-kappa score
Decision tree	61.67%	49.17%	59.17%	58.33%	0.3949
KNN	67.50%	65.83%	70.42%	76.67%	0.6526
GaussianNB	52.08%	48.33%	52.50%	55%	0.3238
Linear SVM	57.92%	57.50%	59.58%	68.33%	0.5238
Radial SVM	66.67%	61.67%	67.50%	80%	0.70
Random forest	69.17%	67.50%	67.92%	73.33%	0.6006

Table 4
Results of training the classifiers in Step 1 on ensemble models.

Methods used	KFold CV	Stratified KFold CV	Leave One Out CV	Test	Cohen-kappa score
Bagging decision tree	49.58%	50.42%	46.67%	46.67%	0.2370
Bagging KNN	69.58%	64.58%	73.75%	80%	0.7240
Bagging linear SVM	58.33%	54.58%	57.50%	65%	0.3775
Bagging radial SVM	60%	62.08%	64.58%	71.67%	0.6271
Bagging GaussianNB	55.42%	51.67%	52.92%	48.33%	0.3702
Bagging Random forest	66.25%	61.67%	65.83%	71.67%	0.5273
Boosting Decision tree	50.83%	41.25%	50.42%	46.67%	0.2163
Boosting GaussianNB	53.33%	52.50%	56.67%	56.67%	0.3532
Boosting Random forest	64.58%	70.42%	66.25%	75%	0.5742
Stacking KNN + RF + GNB	70.42%	65%	73.33%	78.33%	0.6747
Stacking KNN + RF + radial SVM	70%	66.67%	71.25%	78.33%	0.6752
Stacking DT + GNB + linear SVM	59.58%	51.67%	57.50%	71.67%	0.6237

Table 5
Interpretation of Cohen's Kappa statistics score McHugh (2012).

Value of kappa	Level of agreement	Percentage of data that are reliable
0 – 0.20	None	0 – 4%
0.21 – 0.39	Minimal	4 – 15%
0.40 – 0.59	Weak	15 – 35%
0.60 – 0.79	Moderate	35 – 63%
0.80 – 0.90	Strong	64 – 81%
Above 0.90	Almost perfect	82 – 100%

4.6.1. Description of algorithm

For each epoch, real training audio data is taken for composer A and B. From noise, fake audio data in the style of composer A and B are then generated after training the GAN. Following this, fake audio data after style transfer from composer style A to composer style B is generated. Similar style transfer is also done from composer style B to A. The cycled fake audio data for each of the composers A and B are also generated. Then after updating the fakes from audio pools, each composite model A, B, A to B and B to A are trained on the training batch. Discriminators A and B are also updated.

4.7. Results

The obtained results have been demonstrated and explained in detail as follows.

4.7.1. Classification of composers

The feature set X for each song is of dimension (N x F) where N is the number of frames/song and F = number of features. The mean for each song is thus given in Eq. (5).

$$Mean = \frac{\sum_{n=1}^{1000} X_n{}^f}{N} \forall f \in F \tag{5}$$

The mean thus creates a single vector per song of dimension (1 x F). The final feature set comprising of all songs (S) is of the dimension (S X F). Following are the results after training the models with the mean feature vector created above (see Table 5).

4.7.1.1 Basic models and Ensemble models

As seen from Tables 3 and 4 above, highest test accuracy of 80% is observed. Leave one out cross validation performs better than the other cross-validation techniques with highest accuracy of 73.75%. Among the basic models, KNN is seen to perform best among others with accuracy as high as 70.42% and least accuracy of 52.50% by the Gaussian NB model. Among the ensemble models, bagging KNN is seen to perform better than others attaining the highest accuracy of 73.75% and the least accuracy of 41.25% by boosting of DT models. Stacking is seen to perform almost similar to bagging KNN models with highest accuracy of 73.33% which is better than most other models mainly because it is an ensemble of heterogeneous basic models. This helps it to leverage the effectiveness of multiple basic models together which makes it powerful and a strong learner in this case. Since here, the features extracted are also simply the mean, complex models are observed to perform better.

The Cohen's Kappa statistics are very important to test the interrater reliability McHugh (2012). As seen from Tables 3 and 4, the kappa statistics for the highest accuracy models are moderate indicating high inter rater reliability (see Table 5). This not only boosts the confidence of the achieved accuracy scores but also explains the reliability of the models.

4.7.2. Towards explainable AI: Shapley values

In this work, the Shapley values Kuhn and Tucker (1953) are presented in order to make our classification models more explainable. This would help in explaining which feature had more impact in the prediction of composer classes for each basic model. Table 6 shows the mapping of each feature with its description. Table 7 shows a mapping of composer classes with the composers as taken during the entire experimentation.

An overview of the importance of features for each basic model has been portrayed in this section (see Supplementary Figures S1 to S6). From the figures, it is observed that Feature 8 which is one of the chromagram feature is having a commendable impact on the classification of all 3 composer classes for all the basic models. For the Decision Tree model, feature 8 has highest impact for determining the composer classes Liszt and Schubert (see Supplementary Figure S1). However, feature 3 stands out of the rest in terms of the highest impact it has on classifying the composer class Chopin. For KNN model on the other hand, feature 8 is seen to have the highest impact value for determining the classes of composers Liszt and Chopin (see Supplementary Figure S2). For classifying composer Schubert using the KNN model, feature

Table 6
Feature set map for classification.

Features	Description
Feature 0 to 11	Chromagrams
Feature 12 to 24	MFCC
Feature 25	Spectral centroid
Feature 26	Spectral rolloff
Feature 27	Zero crossing rate

Table 7
Class map for classification.

Class	Description
Class 0	Liszt
Class 1	Chopin
Class 2	Schubert

6 is seen to have the highest impact (see Supplementary Figure S2). Supplementary Figure S3 shows the feature importance using Gaussian Naive Bayes. Feature 8, feature 25 and feature 6 seem to dominate the rest for classifying composers Liszt, Chopin and Schubert respectively using Gaussian Naive Bayes. Using Random Forest classifier, feature 8 seems to have highest impact for the classification of composer classes Liszt and Schubert as shown in Supplementary Figure S4. For classifying composer Chopin however, feature 25 leads the way using Random Forest as the classifier. Classifiers SVM with linear and radial kernels show similar feature importance in terms of classifying composers Liszt and Schubert as shown in Supplementary Figures S5 and S6 with feature 8 as having the highest impact. The only difference is seen during classification of composer Chopin where feature 4 and 6 seem to have the highest impact value using linear SVM and radial SVM respectively.

4.7.2.1 Composer class wise drill down

In the composer level Shapley values representation, Supplementary Figures S7 to S24 shows the impact and importance of each feature on the classification task for each composer. As it is seen, feature 8 is having consistently high impact in determining the composer classes.

For composer Liszt, Supplementary Figures S7 to S12 shows the comparison among all the basic models used for the task of classification. Feature 8 seem to consistently dominate the rest of the features in terms of impact value and importance as seen in Supplementary Figures S7, S8, S9, S10, S11 and S12.

For composer Chopin however, Supplementary Figures S13 to S18 does not show any consistently high impact feature. Feature 19 in Supplementary Figure S13 is seen to have the highest impact for Decision Tree model. On the other hand, features 4, 1 and 16 is seen to have the highest impact on the model outputs for KNN, Gaussian Naive Bayes and Random Forest respectively as in Supplementary Figures S14, S15 and S16. Feature 4 seem to have the highest impact on the model output using SVM with linear and radial kernels as shown in Supplementary Figures S17 and S18.

For composer Schubert, Supplementary Figures S19 to S24 shows feature 6 as the consistent high impact feature for all the models except Decision Tree as seen in S20, S21, S22, S23 and S24. For Decision Tree model, feature 27 seem to dominate over the others as seen in Supplementary Figure S19.

4.7.3. Style transfer from the style of composer chopin to composer liszt

This section presents a comparison among the results obtained on training the proposed hybrid model and the comparison paper's model (Brunner et al., 2018b) keeping the following constant:

1. The dataset: The dataset of 3 composers (Liszt, Chopin and Schubert) taken from the Maestro dataset (Hawthorne et al., 2019) are used to train both the networks.
2. Number of epochs: 300 for both the cases
3. Other parameters such as learning rates and other training parameters are also kept constant for both the cases.

One can note that since there is no recent works on music composer specific composition generation and style transfer, comparison has been done strictly on the model architectures. Keeping everything else constant, this section demonstrates how the proposed hybrid model and compared model in Brunner et al. (2018b) performs on the dataset after evaluation using the classifiers trained in step 1.

4.7.4. Key differences between the 2 model architectures

The main differences between the proposed hybrid style transfer model and the comparison model are given in Table 8. Since here comparison is done architecture wise (due to unavailability of work on composer style transfer), Table 8 captures in detail the major differences between the 2 models in the next sections.

4.7.5. Composer A (Chopin) to composer B (Liszt)

Fig. 21 shows the variation of discriminator loss while training the proposed hybrid model. This clearly shows that as the number of epochs increase, the loss decreases. The graph includes 6 different losses as observed from 2 different discriminators dA, dB as trained on real, fake and noise music. In our hybrid model, we have used just 2 discriminators dA and dB. Discriminator dA is trained to recognize real/fake A type melodies while discriminator dB is trained for B type melodies.

Fig. 22 indicates the discriminator loss variation while training the comparison model. In the comparison paper, the discriminators are 4 in number which are trained as a part of the cycle GANs. The 2 discriminators are being trained by the real and fake data of the respective composers A and B and the other 2 discriminators are trained by data of the respective composer and mixed composers. As we can see in the plot in Fig. 22, there are in total 8 curves which are plotted. They represent the training losses when discriminator of each type is trained by real data and then by fake data. That is, discriminator A (real, fake), discriminator B (real, fake), discriminator A_{mixed} (real, fake) and discriminator B_{mixed} (real, fake). Here, A_{mixed} and B_{mixed} refers to the training of the respective discriminators using mixed data (from all the 3 composers). There is a pattern which can be noticed in Fig. 22. All real data trainings follow a parabolic curve while the fake data training curves follow zigzag pattern.

Figs. 23 and 24 represents the accuracies obtained for each of the discriminators A and B respectively throughout the training for the proposed hybrid model. In this case, there are only 2 discriminators which are shared for both the cases, with and without style transfer. This not only promotes parameter sharing but also makes the generations consistent and more robust. Each of these 2 figures contains 3 accuracy plots for real, fake and noise data discrimination. The real data is the data which is the ground truth and is used to train the discriminators to identify the real and the fake compositions. The fake data is the data which is generated by style transfer between composer A and composer B. The noise data is the data which is unique composer specific compositions that are generated without style transfer. For both the discriminator A and B, the accuracy for identifying real data is considerably high in the range of [0.95, 1.0] approximately in the final epochs as seen from the figure. Then comes the accuracy for classifying the noise data (unique composer specific compositions without style transfer) for which the range of accuracy is approximately between [0.7, 1.0] in the final epochs. Finally comes the fake style transferred data which is identified by the discriminators with an accuracy of approximately [0.6, 1.0] in the final epochs.

Figs. 25 and 26 showcases the discriminators accuracies as obtained by training the compared model. In this model, there are 2 generators and 4 discriminators which are trained to generate only style transferred compositions. There is no provision for generating composer specific compositions without style transfer here. Each of the

Table 8
Key Differences between the proposed model and the comparison paper model.

Metric	Proposed Hybrid Style Transfer Model	Comparison model
Generators	Paired Vanilla GAN (2 GANs) and Cycle GAN (2 GANs)	Cycle GAN (only 2 GANs)
Discriminators	Discriminator for composer A, Discriminator for composer B	Discriminator for composer A, Discriminator for composer B, Discriminator for composer A (mixed) and Discriminator for composer B (mixed)
Steps	Unique composer specific compositions (from noise) and Music composer style transferred compositions	Music genre style transfer
Input and Output data	Input can be either a target composer label in which case the output is a unique composition in the style of chosen composer. The input can also be both source audio A and a target composer label B in which case output is the style transferred composition (A to B) in the style of target composer B.	Input can be of one type only in which case the output is the audio of one genre A is to be converted to the style of genre B.

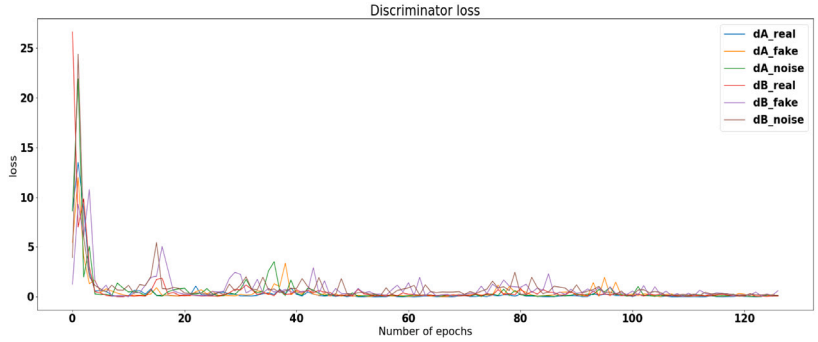

Fig. 21. Proposed hybrid style transfer model discriminator loss.

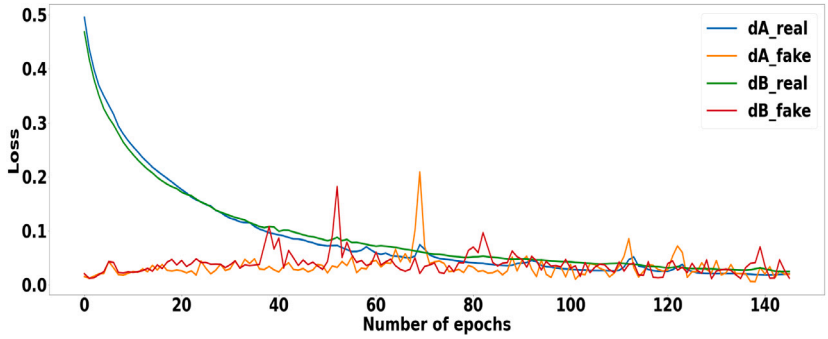

Fig. 22. Comparison model discriminator loss.

figures above contain 4 plots (real/fake) for each of the discriminators. Fig. 25 contains plots for 2 discriminators (discriminator A and discriminator A_{mixed}) while Fig. 26 contains plots for 2 discriminators (discriminator B and discriminator B_{mixed}). The mixed data indicates that the corresponding discriminators have been trained with mixed real data from all the 3 composers and fake generated data for that specific composer (say dBM is trained with both mixed real data and fake data for composer B). As we can see from the figures above, real data identification accuracies are high for all the discriminators in the range of [0.95, 1.0] approximately. Then comes the accuracies for identifying the style transferred fake data in the range of [0.8, 1.0] approximately. Finally, the least is the accuracy for identifying the composer specific fake data by the mixed discriminators in the range of [0.7, 1.0] approximately.

Figs. 27 and 28 represents the loss trajectories of the respective style transfer generators. There are 2 generators shown above (genAB and genBA) which refers to the respective style transfer generators responsible for style transfer from composer A ton the style of composer B. As seen in Fig. 27, which is the training loss for style generators in the proposed models, the trajectory represents a crooked parabola slowly converging to the optima. Similar is the case for the compared model generators as well (Fig. 28) with an extra peak just before the convergence.

Tables 9 and 10 tabulates the results obtained on evaluating the generated compositions from both the proposed hybrid style transfer model and the comparison model after style transfer from composer Chopin to the style of composer Liszt on the pre-trained classification models.

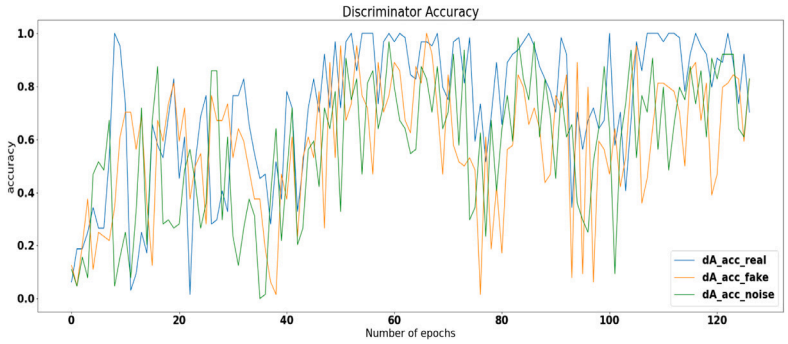

Fig. 23. Proposed hybrid style transfer model discriminator A accuracy.

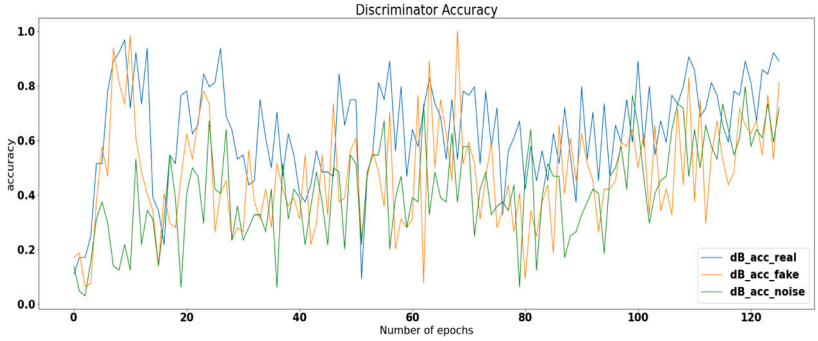

Fig. 24. Proposed hybrid style transfer model discriminator B accuracy.

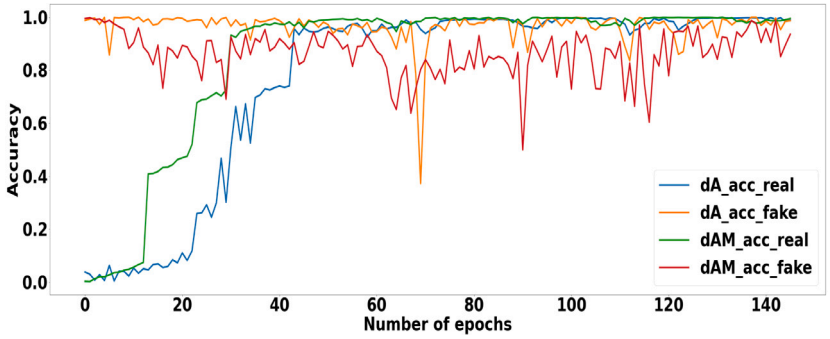

Fig. 25. Comparison model discriminator A accuracy.

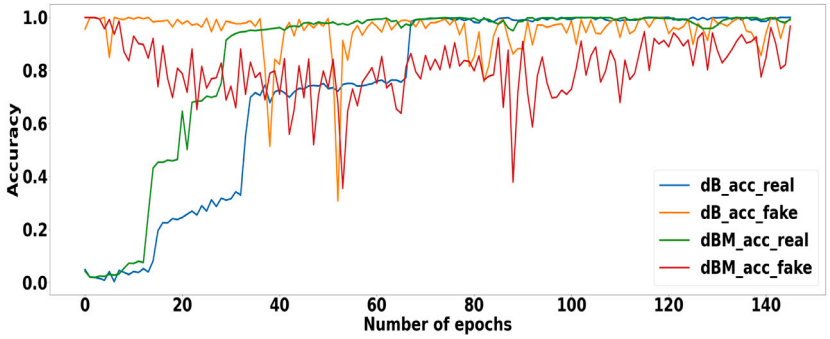

Fig. 26. Comparison model discriminator B accuracy.

Table 9 shows the performance of our basic ML classifiers when it is fed with style transferred compositions from composer Chopin to the style of composer Liszt from both the models (proposed model and comparison model). As seen from the table, style transferred compositions by the proposed hybrid model dominates the accuracies obtained with respect to the generations by the comparison model. With 77.27% accuracy, SVM with linear kernel overpower the other models in classifying the compositions from the hybrid model. On the other hand, the compositions from the comparison model are able to achieve only 56.67% accuracy by SVM_radial with the style transferred compositions they

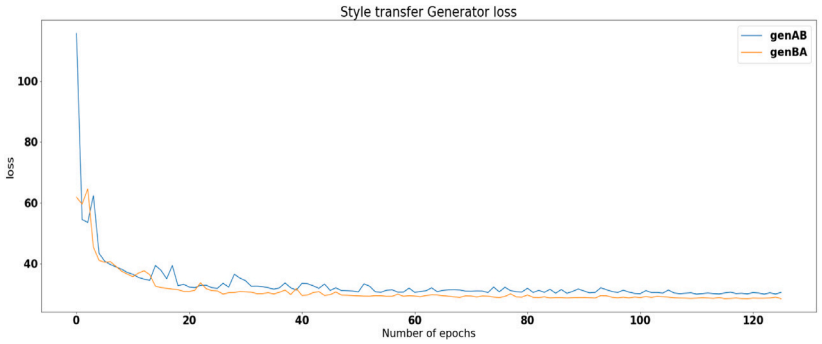

Fig. 27. Proposed hybrid style transfer model generator loss.

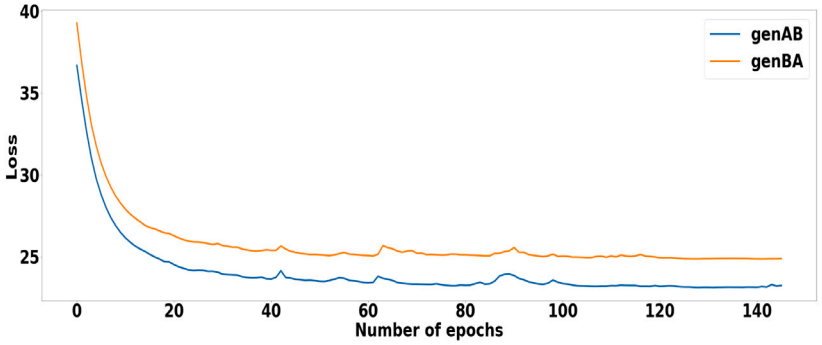

Fig. 28. Comparison model style transfer generator loss.

Table 9
Results of evaluating the generated style transferred compositions using the proposed hybrid style transfer model and that by the comparison model on the pre-trained basic classifiers of Step 1.

Models	Proposed hybrid model	Compared model
Decision tree	45.45%	43.33%
KNN	22.72%	66.67%
GaussianNB	18.18%	40%
Linear SVM	77.27%	43.33%
Radial SVM	35.36%	56.67%
Random forest	54.54%	50%

Table 10
Results of evaluating the generated style transferred compositions using the proposed hybrid style transfer model and that by the comparison model on the pre-trained ensemble classifiers of Step 1.

Models	Proposed hybrid model	Compared model
Bagging decision tree	59.09%	56.67%
Bagging KNN	13.63%	60%
Bagging linear SVM	77.27%	46.67%
Bagging radial SVM	31.81%	53.33%
Bagging GaussianNB	13.63%	46.67%
Bagging Random forest	40.90%	53.33%
Boosting Decision tree	40.90%	36.67%
Boosting GaussianNB	18.18%	36.67%
Boosting Random forest	54.54%	53.33%
Stacking KNN + RF + GNB	31.81%	56.67%
Stacking KNN + RF + radial SVM	36.36%	53.33%
Stacking DT + GNB + linear SVM	27.27%	43.33%

have generated. All the rest of the models perform with an average accuracy of 25%.

Table 10 tabulates the results from the ensemble models after predicting the labels for the input style transferred compositions for both the models. The accuracies for the respective bagging, boosting and stacking models are noted. Although the scenario remains the same in case of ensemble models as well, the proposed hybrid model is seen performing better than the comparison model, however the ensemble technique seems not to have a significant impact on the classifier performances. Highest accuracy for the hybrid model still remains at 77.27% and that for the comparison models at 56.67%.

In a nutshell, in both the cases, using basic ML models as well as ensemble techniques, style transferred compositions by the proposed hybrid model is seen to perform better in terms of classification accuracy than the comparison model when evaluated by the classifiers.

4.7.6. Subjective evaluation of music creativity

The creativity of the proposed hybrid model for composer style transfer of music is evaluated using the Consensual Assessment Technique (CAT) as done in De Prisco et al. (2020). In this work, CAT evaluates the quality of music in terms of creativity, technical quality, and expressiveness using domain experts. Fifteen experts from different regions of India are invited through emails to rate the quality of music. Each time, we asked experts to listen to a source music clip with a specific composer and style-transferred result (style of one of the other two composers) of the proposed hybrid model. The experts listen to the music clips and answer the seven questions (given in Table 11) included in the consensual assessment form on a 7-point Likert scale. Each expert evaluates ten (5 target clips from each composer) style transferred music clips.

Results of the creativity analysis are given in Table 12. The proposed hybrid model effectively transfers the style of pieces of music from source to target composer with mean ratings (mean of all criteria) of 4.04 or above. It is worth noting that domain experts are impressed with the technical quality (T1, T2 and T3) of the style transferred music clips. In particular, the melodic aspect (T3) of the style transferred music clips rated highly positively. One can observe that style-transferred music clips from composer Liszt to Schubert or from Schubert to Liszt received a high positive response from experts compared to other style transfers. Experts are also happy with the listenability (E2) of the compositions.

Table 11
Consensual Assessment Technique questions.

Criteria	Questions
Creativity	C1: Does the composition show original and imaginative musical ideas? C2: Does the composition show coherent and organized musical ideas?
Technical quality	T1: How successful is the composition after style transfer from A to B? T2: Do the harmonic elements of the style transferred version B show technical correctness concerning the style of composer B? T3: Do the melodic elements of the style transferred version B show technical correctness concerning the style of composer B?
Expressiveness	E1: Is the composition musically expressive? E2: How do you evaluate the listenability of the composition?

Table 12
The mean ratings of experts for each question to measure the creativity of the proposed hybrid model that transfers the style from source to target composer.

Source	Target	C1	C2	T1	T2	T3	E1	E2	Mean
Liszt	Chopin	3.76	3.94	4.48	4.17	4.58	3.94	4.01	4.12
	Schubert	3.94	4.00	4.48	4.10	4.56	4.05	4.06	4.17
Chopin	Liszt	3.76	3.84	4.42	4.05	4.60	3.88	3.89	4.06
	Schubert	3.58	3.70	4.60	3.98	4.72	3.81	3.93	4.04
Schubert	Liszt	3.76	3.89	4.60	4.30	4.72	4.14	4.24	4.23
	Chopin	3.53	3.72	4.52	4.17	4.69	3.85	4.01	4.07

C1 and C2: Questions correspond to creativity criteria.
T1, T2 and T3: Questions correspond to technical quality criteria.
E1 and E2: Questions correspond to expressiveness criteria.

Further, ratings of the experts submitted for each of the criteria given in Table 11 are analyzed for inter-expert reliability using Cronbach's alpha coefficient. It is observed that an alpha of 0.74 or higher is achieved in all cases and it represents good reliability.

5. Discussion, limitations and conclusions

5.1. Discussion

Technological advancement has always been beneficial in determining the advancement in everything that we do now in a more precise and useful way. Holding the hands of Machine Learning and Artificial Intelligence, today we are empowering the machines to perform activities similar to human beings. From healthcare, space research, and other scientific fields to the artistic fields, machines have started exploring and paving their paths towards being more human like. Although tremendous research efforts are being put towards improving the impact of machines in various scientific fields, relatively less research has been done towards bringing the impact of machines in the creative fields like music. Besides being an assistant to human beings in critical activities like surgeries, it would be an added advantage if the machines can think creatively like humans. Moreover, it would not only foster the creative side of machines but also help preserve and recreate some long lost musical compositional styles by music composers worldwide. Human talent is such a thing which cannot be imitated or even replaced. It is the unique ability of every individual that is lost with the individual and is only limited over their lifetime. Empowering machines in creating compositions would help remove the constraint of time thereby helping in creating unknown and style transferred compositions by composers of our choice on demand. Accordingly, this research is an attempt to dig deeper into the less treaded paths of the machines into the world of music. Overall, the research study of the 3 objectives was very exciting. In the first objective, various classification feature sets was performed. Highest composer classification accuracy obtained is 80%. However, the models seem to be under trained which can be improved if trained with more training data. Then comes the second objective in which given a composer of choice, the model will generate unique compositions in the style of the preferred composer. In this, no style transfer takes place but melody in the style of preferred composer is generated from noise. The third objective performs the actual style transfer from composer A to preferred composer B. In this objective, when given a melody composed by composer A, the model generates compositions in the style of the preferred composer B. The generated compositions obtained in objectives 2 and 3 are then evaluated to understand success of the models in generating compositions in the respective composer styles. The feature set comprising of 28 features are extracted for 1000 frames per song similar to objective 1. The frames are then flattened using mean in order to generate 1 frame per song. The final feature set is then evaluated as an unseen test set for the classification models, pre-trained in objective 1 using mean as the frame flattening technique. The models are expected to classify the compositions in the class of the target composer after style transfer and not the class of the original composer of the melody. The highest accuracy obtained is 77.27% on the unseen style transferred test set which indicates the degree of prediction accuracy. Training the style transfer model for a greater number of epochs (say 3000 or more) could be beneficial in improving the final prediction accuracy further thereby improving more accurate style transferred compositions. Here, the model is trained for only 200 epochs due to hardware constraints which is very less. Thus, overall, the research study was very interesting although it has some limitations as mentioned in the next section.

5.2. Limitations

The following are the limitations of the research study:

1. The style transfer architecture has been trained with only 100 songs from each composer. This is mainly due to limited computational resources. The model finally generates 20 compositions after training for 200 epochs which can be improved with training for more epochs.
2. Experimenting more with the hyper parameters can be beneficial in generating more accurate compositions.

3. More experimentation regarding the method of extracting useful lossless information from the frames of each song can help in improving the classifiers further in Step1. This can further help in better classification of the generated target composer style compositions.

5.3. Conclusions

Since time immemorial, technological boons have touched the lives of human beings in unprecedented ways. This research is an attempt to dig deeper into the less treaded paths of the machines into the world of music. Empowering machines in creating compositions would help remove the constraint of time thereby helping in creating unknown and style transferred compositions by composers of our choice on demand. Therefore, the proposed music compositional style transfer algorithm aims to systematically address the problem of style transfer in an effective manner. Starting with the training of the classifiers, to generating compositions from noise to finally combining them together with the addition of the style transfer model is an end to end approach to create as well as evaluate the success of style transfer. The highest classification accuracy obtained in Step 1 is 80% and 77.27% for the classification of the generated style transferred version of a composition into the target composer class using the pre-trained classifiers. The work has also been compared with another paper and the highest accuracy obtained using the style transfer model of the comparison paper being 56.67%. The dataset used here is the Maestro dataset (Hawthorne et al., 2019). Some of the future directions in this aspect can be to integrate vocals with melody in compositional style transfer, replacing the static components in music generation with a layer of deep neural network and training the classifiers with more data and observing the performance. Thus, empowering machines in creating compositions would help remove the constraint of time thereby helping in creating unknown and style transferred compositions by composers of our choice on demand.

CRediT authorship contribution statement

Sreetama Mukherjee: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Visualization. **Manjunath Mulimani:** Conceptualization, Methodology, Validation, Investigation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2021.116195.

References

Abdulatif, S., Armanious, K., Guirguis, K., Sajeev, J. T., & Yang, B. (2019). Aegan: Time-frequency speech denoising via generative adversarial networks. CoRR http://arxiv.org/abs/1910.12620.

Agarwala, N., Inoue, Y., & Sly, A. (2017). Music composition using recurrent neural networks. CoRR http://arxiv.org/abs/1412.3191.

Ananthabhotla, I., & Paradiso, J. A. (2017). Visualsoundtrack: An approach to style transfer in the context of soundtrack prototyping. In *International computer music conference, (ICMC)*.

Bachu, R., Kopparthi, S., Adapa, B., & Barkana, B. D. (2010). Voiced/unvoiced decision for speech signals based on zero-crossing rate and energy. In *Advanced techniques in computing sciences and software engineering* (pp. 279–282). Springer.

Bao, H., Huang, S., Wei, F., Cui, L., Wu, Y., Tan, C., et al. (2019). Neural melody composition from lyrics. In *Lecture notes in computer science: Vol. 11838, International conference on natural language processing and chinese computing (NLPC)* (pp. 499–511). Springer.

Bartsch, M. A., & Wakefield, G. H. (2005). Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, *7*(1), 96–104.

Brunner, G., Konrad, A., Wang, Y., & Wattenhofer, R. (2018a) MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer. In *International society for music information retrieval conference (ISMIR)* (pp. 747–754).

Brunner, G., Wang, Y., Wattenhofer, R., & Zhao, S. (2018b). Symbolic music genre transfer with cyclegan. In *International conference on tools with artificial intelligence (ICTAI)* (pp. 786–793). IEEE.

Chen, Y.-H., Wang, B., & Yang, Y.-H. (2019). Demonstration of performancenet: a convolutional neural network model for score-to-audio music generation. In *International joint conference on artificial intelligence (IJCAI)* (pp. 6506–6508).

Choksi, B., Sawant, A., & Mali, S. (2017). Style transfer for audio using convolutional neural networks. *International Journal of Computer Applications*, *175*, 17–20.

Chuan, C.-H. (2013). A multimodal approach to song-level style identification in pop/rock using similarity metrics. In *International conference on machine learning and applications (ICMLA)* (pp. 321–324). IEEE.

Colombo, F., & Gerstner, W. (2018). Bachprop: Learning to compose music in multiple styles. CoRR http://arxiv.org/abs/1802.05162.

Colombo, F., Muscinelli, S. P., Seeholzer, A., Brea, J., & Gerstner, W. (2016). Algorithmic composition of melodies with deep recurrent neural networks. CoRR http://arxiv.org/abs/1606.07251.

Dai, S., Zhang, Z., & Xia, G. G. (2018). Music style transfer: A position paper. arXiv preprint arXiv:1803.06841.

De Prisco, R., Malandrino, D., Zaccagnino, G., Zaccagnino, R., & Zizza, R. (2017). A kind of bio-inspired learning of music style. In *International conference on evolutionary and biologically inspired music and art* (pp. 97–113). Springer.

De Prisco, R., Zaccagnino, G., & Zaccagnino, R. (2020). Evocomposer: An evolutionary algorithm for 4-voice music compositions. *Evolutionary Computation*, *28*(3), 489–530.

Dong, H.-W., Hsiao, W.-Y., & Yang, Y.-H. (2018). Pypianoroll: Open source Python package for handling multitrack pianoroll. In *International society for music information retrieval conference*.

Dong, H.-W., & Yang, Y.-H. (2018). Convolutional generative adversarial networks with binary neurons for polyphonic music generation. In *International society for music information retrieval conference (ISMIR)* (pp. 190–196).

Engel, J., Agrawal, K. K., Chen, S., Gulrajani, I., Donahue, C., & Roberts, A. (2019) Gansynth: Adversarial neural audio synthesis. In *International conference on learning representations (ICLR)*.

Gatys, L. A., Ecker, A. S., & Bethge, M. (2016). Image style transfer using convolutional neural networks. In *International conference on computer vision and pattern recognition (CVPR)* (pp. 2414–2423).

Giannakopoulos, T. (2009). *Vol. 2, A method for silence removal and segmentation of speech signals, implemented in Matlab*. University of Athens, Athens.

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. In *International conference on computer vision and pattern recognition (CVPR)* (pp. 2414–2423). IEEE Computer Society.

Goulart, A. J. H., Maciel, C. D., Guido, R. C., Paulo, K. C. S., & da Silva, I. N. (2011). Music genre classification based on entropy and fractal lacunarity. In *International symposium on multimedia (ISM)* (pp. 533–536). IEEE Computer Society.

Hantrakul, L., Engel, J. H., Roberts, A., & Gu, C. (2019). Fast and flexible neural audio synthesis. In *International society for music information retrieval conference (ISMIR)* (pp. 524–530).

Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., et al. (2019). Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International conference on learning representations*.

Hung, Y.-N., Chiang, I., Chen, Y.-A., Yang, Y.-H., et al. (2019). Musical composition style transfer via disentangled timbre representations. In *International joint conferences on artificial intelligence (IJCAI)* (pp. 4697–4703).

Johnson, D. D. (2017). Generating polyphonic music using tied parallel networks. In *International conference on evolutionary and biologically inspired music and art* (pp. 128–143). Springer.

Kaneko, T., Takaki, S., Kameoka, H., & Yamagishi, J. (2017). Generative adversarial network-based postfilter for stft spectrograms. In *Annual conference of the international speech communication association (INTERSPEECH)* (pp. 3389–3393). ISCA.

Kuhn, H. W., & Tucker, A. W. (1953). *Vol. 2, Contributions to the theory of games*. Princeton University Press.

Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., Teoh, W. Z., Sotelo, J., et al. (2019). Melgan: Generative adversarial networks for conditional waveform synthesis. In *Annual conference on neural information processing systems* (pp. 14881–14892).

Le, P. N., Ambikairajah, E., Epps, J., Sethu, V., & Choi, E. H. (2011). Investigation of spectral centroid features for cognitive load classification. *Speech Communication*, *53*(4), 540–551.

Lederle, M., & Wilhelm, B. (2018). Combining high-level features of raw audio waves and mel-spectrograms for audio tagging. CoRR http://arxiv.org/abs/1811.10708.

Liu, H.-M., & Yang, Y.-H. (2018). Lead sheet generation and arrangement by conditional generative adversarial network. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)* (pp. 722–727). IEEE.

Lu, C.-Y., Xue, M.-X., Chang, C.-C., Lee, C.-R., & Su, L. (2019). Play as you like: Timbre-enhanced multi-modal music style transfer. In *International conference on artificial intelligence (AAAI)* (pp. 1061–1068). AAAI Press.

Luo, J., Yang, X., Ji, S., & Li, J. (2020). Mg-VAE: Deep Chinese folk songs generation with specific regional styles. In *International conference on sound and music technology (CSMT)* (pp. 93–106). Springer.

Mao, H. H., Shin, T., & Cottrell, G. (2018). Deepj: Style-specific music generation. In *International conference on semantic computing (ICSC)* (pp. 377–382). IEEE.

Marafioti, A., Perraudin, N., Holighaus, N., & Majdak, P. (2019) Adversarial generation of time-frequency features with application in audio synthesis. In *International conference on machine learning* (pp. 4352–4362).

McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochemia Medica*, *22*(3), 276–282.

Mogren, O. (2016). C-RNN-GAN: Continuous recurrent neural networks with adversarial training. CoRR http://arxiv.org/abs/1611.09904.

Nakamura, E., Shibata, K., Nishikimi, R., & Yoshii, K. (2019). Unsupervised melody style conversion. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 196–200). IEEE.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., et al. (2016). Wavenet: A generative model for raw audio. In *Speech synthesis workshop* (p. 125). ISCA.

Panteli, M., Bittner, R., Bello, J. P., & Dixon, S. (2017). Towards the characterization of singing styles in world music. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 636–640). IEEE.

Raffel, C., & Ellis, D. P. (2014). Intuitive analysis, creation and manipulation of midi data with pretty midi. In *International society for music information retrieval conference late breaking and demo papers* (pp. 84–93).

Wakefield, G. H. (1999). Mathematical representation of joint time-chroma distributions. *Vol. 3807*, In *Advanced signal processing algorithms, architectures, and implementations* (pp. 637–645). International Society for Optics and Photonics.

Wang, C.-i., & Tzanetakis, G. (2018). Singing style investigation by residual siamese convolutional neural networks. In *International conference on acoustics, speech and signal processing (ICASSP)* (pp. 116–120). IEEE.

Weiß, C., Brand, F., & Müller, M. (2019). Mid-level chord transition features for musical style analysis. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 341–345). IEEE.

Yang, L.-C., Chou, S.-Y., & Yang, Y.-H. (2017) MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. In *International society for music information retrieval conference (ISMIR)* (pp. 324–331).

Yu, X. (2020). Emerging applications of generative adversarial networks. *MS&E*, *740*(1), Article 012132.

Yu, L., Zhang, W., Wang, J., & Yu, Y. (2017). Seqgan: Sequence generative adversarial nets with policy gradient. In *International conference on artificial intelligence* (pp. 2852–2858). AAAI Press.

Zhang, L., Shu, C., Guo, J., Zhang, H., Xie, C., & Liu, Q. (2020). Generative adversarial network-based neural audio caption model for oral evaluation. *Electronics*, *9*(3), 424.

Zheng, F., Zhang, G., & Song, Z. (2001). Comparison of different implementations of MFCC. *Journal of Computer Science and Technology*, *16*(6), 582–589.

Zhou, J., Li, X., & Mitri, H. (2016). Classification of rockburst in underground projects: Comparison of ten supervised learning methods. *Journal of Computing in Civil Engineering*, *30*, Article 04016003. http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000553.

Zhou, J., Qiu, Y., Jahed Armaghani, D., Zhang, W., Li, C., Zhu, S., et al. (2020). Predicting TBM penetration rate in hard rock condition: A comparative study among six XGB-based metaheuristic techniques. *Geoscience Frontiers*, *12*, http://dx.doi.org/10.1016/j.gsf.2020.09.020.

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International conference on computer vision (ICCV)* (pp. 2242–2251). IEEE Computer Society.