UNSUPERVISED MUSICAL TIMBRE TRANSFER FOR NOTIFICATION SOUNDS

Jing Yang[1] *Tristan Cinquin*[1] *Gábor Sörös*[2]

[1]Department of Computer Science, ETH Zurich, Switzerland
[2]Nokia Bell Labs, Budapest, Hungary

ABSTRACT

We present a method to transform artificial notification sounds into various musical timbres. To tackle the issues of ambiguous timbre definition, the lack of paired notification-music sample sets, and the lack of sufficient training data of notifications, we adapt the problem for a cycle-consistent generative adversarial network and train it with unpaired samples from the source and the target domains. In addition, instead of training the network with notification sound samples, we train it with video game music samples that share similar timbral features. Through a number of experiments, we discuss the efficacy of the model in transferring the timbre of monophonic and even homophonic notifications while preserving their original melody envelopes. We envision notification timbre transfer as a way of less distracting information delivery, and we demonstrate example music pieces augmented with notifications after timbre transfer.

Index Terms— Audio style transfer, musical timbre transfer, sound notification, human-machine interface

1. INTRODUCTION

Audio style transfer techniques have been developed to change the style of human voice [1, 2] and music [3]. Regarding music, *style* could refer to instrument timbre [4, 5], music genre [6, 7], a global musical structure [8], etc. Typical audio style transfer techniques include convolutional neural network (CNN)-based models [3, 9, 10], (variational) autoencoders [11, 12, 13], generative adversarial network (GAN)-based models [4, 6, 14], etc. Among different types of music style transfer, timbre transfer has been a popular topic [4, 5, 12], and is also the focus of this work.

To the best of our knowledge, no related work has focused on notification sounds so far. We define notification sounds as the short musical sequences commonly used in digital smart assistant devices and popular applications to remind the user of new information arrival. Notification sounds usually have a short duration (around $1\,s-15\,s$). Many of them are monophonic with several notes or a simple melody, while some are homophonic musical pieces.

Our goal is to change the style of notification sounds. More precisely, we aim to change the musical timbre of notifications while preserving their original melody envelope. We envision a potential application of notification timbre transfer as a way of less distracting information delivery. Notifications have been shown as a source of stress and distraction that affects people's task performance [15], but disabling them is not a satisfying solution either, since this might make users unaware of their activity context [16]. Since digital music has been widespread, researchers have explored to deliver notifications in a less pronounced way by modifying the music that a user is listening to. For example, adding acoustic effects like reverb to the music [17], or slightly modifying the notes and rhythm of the music [18]. Timbre transfer might indicate a new perspective to this issue. While preserving the melody envelope still associates a notification with its interpretation, it might help to alleviate distraction by harmoniously embedding the notification into the music, after changing the notification into the timbre of the music.

We see two main challenges that differentiate the timbre transfer of notifications from general audio style transfer. First, unlike music played with specific instrument(s), it is difficult to define the timbre of notifications since they could be generated in several ways (real instruments, electronically synthesized, etc.). Among the commonly used notifications that we collected from iOS, Android, and popular applications like Skype, except those that either have a clear instrumental timbre or barely have a musical structure (e.g. frog croak), the majority tends to have a timbre of, or a timbre similar to, electronically synthesized tones. We focus on these notifications in this work. The second challenge is the insufficient amount of training data of such notifications in general, and a complete lack of pairs of notifications and corresponding music sequences, which largely limits the set of applicable conversion methods.

This paper contributes one of the first explorations on musical timbre transfer for artificial notification sounds. We address the above challenges by developing a timbre transfer model based on the cycle-consistent generative adversarial network (CycleGAN) [19], which was initially proposed for image style transfer, but can be trained with unpaired sample sets in general. We observed that a large portion of video game music (e.g. Super Mario) shares similar timbral features with notifications. We thus collected video game music to train the transfer model for notifications. The paper also presents an extensive discussion on the efficacy, potential, and limitations of our model, supported by quantitative and qualitative evaluations and a user perception study. We further present example music pieces to demonstrate our idea of less distracting information delivery with timbre-transformed notifications.

2. TIMBRE TRANSFER METHOD FOR NOTIFICATIONS

We developed our method based on the CycleGAN [19] structure, which has shown high-quality performance in image [19] and audio [4, 20] style transfer with unpaired training data. For training a style transfer model, raw audio data is typically represented using short-time Fourier transform (STFT) [3], Mel-spectrogram [14], constant-Q transform (CQT) [4], or a combination of these representations [10]. Considering the human auditory perception on a logarithmic frequency scale and the convenience of converting the spectrogram back to the waveform for playback, we used the Mel-spectrograms for training the model.

Fig. 1 shows our timbre transfer pipeline. The original notification sound wave is transformed into its Mel-spectrogram and input to a CycleGAN generator, which generates a new Mel-spectrogram in the target timbre. The output spectrogram is converted back to the time domain by applying e.g. the Griffin-Lim algorithm [21] or the gradient-based inversion algorithm [22], which can reconstruct the phase of the sound wave reasonably well. In the following, we introduce the design of the CycleGAN model and the mechanism to train the model for notifications of arbitrary length.

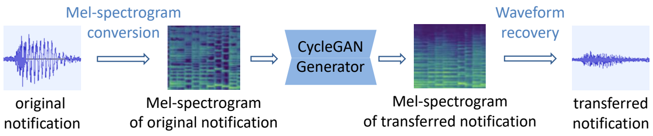
Mel-spectrogram conversion
original notification
Mel-spectrogram of original notification
CycleGAN Generator
Mel-spectrogram of transferred notification
Waveform recovery
transferred notification

Fig. 1. The pipeline of notification timbre transfer.

2.1. CycleGAN Model

With X being the source audio domain and Y being the target audio domain, a CycleGAN model consists of two generators $G : X \rightarrow Y$ and $F : Y \rightarrow X$, which in our case have a U-Net [23] architecture, and two discriminators D_X and D_Y, which in our case are convolutional PatchGAN discriminators [24]. These generators and discriminators were shown to be adequate for audio style transfer [14]. The generators include three downsampling and three upsampling layers with 2D convolutions with stride 2. Leaky ReLU activations are used in all 2D convolutions except in the last transposed 2D convolution layer that uses $\tanh$ activation. For both downsampling and upsampling phases, we set the receptive field to cover the entire frequency domain to capture the whole spectra of the input audio. The discriminators consist of three downsampling layers with leaky ReLU activation and a final dense layer. We apply spectral normalization to each convolutional filter of the generators and discriminators to improve the training stability as inspired by [25, 26].

2.2. Training Objectives

The training objective of the CycleGAN model is that G generates samples that D_Y cannot distinguish from real, while F generates samples that D_X cannot distinguish from real, such that any $XY\hat{X}$ and $YX\hat{Y}$ cycle gives a sample closest to the original. Formally:

$$\begin{aligned} L(G, F, D_X, D_Y) = L_{GAN}(G, D_Y) + L_{GAN}(F, D_X) + \\ \lambda_{cycle} L_{cycle}(G, F) + \lambda_{id} L_{id}(G, F) \end{aligned} \quad (1)$$

This objective consists of two GAN adversarial losses $L_{GAN}(G, D_Y)$ and $L_{GAN}(F, D_X)$, a cycle consistency loss $L_{cycle}(G, F)$, and an identity loss $L_{id}(G, F)$. The GAN adversarial losses $L_{GAN}(G, D) = L_{GAN,G}(G, D) + L_{GAN,D}(G, D)$ consist of the following generator and discriminator objectives:

$$L_{GAN,G}(G, D) = -\mathbb{E}_{x\sim p_{data}(x)}[D(G(x))] \quad (2)$$

$$\begin{aligned} L_{GAN,D}(G, D) = -\mathbb{E}_{x\sim p_{data}(x)}[min(0, -1 - D(G(x)))] \\ -\mathbb{E}_{y\sim p_{data}(y)}[min(0, -1 + D(y)] \end{aligned} \quad (3)$$

The cycle consistency loss is formulated as

$$\begin{aligned} L_{cycle}(G, F) = \mathbb{E}_{x\sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \\ \mathbb{E}_{y\sim p_{data}(y)}[\|G(F(y)) - y\|_1] \end{aligned} \quad (4)$$

and we included the identity loss

$$\begin{aligned} L_{id}(G, F) = \mathbb{E}_{x\sim p_{data}(x)}[\|G(x) - x\|_1] + \\ \mathbb{E}_{y\sim p_{data}(y)}[\|F(y) - y\|_1] \end{aligned} \quad (5)$$

By tuning the weights of the cycle consistency loss and the identity loss, we aim to keep a good balance between the melody preservation and timbre modification. We included two more techniques to improve the training stability. First, we updated the generator more often than the discriminator since the discriminator normally learns faster [27]. Our experiments showed that three generator updates per discriminator update worked well in our case. Second, we updated the discriminator using a random selection of 50 generated spectrograms from the history buffer, since using only the latest data might cause divergence of the adversarial training [28].

2.3. Training for Notifications of Arbitrary Lengths

In order to process notification sounds of arbitrary lengths, following [14], we implemented a splitting-and-concatenation mechanism when training and applying the CycleGAN model. The key idea is to train the generator to create a complete spectrogram from split-up chunks. Suppose that the spectrogram has a size of $F \times T$, where F is the number of Mel frequency channels and T is the time length that varies from sample to sample. During the training process, the spectrograms of training data are split into chunks with a determined width $L1 < T$ for all T in the source and the target domain. The split chunks are fed into the generators, from which the outputs are concatenated and fed to the discriminator. The discriminator then compares the concatenated spectrograms with the real spectrograms. This trains the generator to be able to produce chunks that result in realistic spectrograms when concatenated together. After the training, given a notification, its input spectrogram is split into chunks of length $L2$ to infer the output, where $L2$ can be different from $L1$.

3. EXPERIMENTS

We evaluate the performance of the model in changing the notification timbre while preserving the original melody envelope. The following experiment focuses on the transfer to the timbre of piano, but we also discuss the potential and limitations of our model based on the experiments on several other target timbres. As a reference, related audio examples can be found in the supplement[1].

3.1. Training Data Collection

We collected around 1.5 hours of video game music from YouTube to train the transfer model for notifications. Music of target instrumental timbres was collected from the MusicNet dataset [29] and YouTube. We collected 65 notification sounds from iOS/Android and popular applications. Lengths of these notifications range from 1 to 15 seconds. All collected data was preprocessed into WAV format with a sampling rate of 16 kHz.

Video game music was selected to train the model since it shares similar timbral features with notification sounds, and there exists plenty of video game music freely available. To prove the timbral similarity, we extracted 128-dimensional VGGish feature vectors (that are suitable for recognizing different musical instruments and soundtracks [30, 31]) from notifications, from video game music, and from other music of 11 styles. Next, we visualized the feature vectors in 2D space using the popular t-SNE [32] algorithm. Fig. 2 shows how the VGGish feature vectors of the same timbre are clustered together, and how the similar timbres share some overlap, e.g. between classical music (light pink) and string quartet music (brown). As highlighted in the figure, most notification sounds

[1] Supplementary materials: `https://gladys0313.github.io/notification-timbre-transfer/`

(cyan) overlap the video game music (olive), and both of them are generally separated from the other styles. This indicates their timbral similarity and hence the rationality of training a timbre transfer model for notification sounds using the video game music as the source data. We have also explored the use of electronic music as the source domain since we assumed some timbral similarity between the electronic music and the notifications, but it turned out to be an inferior choice (see supplement[1]).

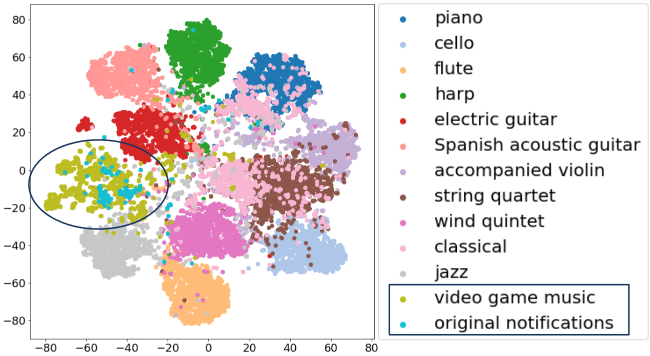

Fig. 2. Clusters of musical timbres based on extracted VGGish feature vectors. Notifications and video game music overlap.

3.2. Implementation

To train our timbre transfer model, we randomly extracted 1200 pieces of $30\,s$ audio clips from the video game music dataset and the target style music dataset, respectively. The model was trained using Adam optimizer, a learning rate of 2×10^{-4}, and batch size 16. For the training objective, we chose parameters $\lambda_{cycle} = 1$ and $\lambda_{id} = 6$. Regarding the splitting-and-concatenation (see Sec. 2.3), we used a chunk size of $3.84\,s$ during the training process, given the same length of $30\,s$ for all training data. After training, we used a chunk size of $0.384\,s$ for transferring notification sounds. The Mel-spectrograms were calculated with $hop\,size = 192$ and $window\,size = 6 \times hop\,size$. The network was implemented using the TensorFlow library. Each model was trained for one target timbre on a Tesla V100 GPU for around 6 days.

3.3. Performance Evaluation

We show the model performance on notification timbre transfer into the piano style, and present samples including other timbres in the supplement[1]. Remember that our goal is to change the notification timbre but preserve the original melody envelope.

3.3.1. *Timbre*

We plot the VGGish feature vectors of the original notifications, the transferred notifications, and the piano music in Fig. 3a. It shows that most original notifications are clustered at the top right corner, and they are the majority group that overlaps the video game music. However, some original notifications are scattered around and a few are already similar to piano music. With this distribution, we anticipate that the transfer of some notifications could be difficult. Still, we can see that the transferred notifications are overall more similar to the piano music than the original ones.

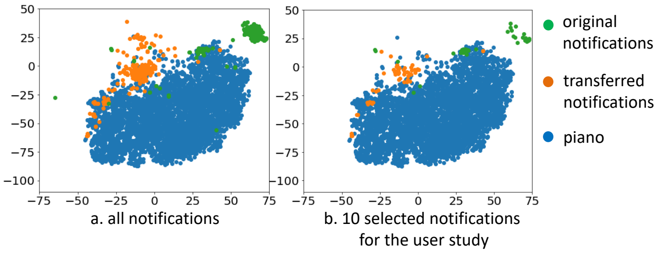

Fig. 3. Clusters of original notifications, transferred notifications, and piano music. The cluster distribution shows that the transferred notifications are in general more similar to piano music. Note: one dot represents a vector extracted from a $1\,s$ segment of the music.

3.3.2. *Melody*

Melody is regarded as a combination of pitch and rhythm. There exist numerical evaluation methods of these two properties. Note that for the pitch sequence, rather than the absolute values of each note, we concern the overall envelope and the corresponding perception. This is motivated by the fact that the notification sounds and the target instrument have mismatching ranges of fundamental frequencies and overtones. For example, a piano has a fundamental frequency range from $27.5\,Hz$ to $4186\,Hz$, but some notification sounds (and video game music) have high-energy pitches up to around $8\,kHz$ (e.g., Fig. 4a). For such notifications, our model is able to lower the tune (e.g., Fig. 4b) and render the overall tonal color as the target timbre. Therefore, the absolute pitch sequence of the output notification could be rather different from the original one, but their overall envelopes might sound similar.

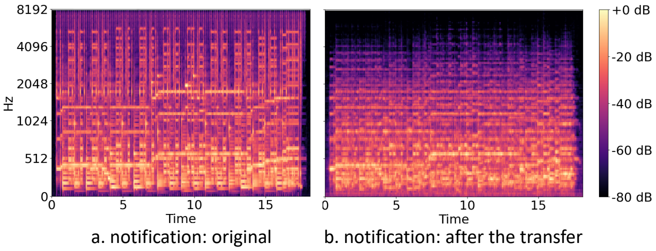

Fig. 4. Comparison of the Mel-spectrograms of the original notification (a) and its timbre-transferred version (b). Our model is able to adjust the pitches and render the overall tonal color of the original melody to match the target instrument.

We compared the original and the transferred notifications by calculating the pitch (envelope) similarity score [33] and the rhythm similarity score [9], both ranging from 0 (different) to 1 (identical). On average, the pitch similarity score was 0.458 (± 0.014) and the rhythm similarity score was 0.357 (± 0.038). This indicates that our model is able to preserve the original melody (envelope) to some extent but probably with noticeable defects.

Although numerical evaluations are objective and unresponsive to individual perception difference, a common problem is that the formulations of the metrics include less perceptually-relevant features (e.g. STFT that is on linear instead of logarithmic frequency scale), so the similarity scores might not perfectly reflect people's real auditory perception. We thus conducted a user study to explore normal people's perception of the notifications. In order to control the study time to avoid fatigue, we randomly selected 10 notifications (see Fig. 3b) for the study. As references for the timbre and the melody, we provided piano samples and the original notifications.

We collected the mean opinion score (MOS) of 53 participants (33 males, 20 females, $\overline{age}$ = 28.13) on two questions: (1) How well does the timbre of the test sample sound like piano? (2) How well does the melody of the test sample match the original notification melody? We used the typical 5-point MOS ranging from 1-bad (very different) to 5-excellent (imperceptible difference), with the neutral score 3-fair meaning "perceptible difference but acceptable".

The average timbre score was 3.345 (±0.861), indicating an overall acceptable transfer to the piano timbre but with big variation among samples. The less efficacy of the model on several notifications could be because the original timbre of these notifications was a bit different from the training data. Besides, even if the pitches of a notification have been adjusted to match the fundamental frequency range of the target instrument, the overtone pattern that is closely related to the timbre perception could still be difficult to learn.

The average melody score was 3.720 (±0.261), close to the MOS score 4-good (slightly perceptible difference, can be recognized as the same notification melody). Some participants recognized the pitch shift but they still gave a high score because the overall melody sounded to match the original one. However, some participants perceived notes that were not perceived in the original notification. Some such notes still fit the overall sequence while some caused slight discord. In general, the MOS scores and the feedback indicate an overall satisfying performance of our model in preserving the notification melody envelope after changing the timbre.

3.4. Discussion

Based on our experiments with a number of target timbres, in the following we discuss the limitations, advantages, and potential applications of our model.

First, although most of the collected video game music and notification sounds share considerable similarity, some samples are scattered away from the major cluster. Consequently, a trained model has limited performance in transferring these notifications into the target timbre. In addition, we need to re-train the model for each new target timbre, so the model is not able to live adapt to a new style at runtime, which would be desirable in some applications.

Second, most of the collected video game music and notifications have a single-instrument timbre, and most notifications are monophonic. Compared to the transfer to a single-instrument timbre (e.g. piano, cello), it is difficult to train a model that maps a monophonic notification to a homophonic or even polyphonic music piece with several timbre tracks, e.g. piano-accompanied violin. The original melody (envelope) can be kept to a good extent, but the timbre of the output notification is often unconvincing. In the case of piano-accompanied violin, perceptual evaluation showed that some output notifications sounded more similar to violin or piano, while most output notifications had an unnatural mixed timbre of these two instruments (also see Fig. 5).

An advantage of our method is its short runtime. Taking a 2.5 s notification as example, the whole pipeline in Fig. 1 takes around 3 s. Generating the output Mel-spectrogram from the input audio wave takes around 10 ms on a Tesla V100 GPU, or 50 ms on a 2.3GHz quad-core Xeon CPU, while the rest of the time is spent on reconstructing the audio wave from its Mel-spectrogram. A delay of seconds might be acceptable for delivering a notification, since in most cases notifications do not require an urgent response.

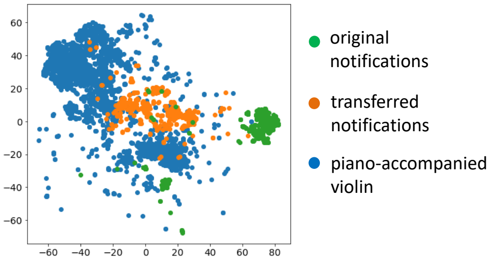

Fig. 5. Notification timbre transfer to a multi-instrument style, piano-accompanied violin. Two blue sub-clusters stand for piano and violin, respectively. Most output notifications have an unnatural mixed timbre of these two instruments.

4. NOTIFICATION TIMBRE TRANSFER FOR GENTLE INFORMATION DELIVERY

A potential application of notification timbre transfer might be less distracting information delivery. Related work [17, 18] focused on delivering notifications in a gentle way by adding acoustic effects or slightly modifying the notes and the rhythm of the music that a user is listening to. This concept of embedding notification into music might also be achieved with the notification timbre transfer technique. We could first change the timbre of the notification to match the music being played, then embed the timbre-transferred notification into the music track, so to deliver notifications in a less intrusive manner. To demonstrate this concept and to achieve seamless music augmentation, we propose an embedding approach including the following steps: (1) Scale the amplitude of the notification sound to match the amplitude of the music; (2) Adjust the speed of the notification to match the tempo of the music; (3) Integrate the notification into the music with a fade-in and fade-out effect. While a thorough exploration of the user experience with this kind of notification delivery is out of the scope of this paper, we expect adequate usability and acceptance of our approach with reference to the related work [18]. We provide corresponding audio samples in the supplement[1].

5. CONCLUSIONS AND FUTURE WORK

We proposed a CycleGAN-based pipeline and trained it in an unsupervised manner with unpaired audio samples to change the timbre of artificial notification sounds. We found that video game music could be used as the source domain for training the model for notifications, to overcome the issues of insufficient training samples and the ambiguous timbre of notifications. We implemented a splitting-and-concatenation method in the model to handle notification sounds of arbitrary lengths. We demonstrated the efficacy and discussed the limitations of our model in changing the notification timbre while preserving the original melody envelope.

Our future work will focus on more complex notification timbre transfer. We are especially interested to explore the challenging task of transferring the single-timbre and monophonic notifications to a style that contains several timbre tracks in a homophonic or polyphonic piece. As we envision applications in less distracting information delivery, we will also conduct studies exploring the usability of various music blending approaches. As one of the first explorations in the field of notification timbre transfer, we hope to inspire future research in this area and its applications in multimedia and human-computer interfaces.

6. REFERENCES

[1] T. Kaneko and H. Kameoka, "Parallel-data-free voice conversion using cycle-consistent adversarial networks," *arXiv:1711.11293*, 2017.

[2] Y. J. Luo, C. C. Hsu, K. Agres, and D. Herremans, "Singing voice conversion with disentangled representations of singer and vocal technique using variational autoencoders," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[3] E. Grinstein, N. QK. Duong, A. Ozerov, and P. Pérez, "Audio style transfer," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[4] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "TimbreTron: A WaveNet (CycleGAN (CQT(audio))) pipeline for musical timbre transfer," *arXiv:1811.09620*, 2018.

[5] J. W. Kim, R. Bittner, A. Kumar, and J. P. Bello, "Neural music synthesis for flexible timbre control," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[6] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, "Symbolic music genre transfer with CycleGan," in *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2018.

[7] E. Nakamura, K. Shibata, R. Nishikimi, and K. Yoshii, "Unsupervised melody style conversion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[8] K. Choi, C. Hawthorne, I. Simon, M. Dinculescu, and J. Engel, "Encoding musical style with transformer autoencoders," *arXiv:1912.05537*, 2019.

[9] M. Tomczak, C. Southall, and J. Hockman, "Audio style transfer with rhythmic constraints," in *Digital Audio Effects (DAFx)*, 2018.

[10] S. Barry and Y. Kim, ""style" transfer for musical audio using multiple time-frequency representations," *https://openreview.net/forum?id=BybQ7zWCb*, 2018.

[11] N. Mor, L. Wolf, A. Polyak, and Y. Taigman, "A universal music translation network," *arXiv:1805.07848*, 2018.

[12] Y. N. Hung, I. Chiang, Y. A. Chen, Y. H. Yang, et al., "Musical composition style transfer via disentangled timbre representations," *arXiv:1905.13567*, 2019.

[13] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer," *arXiv:1809.07600*, 2018.

[14] M. Pasini, "MelGAN-VC: Voice conversion and audio style transfer on arbitrarily long samples using spectrograms," *arXiv:1910.03713*, 2019.

[15] G. Mark, D. Gudith, and U. Klocke, "The cost of interrupted work: more speed and stress," in *ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, 2008.

[16] S. T. Iqbal and E. Horvitz, "Notifications and awareness: a field study of alert usage and preferences," in *ACM Conference on Computer Supported Cooperative Work (CSCW)*, 2010.

[17] L. Barrington, M. J. Lyons, D. Diegmann, and S. Abe, "Ambient display using musical effects," in *International Conference on Intelligent User Interfaces (IUI)*, 2006.

[18] I. Ananthabhotla and J. A. Paradiso, "SoundSignaling: Realtime, stylistic modification of a personal music corpus for information delivery," *ACM Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 2, no. 4, 2018.

[19] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[20] F. Fang, J. Yamagishi, I. Echizen, and J. Lorenzo-Trueba, "High-quality nonparallel voice conversion based on cycle-consistent adversarial network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[21] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, 1984.

[22] R. Decorsière, P. L. Søndergaard, E. N. MacDonald, and T. Dau, "Inversion of auditory spectrograms, traditional spectrograms, and other envelope representations," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 1, 2014.

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2015.

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv:1802.05957*, 2018.

[26] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International Conference on Machine Learning (ICML)*, 2019.

[27] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv:1701.04862*, 2017.

[28] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[29] J. Thickstun, Z. Harchaoui, and S. Kakade, "Learning features of music from scratch," *arXiv:1611.09827*, 2016.

[30] S. Hershey, S. Chaudhuri, D. PW. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, et al., "CNN architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[31] "VGGish feature extractor trained on YouTube data," https://bit.ly/2Gttm9v, accessed: 2020-10-08.

[32] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, 2008.

[33] J. Urbano, J. Lloréns, J. Morato, and S. Sánchez-Cuadrado, "Melodic similarity through shape similarity," in *International Symposium on Computer Music Modeling and Retrieval*, 2010.