LEARNING TIMBRE SPACES DISENTANGLED FROM PITCH USING AUTOENCODER AND ADVERSARIAL PITCH CLASSIFIER

Yena Yoo, Kyogu Lee

Music Audio Research Group, Seoul National University

ABSTRACT

Neural synthesizer generates novel sound by interpolating latent codes. However, because the pitch information is not decoupled with the hidden code, generated sound gets different harmonics or jumps to different pitch in a sudden. Therefore, we aim to manage two components separately for synthesizing creative timbres, not for novel harmonics by disentangling pitch and timbre. In this paper, Autoencoder with adversarial pitch classifier is proposed in order to decouple pitch and timbre, and Autoencoders and Adversarial Autoencoders are used for comparison. We show each models have different aspect of interpolation qualities by suggested benchmark with respect to the kind of instruments.

Index Terms— Synthesizer, encoder adversarial loss, disentanglement, autoencoders, adversarial autoencoders, conditional autoencoders, timbre

1. INTRODUCTION

Sound synthesis is the process of generating the similar electronic signal which stimulates the intended psychoacoustic sense. In the mathematical view, fundamental frequency and intensity are used as variables to the algorithms, which consists filter and envelope generator functions. By professional synthesists' demands for more fancy and complicated timbres, synthesizer has been developed more structurally complicated by adding numerous parameters. Lots of parameters result beginners have difficulty in making sound. In addition, low correlation between parameters and auditory psychoacoustic attributes is another reason for hard operation [1].

Neural audio synthesizers bring much more intuitive paradigm shift to audio synthesis, which combines audio data points given by users, to resolve the complex operation issue [2,3]. However, they don't disentangle timbre and pitch information in the latent space. Since Nsynth's decoder disregards conditioned pitch, generated sound obtains different harmonics compared with inputs' pitch or alters pitch in a sudden. For users who need to fusion timbre with specific pitch, these components should be coupled each other.

Recently, neural generator was proposed for decoder to recreate instrument from the symbol as form of the one-hot encoding of separate information such as pitch, instrument, velocity, and time [4]. However, these 4 symbols prevent users from trying their own audio from out of domains. Also, they need encoders to extract all of the independent components for reconstruction.

Disentanglement method such as conditional autoencoders [5,6,7] and supervised adversarial autoencoders [8] have been proposed. Among conditional autoencoders models, fader network generates realistic images of high resolution without needing GAN to decoder output, which results in much simpler training schemes. In addition, AAEs have more compact and even latent spaces than VAE, which improves interpolation.

In this paper, we demonstrate empirically which architecture improves interpolations among the above models, Autoencoders, Autoencoders with adversarial pitch classifier, and Supervised Adversarial Autoencoders.

We compare models qualitatively by decoding interpolations of latent codes from two inputs. Also, by measuring distance between MFCC of the two data points and that of the decoded combination of the latent codes, we search the model which has the closest MFCC distances.

2. ADVERSARIAL PITCH CLASSIFIER

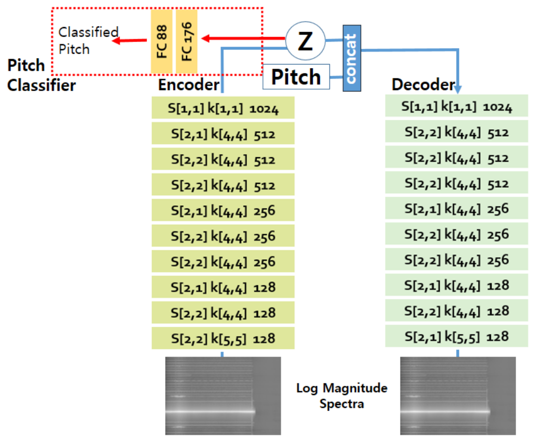


Fig. 1. : Encoder's block represents 2D convolution with stride(s), kernel size(k), and channels, and decoder's one is convolution transpose.

Previous work [5] focused on the disentanglement of the salient information of images and binary attribute values such

as $y = \{0,1\}^n$ to modify an image like faders. However, because in this work we care about monophonic instrument, we condition on pitch by providing one-hot encoding $\in \mathbb{R}^{88}$. The other difference is pitch embedding is concatenated with hidden code, not every layers of decoder.
Each layer is followed by a Leaky-RELU (0.1) nonlinearity and batch normalization, and the decoder is trained to recover the log magnitude of the power spectra, which is normalized to be between 0 and 1 with shape of $\mathbb{R}^{512} \times \mathbb{R}^{128}$, from hidden embedding $\in \mathbb{R}^{1984}$ concatenated with one-hot pitch representation $\in \mathbb{R}^{88}$.

2.1. Pitch classifier objective

Pitch classifier is trained to identify pitch information from the hidden codes, and it outputs probabilities of each pitches $P_{\theta_{pit\_c}}(y|E_{\theta_{enc}}(x))$, where θ_{pit_c} are the classifier's parameters.

$$L_{pit\_c}(\theta_{pit\_c}|\theta_{enc}) = -\frac{1}{m}\sum_{(x,y)\in D} y * \, log\, P_{\theta_{pit\_c}}(y|E_{\theta_{enc}}(x))$$

2.2. Adversarial objective

Encoder is trained to fool pitch classifier not to recognize pitch label, so it drives the classifier output other labels. λ controls the trade-off between the reconstruction quality and the degree of disentanglement.

$$\mathrm{L}(\theta_{pit\_c}, \theta_{dec}|\theta_{pit\_c}) = \frac{1}{N}\sum_{(x,y)\in D} \left\|D_{\theta_{dec}}(E_{\theta_{enc}}(x), y) - x\right\|^2$$
$$-\lambda * (1-\mathrm{y}) * \log P_{\theta_{pit\_c}}(1-y|E_{\theta_{enc}}(x))$$

Pitch classifier is fully-connected neural network of 2 layers of size 196 and 88 respectively. We set the dropout rate 0.3 at the pitch classifier. λ is set zero initially, and it's linearly increased to 0.00001 over the first 30 epochs and stopped to increase up to 60 epochs. λ should be small enough not to make pitch classifier loss not diverged.

3. EXPERIMENTS

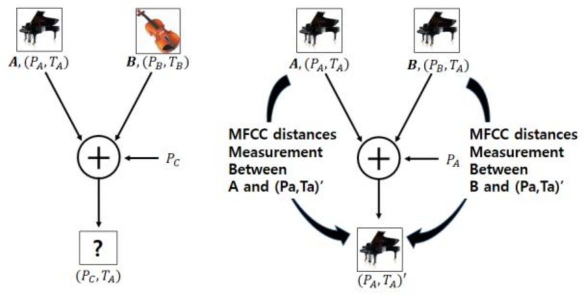
3.1 Dataset

We used Nsynth Dataset [2], which is divided with training set (#: 289205) and test set with 4096 examples. It's annotated by family (instrument), quality (like bright, dark, etc), source (acoustic, electric, synthetic), and pitch. Training

3.2 Models for comparison

Autoencoders and Adversarial Autoencoders are used for comparison.
They have the same pitch conditioned spectral Autoencoder used as baseline in [2]. All models were trained with Adam, using learning rate of 0.0001, $\beta_1 = 0.5$, and batch size of 8.
Autoencoder is represented as AE, and Autoencoder with adversarial pitch classifier as AE-APC, and Supervised Adversarial Autoencoder is denoted as AAE.
We trained AE and AE-APC for 60 epochs, and AAE for 80 epochs.

3.3. Evaluation metrics



$A, (P_A, T_A)$
$B, (P_B, T_B)$
P_C
?
(P_C, T_A)
$A, (P_A, T_A)$
$B, (P_B, T_A)$
MFCC distances Measurement Between A and (Pa,Ta)'
P_A
MFCC distances Measurement Between B and (Pa,Ta)'
$(P_A, T_A)'$

Fig. 2. Left picture shows general interpolations between different instruments. Right one shows interpolation in the same instrument has the specific example.

The combination of audio data points, A and B, could be described as $(P_A, T_A) + (P_B, T_B) + P_C$. P and T indicates pitch and timbre, and P_C indicates condition on pitch. Then, $(P_C, \frac{T_A + T_B}{2})$ should result. However, since $\frac{T_A + T_B}{2}$ doesn't have obvious answers, qualitative evaluation about how much natural and realistic the interpolations has been used.
Therefore, we suggest a benchmark by Mel-Frequency Cepstral Coefficient(MFCC) value between interpolated results and input data points from the case where the audio combination result is defined, such as $(P_A, T_A) + (P_B, T_A) + P_C = (P_C, T_A)$. Then we could measure how close MFCC vector are between the original audio data and interpolated result. However, in this paper, we noticed it's hard to find the original audio from dataset satisfying (P_C, T_A). Therefore, conditioning pitch with $P_C = P_A$ regenerate the input data, (P_A, T_A), where $(P_A, T_A) + (P_B, T_A) + P_A = (P_A, T_A)'$.
MFCC is used to compare the model's interpolation performance, since it defines the identity of instruments, because MFCC measures how frequency intensity in harmonics is related each other. In other words, different instruments with same note are recognized different sound due to their different ratio of the frequency intensity. That's why MFCC is used to find similarity of timbre.
Audio samples are available online.

https://github.com/yena53/hello-world

4. RESULTS

4.1. Interpolation in Timbre

4.1.1. MFCC measurement
MFCC distance is mean squared difference between MFCC from the two audio.

Table 1: Among 4096 testset dataset, MFCC distances between the original input data points (A and B) and the interpolated audio ,$(\boldsymbol{P_A}, \boldsymbol{T_A})'$, are averaged. D(E(A)) represents the reconstruction of A, and D(E(B)) means the reconstructed B.

MFCC distances	AE	AE-APC	AAE
between $(P_A, T_A)'$ and A	452.85	435.18	1355.71
between $(P_A, T_A)'$ and B	506.88	493.98	1415.01
between $(P_A, T_A)'$ and D(E(A))	47.07	49.03	69.99
between $(P_A, T_A)'$ and D(E(B))	121.15	137.41	152.42

Autoencoders with adversarial pitch loss has the smallest distance in overall. The MFCC distances for each instrument is represented in Fig. 2. In overall, AE has the closest MFCC distances, but AE-APC has the better result with respect to hard instruments for reconstruction, such as electronic bass, acoustic brass, acoustic flute, synthetic flute, acoustic string, and synthetic vocal.

4.1.2. Classification accuracy for pitch, family, quality, and source of interpolated audio.
In Nsynth dataset, family consists of the kind of 11 instruments, and quality means the psychoacoustic audio effect like distortion, delay, etc. In the last, the source means electronic, acoustic, and synthetic sound.

Table 2: Classification accuracy for pitch, family, quality and the source of interpolated audio.

Accuracy (# class)	AE	AE-APC	AAE
Pitch (88)	86.20	93.89	82.20
Pitch (12)	88.13	96.09	87.86
Registers(8)	96.75	96.94	84.93
Family (11)	21.97	21.06	21.65
Quality (10)	85.66	83.05	84.25
Source (3)	35.22	33.49	33.91

AE-APC has the better result about pitch reconstruction, but has the worse about other components.

4.1.3. Rainbowgram
In the Fig. 3, it's hard to figure out which one has the better interpolations.

4.2. Reconstruction

4.1.1 MFCC

Table 3: Among 4096 testset dataset, MFCC distances between the original input data points A and D(E(A)).

MFCC distances	AE	AE-APC	AAE
between A and D(E(A))	402.95	388.66	1307.93

AE-APC loss has the smallest distance in overall.

4.1.2. Classification accuracy for pitch, family, quality, and source of reconstructed audio.

Table 4: Spectral loss and classification accuracy for pitch, family, quality and the source of reconstructed audio.

Loss/ Accuracy (# class)	AE	AE-APC	AAE
Spectral loss	0.038	0.051	0.105
Pitch (88)	86.69	93.87	82.66
Pitch (12)	88.50	96.16	88.01
Registers(8)	96.82	96.92	85.40
Family (11)	21.60	20.94	21.48
Quality (10)	85.57	33.49	84.21
Source (3)	35.54	83.15	33.91

4.1.3. Rainbowgram

In the Fig. 4, AAE results noise a lot in case of vocal and brass, which is resulted badly in MFCC experiments in AE, AE-APC too.

4.3. Disentanglement of Pitch and Timbre

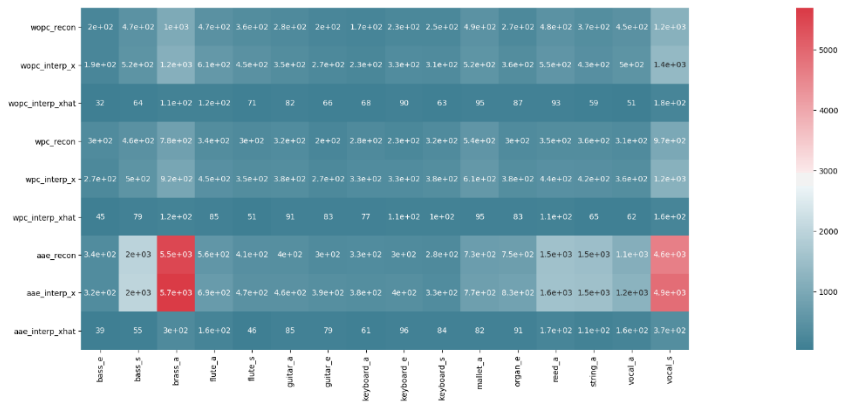
Table 5: Pitch classification on the latent code

	Pitch classification accuracy
AE	0.142
AE-APC	0.101
AAE	0.084

AAE has lack of information about pitch in the hidden code.

5. CONCLUSION

AE's audio output tends to converge to soft sound. AE-APC sounds more robust and distorted, so results better with respect to brass, flute, string, vocal.
The quality of interpolations through visualization of spectrogram couldn't be figured out well, but MFCC distance makes easy to compare the model about reconstructions and interpolations.



	bass_e	bass_s	brass_a	flute_a	flute_s	guitar_a	guitar_e	keyboard_a	keyboard_e	keyboard_s	mallet_a	organ_e	reed_a	string_a	vocal_a	vocal_s
wopc_recon	2e+02	4.7e+02	1e+03	4.7e+02	3.6e+02	2.8e+02	2e+02	1.7e+02	2.3e+02	2.5e+02	4.9e+02	2.7e+02	4.8e+02	3.7e+02	4.5e+02	1.2e+03
wopc_interp_x	1.9e+02	5.2e+02	1.2e+03	6.1e+02	4.5e+02	3.5e+02	2.7e+02	2.3e+02	3.3e+02	3.1e+02	5.2e+02	3.6e+02	5.5e+02	4.3e+02	5e+02	1.4e+03
wopc_interp_xhat	32	64	1.1e+02	1.2e+02	71	82	66	68	90	63	95	87	93	59	51	1.8e+02
wpc_recon	3e+02	4.6e+02	7.8e+02	3.4e+02	3e+02	3.2e+02	2e+02	2.8e+02	2.3e+02	3.2e+02	5.4e+02	3e+02	3.5e+02	3.6e+02	3.1e+02	9.7e+02
wpc_interp_x	2.7e+02	5e+02	9.2e+02	4.5e+02	3.5e+02	3.8e+02	2.7e+02	3.3e+02	3.3e+02	3.8e+02	6.1e+02	3.8e+02	4.4e+02	4.2e+02	3.6e+02	1.2e+03
wpc_interp_xhat	45	79	1.2e+02	85	51	91	83	77	1.1e+02	1e+02	95	83	1.1e+02	65	62	1.6e+02
aae_recon	3.4e+02	2e+03	5.5e+03	5.6e+02	4.1e+02	4e+02	3e+02	3.3e+02	3e+02	2.8e+02	7.3e+02	7.5e+02	1.5e+03	1.5e+03	1.1e+03	4.6e+03
aae_interp_x	3.2e+02	2e+03	5.7e+03	6.9e+02	4.7e+02	4.6e+02	3.9e+02	3.8e+02	4e+02	3.3e+02	7.7e+02	8.3e+02	1.6e+03	1.5e+03	1.2e+03	4.9e+03
aae_interp_xhat	39	55	3e+02	1.6e+02	46	85	79	61	96	84	82	91	1.7e+02	1.1e+02	1.6e+02	3.7e+02

Fig. 3. MFCC distances for each instruments and source, such as acoustic(a), electric(e), and synthetic(s). WOPC means "without pitch classifer", and WPC means "with pitch classifier". Interp_x represents mfcc between interpolation and the original input x, and interp_xhat represent mfcc between interpolation and reconstructed x.

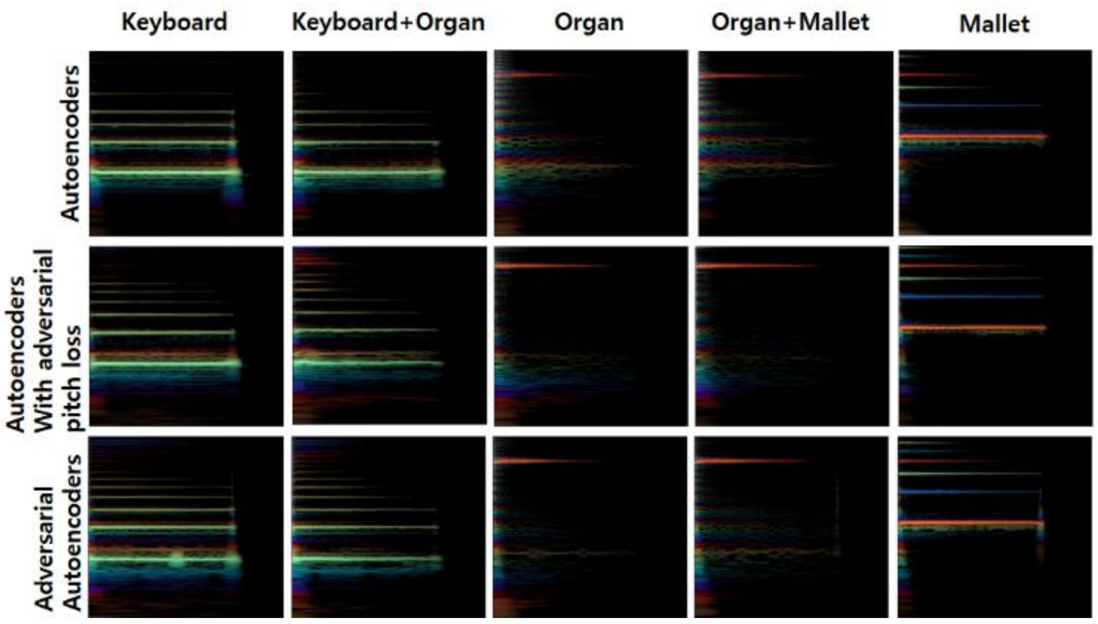


Fig. 4. Rainbowgrams of linear interpolations, pitch condition is provided left side of the interpolation audio. For example, audio combination of Keyboard + organ is provided by keyboard's pitch.

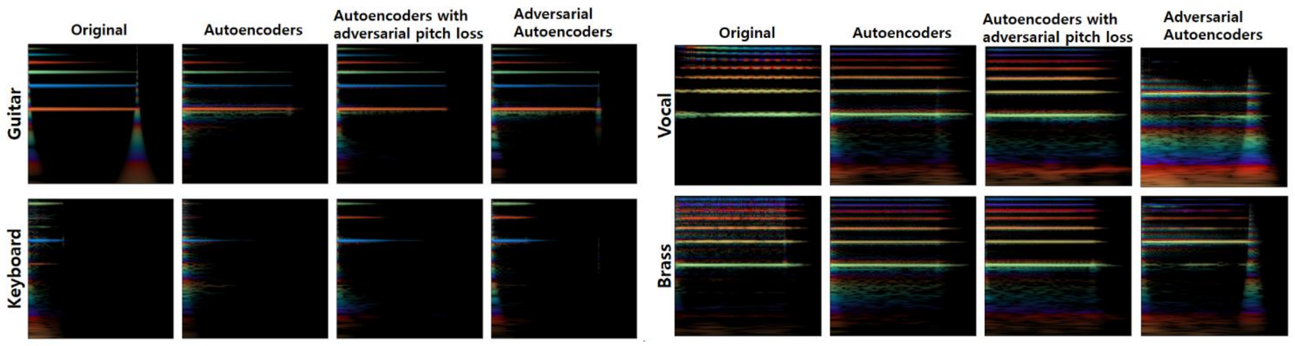


Fig. 5. Rainbowgrams of reconstructions.

12. REFERENCES

[1] J. McDermott, N. J. Griffith, and M. O'Neil, "Timbral, perceptual, and statistical attributes for synthesized sound" in Proc, of the International Computer Music Conference, 2006.

[2] Jesse Engel et al., "Neural audio synthesis of musical notes with wavenet autoencoders," 34th international conference on machine learning, 2017

[3] Phillippe et al, "Bridging audio analysis perception and synthesis with perceptually-regularized variational timbre spaces", international society of music information retrieval, 2018

[4] Alexandre et al, "SING: Symbol-to-Instrument Neural Generator", 32nd Conference on Neural Information Processing Systems, 2018.

[5] Guillaume et al, "Fader Networks: Manipulating Images by Sliding Attributes", 31st Conference on Neural Information Processing Systems, 2017.

[6] Wei Ren et al, "Improved ArtGAN for Conditional Synthesis of Natural Image and Artwork", Transactions on Image Processing, IEEE, 2019

[7] Antonia et al, "Adversarial information factorization", arXiv preprint arXiv: 1711.05175, 2018

[8] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. arXiv preprint arXiv:1511.05644, 2015.