TSM-Net: 以對抗式時序壓縮自編碼器為基礎的音訊變速演算法

TSM-Net: Temporal Compressing Autoencoder with Adversarial Losses for Time-Scale Modification on Audio Signals

國立中山大學資訊工程學系

110 學年度大學部專題製作競賽

組員：B073040018 朱劭璿
B072010029 陳居廷

指導教師：陳嘉平教授

Abstract

We proposed a novel approach in the field of time-scale modification on the audio signals. While traditional methods use framing technique and spectral approaches use short-time Fourier transform to get high-level units. TSM-Net, our neural-network model encodes the raw audio into a high-level latent representation called Neuralgram. Since the resulting Neuralgram is a two-dimensional image with real values, we apply some existing image resizing techniques on the Neuralgram and decode it using our neural decoder to obtain the time-scaled audio. Our method yields little artifacts and opens a new possibility in the research of modern time-scale modification.

Contents

1 Introduction

With the advance of technologies and digitalization, we can store and reproduce multimedia content nowadays. We can even manipulate the materials in a way that we couldn't imagine before digitalization. For example, image resizing and video editing, which change the dimensionality of the digital pictures spatially and temporally, respectively. Another ubiquitous application regarding audio signals called time-scaled modification (TSM) is used in our daily life. It's also known as playback speed control in the video streaming platforms such as YouTube.

With the power of artificial intelligence (AI) and modern computation hardware, however, we haven't discovered any method using AI to refine TSM algorithm and leverage the quality of the synthetic audio to the next level. Consider we have pragmatic AI tools in similar domains like image super-resolution [18] and motion estimation and motion compensation (MEMC) [1], etc.

1.1 Time-scale modification

Time-domain approach The main idea of TSM is that instead of scaling the raw waveforms on the time axis, which leads to pitch shifts due to the changes of wavelengths, we segment the audio into small chunks of fixed length, a.k.a frames or windows to keep the wavelength intact. In order to minimize the boundary breakage after processing, the adjacent frames are overlapped and rearranged to obtain the synthetic audio. As shown in Figure 1. The original distance between the start of each frame is called analysis hop size. After frame relocation, the distance becomes synthesis hop size, and the ratio of the analysis hop size and the synthesis hop size is the rate at which the audio is sped up or down [5]. In addition, the Hann window [6] is applied to each analysis frame to maintain the amplitude of overlapped areas. The main challenge is the harmonic alignment problem, as shown in Figure 2. With significant periodicities, an unconstrained ratio of the analysis to synthesis hop size can cause a discrepancy with the original waveform. Specifically, the phases of the same frequency components in the frames do not synchronize properly, which leads to serious interference despite the identical waveforms. Several enhancements have been proposed to solve the synchronization problem [10][23][36]. However, the resultant sound is usually non-natural and contains audible clipping artifacts. This is the negative effect of the framing technique. Moreover, the time-domain TSM

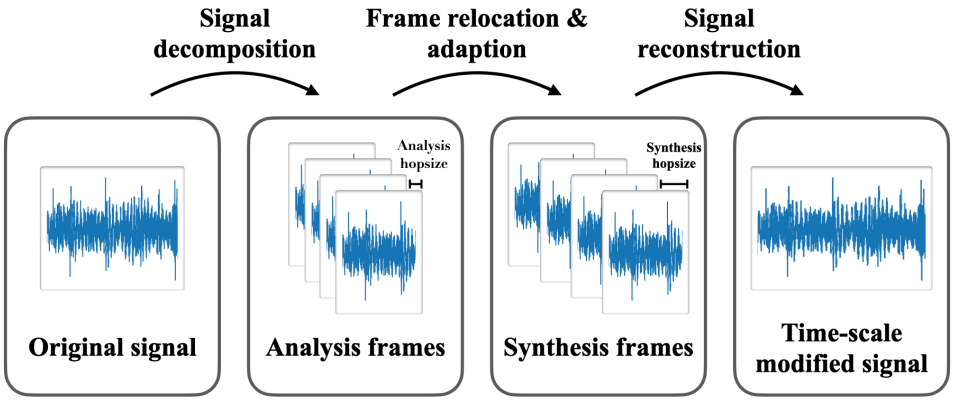

Figure 1: Generic processing pipeline of time-domain time-scale modification (TSM) procedures.

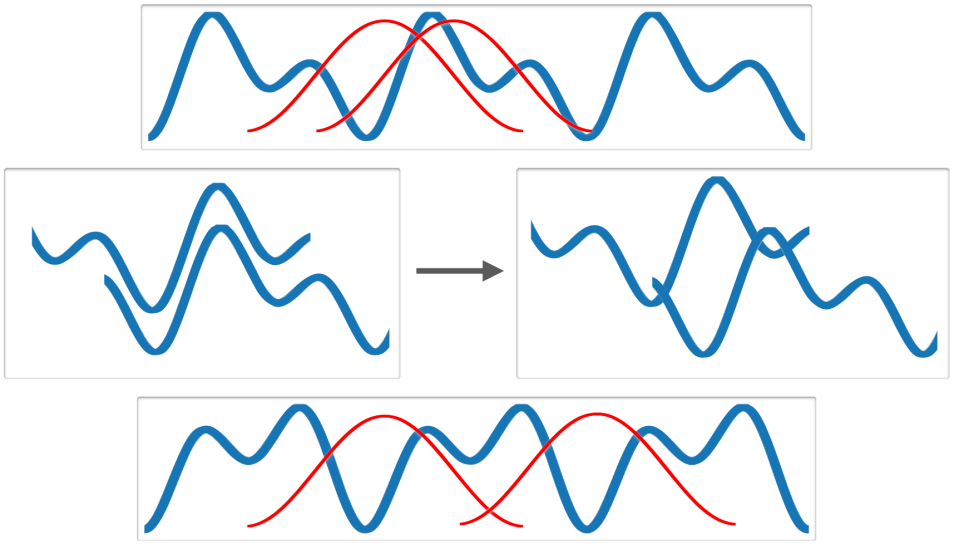

Figure 2: An illustration of the harmonic alignment problem. The red Hann windows indicate the rearrangement of the frames. An unconstrained scale ratio would lead to serious interference.

only preserves the most prominent periodicity. For the audio with a wide range of frequencies composition, like pop music, symphony and orchestra, the less prominent sound are often erased in the process.

Spectral-domain approach Another approach tries to manipulate the audio in the spectral space, using short-time Fourier transform (STFT) to convert the frequency information in the raw waveform to a more semantic representation with complex numbers [17]. The magnitude and phase parts can be further derived. Unfortunately, unlike the magnitudes, which give constructive and straightforward audio features, the phases are relatively complicated and hard to

model. Moreover, due to the heavy correlation between each phase bin, we have to use a phase vocoder [7] to estimate phases and the instantaneous frequencies after carefully relocate STFT bins to avoid a peculiar artifacts, a.k.a. phasiness. Despite some refined methods [14][21][24], which improve both the vertical and horizontal phase coherence, the spectral representation is essentially not directly scalable, and the iterative phase propagation process in the phase vocoder is an inevitable overhead.

1.2 Harnessing the power of neural networks

As we've mentioned above, the task requires a highly temporally compressed representation of the audio signals. The neural networks naturally come into our minds. The neural networks are composed of a sequence of linear transformations joined by nonlinear activation functions. With numerous configurable parameters, also known as weights, they are capable of approximating almost any function we desire, including the compression function that transforms the raw audio waveforms into low-dimensional latent vectors and back. The parameters are initially random noises and can be gradually inferred using a technique called gradient descent, in which we specify a meaningful loss function such as the difference between the networks' output and the target value, then the parameters can be configured based on the gradients of the loss function so the output would slowly move toward the target value.

TSM-Net To transform back and forth in two domains, we can think of the neural networks as an encoder and a decoder, in the jargon of machine learning, this kind of architecture is called autoencoder [15], which encodes the data into a high-level (and typically low-dimensional) latent vectors and tries its best to decode it back to the original data. In our neural network model, the dimension of the latent vectors is 1024 times smaller than the original one, which means one sample in the latent vector can represent more than an entire wave in the raw audio waveform. Since the latent vector is an image-like multi-dimension vector, we can apply the existing image resizing techniques to scale it. Finally, we decode the resized latent vector to obtain the time-scaled audio waveform.

Distribution modeling In order to make the model generalizes on the unseen data, we have to model the data distribution instead of directly measuring the L2 distance on the existing

dataset, which leads to blurry output upon unseen data. Therefore, we employ a discriminative neural network to help us train the autoencoder [8]. The discriminative network computes the adversarial loss, which measures how well it distinguishes the real and generated data. We will go into detail about the training objective and its effectiveness later.

2 Related Works

Modeling audio is not a trivial task for neural networks. To illustrate it, we can take image generation for example. DCGAN [29] is a generative neural network which synthesizes realistic images with dimension of $3 \times 64 \times 64$. There are 12288 pixels that the model needs to estimate. A 5 seconds stereo audio clip with sampling rate of 22050 Hz has $2 \times 5 \times 22050 = 220500$ samples. Not to mention each pixel is stored in 8 bits while each audio sample is stored in 16 bits and has 256 times possible values than an image pixel. Decreasing the sampling rate to simplify the dimensionality is another option. However, the Nyquist-Shannon sampling theorem suggests that the low sampling rate would lead to serious aliasing.

2.1 Neural vocoder

Models that directly generate raw audio waveform are known as vocoder. A vocoder can be conditioned on some high-level abstract features such as linguistic features or spectrograms. The spectrogram is the magnitude part from the output of STFT. It expresses the frequency composition clearly and is easy to model thanks to its smooth variations over time. As mentioned above, the phases are relative hard to estimate, therefore in applications like text-to-speech (TTS) [33] pipeline, the network often predicts the speech spectrogram of given texts, then uses a vocoder to get the raw audio. The early vocoders include Griffin-Lim [9], WORLD [22], etc. The modern neural-based vocoders start with WaveNet [35], which predicts the distribution for each audio sample conditioned on all previous ones. However, the autoregressive model runs too slow to apply on real-time applications. FloWaveNet [13] and WaveGlow [28] are neural vocoders based on bipartite transforms. They present a faster inference speed and high-quality synthetic audio but require larger models and more parameters to be as expressive as the autoregressive models, and thus harder to train. WaveGAN [4], MelGAN [16] and VocGAN [38] employ the

generative adversarial network [8] training architecture in which a discriminator is used to measure the divergence of synthetic audio and the real audio and try to learn the generator optimal weights to make the synthetic audio as realistic as possible. The discriminator usually works in multiple scales in order to take care of different frequency bands in the audio data. This kind of approach allows small models to generate high-fidelity audio samples.

2.2 Nyquist-Shannon sampling theorem

The sampling rate is the number of samples per second that a signal is recorded, and the bit depth is the amplitude precision in which each sample is stored. The analog continuous signal has to be discretized and quantized in order to be stored digitally in the computer. Specifically, discretization and quantization transfer continuous signals into discrete counterparts along the time and amplitude axis, respectively. Common combinations include 44100Hz/24bit, 22050Hz/16bit, 16000Hz/16bit, etc. A lower sampling rate results in lower data dimension, which is beneficial for reducing modeling complexity. However, a lower sampling rate is easier to produce aliasing. Aliasing is a phenomenon where two different signals alias one another in the discrete domain as illustrated in Figure 3. As pointed out and proven by Nyquist [25] and Shannon [31], a signal containing no frequency higher or equal than ωHz, is completely determined by sampling at 2ωHz, i.e., to resolve all frequencies in a signal, it has to be sampled at strictly greater than twice the highest frequency present. Otherwise, the discrete representation cannot determine some high-frequency components which have low-frequency aliases. Aliasing sometimes happens in the real world where some human sensory organs have preceptive limitations. For example, human eyes can only see about 30 to 60 frames per second. For cars driven at a very high speed, the wheels spin too fast to be correctly perceived by human eyes, as if they're spinning backward. This is known as the wagon-wheel effect. Another example is the moiré pattern [27]. It can be usually observed in the photographs of digital monitors. The resolution of the camera is not high enough to capture the pixels on the monitors and leads to artificial patterns.

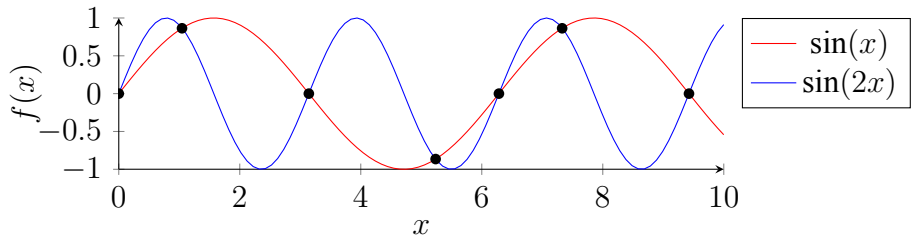

Figure 3: An illustration for aliasing. Suppose the signals are sampled non-uniformly for better illustration, two signals with different frequency components are the aliases for each other in the discrete domain, represented as black dots.

3 Methodology

3.1 Latent representation

We proposed a new representation for audio called Neuralgram to provide a novel approach to TSM. The Neuralgram is a temporally compressed feature map extracted from the middle of a neural autoencoder. A Neuralgram is applicable on TSM only when the following exists.

1. An encoder-decoder pair that is capable of fairly reconstructing the raw waveform.

2. A compression ratio that is high enough to put an entire sinusoid of the lowest frequency present into one sample in the Neuralgram

Instead of directly scaling on the raw waveform, which leads to the pitch shifting, we encode the raw waveform as a real-valued Neuralgram and scale the Neuralgram. Finally, we decode the scaled Neuralgram to get time-scaled audio without pitch shifting, as illustrated in Figure 4. Because one sample in neuralgram encodes more than an entire sinusoid for each frequency component, resizing the Neuralgram using Bilinear [32] or Bicubic [12] interpolation can repeat the entire sinusoids in the reconstructed waveform rather than changing their wavelengths and frequencies.

Neuralgram vs. Spectrogram In the literature, most of the works related to the neural vocoder use the spectrogram family as the modeling condition. Why do we bother to opt for a new representation? These traditional representations encode different frequency information into the same amount of samples in the latent space. This leads to different upsample scal-

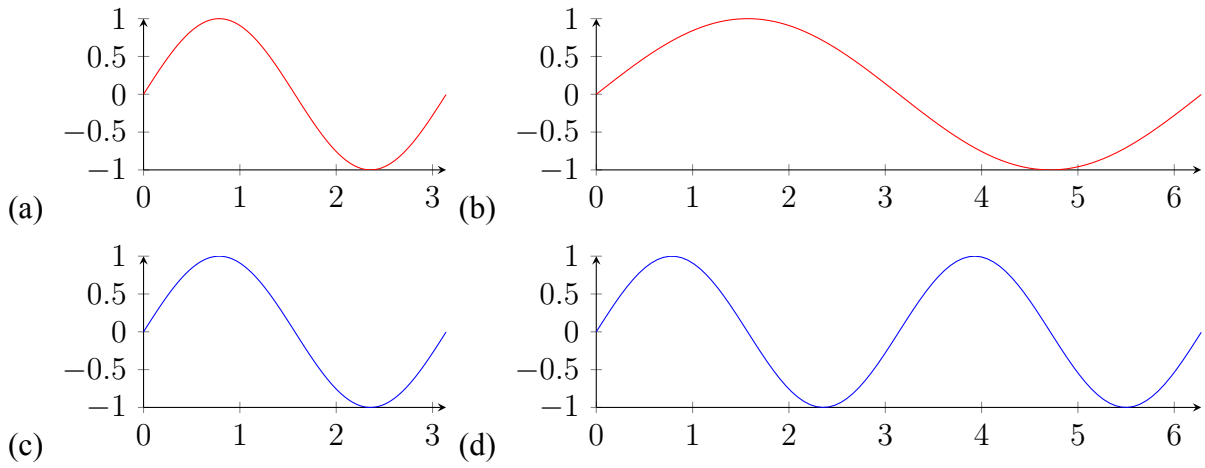

Figure 4: An illustration for the desire TSM. The original signal contains the sinusoid for a single frequency (a,c). The erroneous signal (b) is the result of directly scaling on the raw waveform, which changes the wavelength of the sinusoid and produces pitch shifting. The desired behavior (d) can be achieved by scaling on the Neuralgram, which compresses the entire sinusoid into one sample.

ing ratios during the decoding process depend on the frequency since each frequency has its own wavelength. Transformation function with the variable upsampling ratio is harder to approximate with convolutional generative models. On the other hand, our Neuralgram encodes different frequency components proportionally. This is intuitive for convolutional networks and the model is easier to train.

3.2 The TSM-Net model

Autoencoder Our model is adapted from the MelGAN [16]. In the original generator, the input is a mel-spectrogram, which would be upsampled 256× to a raw audio waveform. In the proposed model, in order to compress the entire sinusoid into one sample, we increase the upsampling rate to 1024×, because for audio with the sampling rate of 22050Hz, a 20Hz sinusoid, which is the lowest frequency the human ear perceives takes up 1102.5 samples. The upsampling is done in 5 stages of 8×, 8×, 4×, 2× and 2× upsampling. In addition, we reverse the modified generator and prepend it to the front to obtain the full autoencoder model A. Both the encoder and the decoder are initiated by an aggregating convolutional layer then followed by each downsampling/upsampling stage. Each stage is composed of a dilated downsampling/upsampling convolutional layer and a residual block that also contains a dilated convolutional

block and a skip-connection for the purpose of increasing receptive fields. As discussed in the MelGAN paper, we also use kernel-size as a multiple of stride in order to avoid checkerboard artifacts [26] and weight normalization [30] is also used after each layer to improve the sample quality. The full architecture of the autoencoder can be found in Table 1a.

Training objective We employ the same discriminators as the ones in the MelGAN. Three discriminators (D_1, D_2, D_3) work in different scales simultaneously. Except for the one works in the original scale, D_1, the downsampling is performed beforehand using strided average pooling with kernel size 4. The architecture of each discriminator can be found in Table 1b. With such arrangements, it's easier to for each discriminator to learn features for different frequency range of the audio. We can now formulate the adversarial losses for training the autoencoder A. We follow the MelGAN and use the hinge loss version of the GAN objective [19] to only penalize on the unstable data distribution. We also inherit the feature matching loss $mathcal L_{FM}$ additionally, which minimizes the L1 norm between the discriminator's feature maps of real and reconstructed audio. We accumulate the feature matching losses at each intermediate layer of all discriminator along with the training of the autoencoder.

$$\mathcal{L}_{FM}\left(A, D_k\right) = \mathbb{E}_x\left[\sum_{i=1}^{T} \frac{1}{N_i} \| D_k^{(i)}(x) - D_k^{(i)}(A(s)) \|_1\right] \tag{1}$$

where $D_k^{(i)}$ represents the ith layer's feature map output of the kth discriminator. N_i is a normalization factor and denotes the number of units in each layer. T represents the number of layers in each discriminator. Our final training objective is shown as below with $\lambda = 10$.

$$\min_{D_k} \sum_{k=1}^{3} \mathbb{E}_x\left[\min\left(0, 1 - D_k\left(x\right)\right) + \min\left(0, 1 + D_k\left(A\left(x\right)\right)\right)\right] \tag{2}$$

$$\min_{A} \left(\mathbb{E}_x\left[- \sum_{k=1}^{3} D_k\left(A\left(x\right)\right)\right] + \lambda \sum_{k=1}^{3} \mathcal{L}_{FM}\left(A, D_k\right) \right) \tag{3}$$

where x represents the raw audio waveform, A represents the autoencoder, and D_k represents each discriminator. According to [20] and [11], the noise input is not necessary when the conditioning information is very strong in the generative model.

4 Experiment

We train our model with Tesla P100 on four Datasets, including FMA [2], Musicnet [34], VCTK [37] and the audio from the Data Structure class recording [3] lectured by Prof. C.B. Yang, NSYSU. All of the audio are resampled to 22050Hz. The audio samples can be found on our website[1]. We also use two additional metrics to monitor our training but they are not included in the training losses.

1. Audio Reconstruction (AR). AR is the L1 norm between the real and reconstructed raw audio waveform

2. Neuralgram Reconstruction (NR). NR is the L1 norm between the Neuralgrams encoded from the real and reconstructed raw audio waveform, i.e., the reconstructed Neuralgram has 1.5 passes through the autoencoder.

4.1 Training techniques

The model is easy to train, taking around a weeks to train from scratch. However, it's not guaranteed to converge in every runs. A successfully converged run is shown in Figure 6. A healthy loss for discriminator tend to fluctuate around 6. As long as the discriminator's loss decreases drastically, neither the autoencoder's loss nor the feature matching loss is able to return to a healthy value and diverges quickly, thus leading to poor quality. We can double check this phenomenon from the AR and NR. Both reconstruction losses increase after the discriminator dominates the training. We can also perceive notable background noises in the reconstructed audio samples.

A broken example Sometimes, the model won't converge at all. As shown in Figure 5, the NR loss becomes zero after 5 thousands iterations. Intuitively, no matter how good a model is, it's nearly impossible to have a loss of zero. Therefore in this case, the model is not optimized. On the contrary, the encoder discards the conditions and generates the same noisy Neuralgram regardless of the input. The reconstructed audio also only contains the same noise no matter what input is fed into the model. This failure mostly happens on the FMA dataset, which contains lots of Pop music. We suspect the wide frequency range in the Pop music makes it more

[1] `https://dodoron241237.github.io/tsm-net-demo/`.

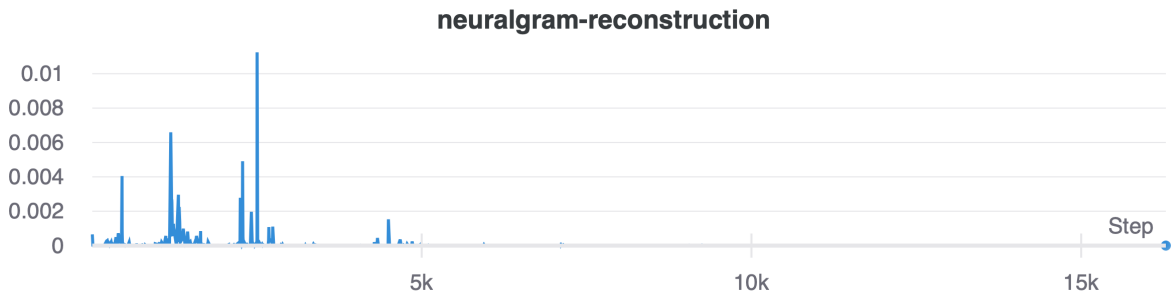

Figure 5: A broken training run.

difficult to train from scratch. We try to pre-train the autoencoder or the discriminator on the Classical music dataset, Musicnet. A pre-trained autoencoder succeeds in making the discriminator's loss stable via transfer learning. On the contrary, a pre-trained discriminator cannot guide the autoencoder to perform proper reconstruction.

References

[1] Bao, W., Lai, W.-S., Zhang, X., Gao, Z., and Yang, M.-H. Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence 43*, 3 (2021), 933–948.

[2] Benzi, K., Defferrard, M., Vandergheynst, P., and Bresson, X. FMA: A dataset for music analysis. *CoRR abs/1612.01840* (2016).

[3] Chu, E. yang-ds-speech, 10 2021.

[4] Donahue, C., McAuley, J. J., and Puckette, M. S. Synthesizing audio with generative adversarial networks. *CoRR abs/1802.04208* (2018).

[5] Driedger, J., and Müller, M. A review of time-scale modification of music signals. *Applied Sciences 6*, 2 (2016).

[6] Essenwanger, O. *Elements of Statistical Analysis*. General climatology. Elsevier, 1986.

[7] Flanagan, J. L., and Golden, R. M. Phase vocoder. *Bell System Technical Journal 45*, 9 (1966), 1493–1509.

[8] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems 27* (2014).

[9] Griffin, D., and Lim, J. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing 32*, 2 (1984), 236–243.

[10] Hejna, D., and Musicus, B. R. The solafs time-scale modification algorithm. *Bolt, Beranek and Newman (BBN) Technical Report* (1991).

[11] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017).

[12] Keys, R. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing 29*, 6 (1981), 1153–1160.

[13] Kim, S., Lee, S., Song, J., and Yoon, S. Flowavenet : A generative flow for raw audio. *CoRR abs/1811.02155* (2018).

[14] Kraft, S., Holters, M., von dem Knesebeck, A., and Zölzer, U. Improved pvsola time-stretching and pitch-shifting for polyphonic audio. In *Proceedings of the International Conference on Digital Audio Effects (DAFx), York, UK* (2012), pp. 17–21.

[15] Kramer, M. A. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal 37*, 2 (1991), 233–243.

[16] Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., Teoh, W. Z., Sotelo, J., de Brebisson, A., Bengio, Y., and Courville, A. Melgan: Generative adversarial networks for conditional waveform synthesis, 2019.

[17] Laroche, J., and Dolson, M. Improved phase vocoder time-scale modification of audio. *IEEE Transactions on Speech and Audio Processing 7*, 3 (1999), 323–332.

[18] Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., and Shi, W. Photo-realistic single image super-resolution

using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017).

[19] Lim, J. H., and Ye, J. C. Geometric gan, 2017.

[20] Mathieu, M., Couprie, C., and LeCun, Y. Deep multi-scale video prediction beyond mean square error, 2016.

[21] Moinet, A., and Dutoit, T. Pvsola: A phase vocoder with synchronized overlap-add. In *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11), Paris, France* (2011).

[22] Morise, M., Yokomori, F., and Ozawa, K. World: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE TRANSACTIONS on Information and Systems 99*, 7 (2016), 1877–1884.

[23] Moulines, E., and Charpentier, F. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication 9*, 5 (1990), 453–467. Neuropeech '89.

[24] Nagel, F., and Walther, A. A novel transient handling scheme for time stretching algorithms. In *Audio Engineering Society Convention 127* (Oct 2009).

[25] Nyquist, H. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers 47*, 2 (1928), 617–644.

[26] Odena, A., Dumoulin, V., and Olah, C. Deconvolution and checkerboard artifacts. *Distill* (2016).

[27] Petersen, D. P., and Middleton, D. Sampling and reconstruction of wave-number-limited functions in n-dimensional euclidean spaces. *Information and Control 5*, 4 (1962), 279–323.

[28] Prenger, R., Valle, R., and Catanzaro, B. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2019), pp. 3617–3621.

[29] Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[30] Salimans, T., and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems* (2016), D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, Curran Associates, Inc.

[31] Shannon, C. E. A mathematical theory of communication. *The Bell System Technical Journal 27*, 3 (1948), 379–423.

[32] Smith, P. Bilinear interpolation of digital images. *Ultramicroscopy 6*, 1 (1981), 201–204.

[33] Tan, X., Qin, T., Soong, F., and Liu, T.-Y. A survey on neural speech synthesis, 2021.

[34] Thickstun, J., Harchaoui, Z., Foster, D. P., and Kakade, S. M. Invariances and data augmentation for supervised music transcription. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2018).

[35] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *CoRR abs/1609.03499* (2016).

[36] Verhelst, W., and Roelands, M. An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech. In *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing* (1993), vol. 2, pp. 554–557 vol.2.

[37] Yamagishi, J., Veaux, C., and MacDonald, K. CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92), 2019.

[38] Yang, J., Lee, J., Kim, Y., Cho, H., and Kim, I. Vocgan: A high-fidelity real-time vocoder with a hierarchically-nested adversarial network, 2020.

Appendix A Model Architecture

Tanh 7 × 1, stride=1 conv 32 Tanh
Residual Block 32
Tanh 4 × 1, stride=2 conv 64
Residual Block 64
Tanh 4 × 1, stride=2 conv 128
Residual Block 128
Tanh 8 × 1, stride=4 conv 256
Residual Block 256
Tanh 16 × 1, stride=8 conv 512
Residual Block 512
Tanh 16 × 1, stride=8 conv 1024
Tanh 16 × 1, stride=8 conv transpose 512
Residual Block 512
Tanh 16 × 1, stride=8 conv transpose 256
Residual Block 256
Tanh 8 × 1, stride=4 conv transpose 128
Residual Block 128
Tanh 4 × 1, stride=2 conv transpose 64
Residual Block 64
Tanh 4 × 1, stride=2 conv transpose 32
Residual Block 32
Tanh 7 × 1, stride=1 conv 1 Tanh

(a) Autoencoder architecture for TSM-Net

15 × 1, stride=1 conv 16 Tanh
41 × 1, stride=4 groups=4 conv 64 Tanh
41 × 1, stride=4 groups=16 conv 256 Tanh
41 × 1, stride=4 groups=64 conv 1024 Tanh
41 × 1, stride=4 groups=256 conv 1024 Tanh
5 × 1, stride=1 conv 1024 Tanh
3 × 1, stride=1 conv 1

(b) Discriminator architecture for TSM-Net

Table 1: Autoencoder and Discriminator architecture for TSM-Net

Appendix B Training Details

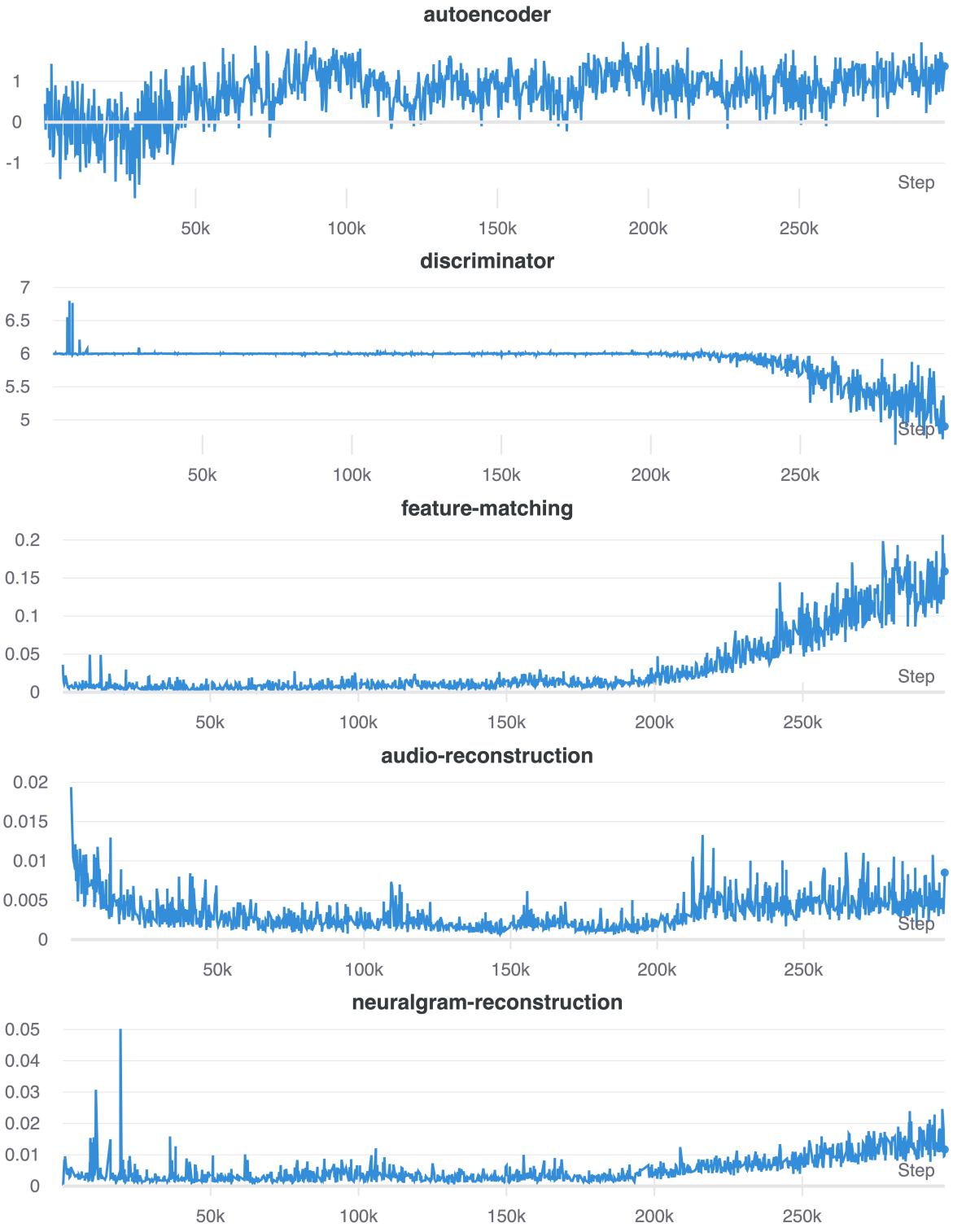

Figure 6: A successful example of the training run. The audio quality keeps improving as long as the discriminator's loss fluctuates around 6.