MUSIC2LATENT: CONSISTENCY AUTOENCODERS FOR LATENT AUDIO COMPRESSION

Marco Pasini[1] **Stefan Lattner**[2] **György Fazekas**[1]
[1]Queen Mary University of London, UK [2]Sony Computer Science Laboratories, Paris, France
m.pasini@qmul.ac.uk

ABSTRACT

Efficient audio representations in a compressed continuous latent space are critical for generative audio modeling and Music Information Retrieval (MIR) tasks. However, some existing audio autoencoders have limitations, such as multi-stage training procedures, slow iterative sampling, or low reconstruction quality. We introduce Music2Latent, an audio autoencoder that overcomes these limitations by leveraging consistency models. Music2Latent encodes samples into a compressed continuous latent space in a single end-to-end training process while enabling high-fidelity single-step reconstruction. Key innovations include conditioning the consistency model on upsampled encoder outputs at all levels through cross connections, using frequency-wise self-attention to capture long-range frequency dependencies, and employing frequency-wise learned scaling to handle varying value distributions across frequencies at different noise levels. We demonstrate that Music2Latent outperforms existing continuous audio autoencoders in sound quality and reconstruction accuracy while achieving competitive performance on downstream MIR tasks using its latent representations. To our knowledge, this represents the first successful attempt at training an end-to-end consistency autoencoder model. Pretrained weights are available under [this link]. [1]

1. INTRODUCTION

The ability to faithfully and efficiently represent high-dimensional audio data in a compressed latent space is crucial for a variety of applications, including generative modeling, music information retrieval (MIR), and audio compression. Generative models trained on latent representations of audio can be significantly more efficient than models trained directly on the data space, especially considering the high dimensionality of high-sample rate waveform samples. Additionally, a well-designed latent space can facilitate downstream MIR tasks by including musically relevant features in low-dimensional embeddings. However, existing state-of-the-art audio autoencoders often present limitations, such as a multi-stage training process, the use of an unstable adversarial objective that requires multiple discriminators, and slow iterative sampling to reconstruct audio waveforms.

In this work, we introduce Music2Latent, a novel consistency autoencoder that encodes audio samples into a continuous latent space with a high compression ratio. Music2Latent is trained fully end-to-end using a single consistency loss function, making it easier to train than many existing audio autoencoders that require a careful balance between multiple competing loss terms [1–4]. Additionally, considering the underlying consistency model [5, 6], Music2Latent can reconstruct samples with high fidelity in a single step, enabling fast and efficient decoding. We evaluate Music2Latent on audio compression metrics, that measure the discrepancy between input and reconstructed samples, and on audio quality metrics, that establish the general audio quality of the reconstructions. Despite not being the primary focus of our model, we also investigate the downstream performance of encoded representations on standard Music Information Retrieval (MIR) tasks. Our experiments demonstrate that Music2Latent reconstructs samples more accurately and with higher audio quality compared to existing continuous autoencoder baselines while providing comparable or better performance on downstream tasks. Our contributions are as follows:

- We introduce Music2Latent, a consistency autoencoder that encodes waveforms into a continuous latent space with a 4096x time compression ratio.
- We show how it is possible to achieve high-quality reconstructions with a fully end-to-end training process relying on a single loss function.
- We introduce a frequency-wise self-attention and a frequency learned scaling mechanism, and demonstrate how they improve audio quality.
- We demonstrate that Music2Latent surpasses existing continuous autoencoder models in terms of reconstruction accuracy and audio quality while achieving competitive performance on downstream MIR tasks.

To the best of our knowledge, we are the first to successfully use consistency training in the music and audio field, and we are the first across all fields to successfully train an end-to-end consistency autoencoder model.

[1] https://github.com/SonyCSLParis/music2latent

2. RELATED WORK

2.1 Autoencoders for Latent Generative Modeling

Several autoencoder approaches have been explored in both the image and audio domains.

Image Domain: Vector Quantized Variational Autoencoders (VQ-VAE) [7] introduced the concept of learning discrete latent representations of images through vector quantization. VQ-VAE-2 [8] extended this approach to hierarchical codebooks, enabling the generation of realistic images using autoregressive models trained on the learned discrete latent codes. Vector Quantized Generative Adversarial Networks (VQGAN) [9] combine the VQ-VAE framework with adversarial training, incorporating a discriminator network to improve the perceptual quality of generated images. Latent Diffusion Models (LDMs) [10] leverage diffusion models trained on the latent space of a pre-trained autoencoder. By operating on a compressed representation of the data, LDMs achieve high-quality image synthesis with reduced computational requirements compared to pixel-based diffusion models. Diffusion autoencoders [11] combine a learnable encoder with a diffusion model as the decoder, aiming to learn a meaningful and decodable representation of images in a fully end-to-end manner. However, they still require a slow iterative sampling process to reconstruct samples.

Audio Domain: The audio autoencoder proposed in the Musika music generation system [1] encodes audio into a continuous latent space by reconstructing the magnitude and phase components of a spectrogram. While Musika achieves fast inference, it requires a two-stage training process combined with an unstable adversarial objective. Moûsai introduces a diffusion autoencoder [12] to learn a compressed invertible audio representation. However, it requires multiple sampling steps for reconstruction. Several audio autoencoders employ Residual Vector Quantization (RVQ) to learn discrete latent representations. Examples include SoundStream [2], EnCodec [3], and Descript Audio Codec (DAC) [4]. These models are well-suited for training autoregressive models on the latent representations but are less suitable for other generative models such as diffusion, consistency, or GAN-based methods. They also generally produce (discrete) representations at a significantly lower time compression ratio than continuous models, and are thus not directly comparable to our work.

2.2 Consistency Models

Consistency models [5, 6] offer a novel approach for efficient generative modeling by learning a mapping from any point on a diffusion trajectory to the trajectory's starting point. They have been successfully applied to image generation tasks [13], achieving high-quality results with single-step sampling. The application of consistency models to audio generation is still relatively unexplored. CoMoSpeech [14] explores consistency distillation for speech synthesis, but it requires a pre-trained diffusion model to be trained.

3. BACKGROUND

3.1 Consistency Models

Consistency models represent a novel family of generative models capable of producing high-quality samples in a single step, without the need for adversarial training or iterative sampling. They are grounded in the probability flow ordinary differential equation (ODE) introduced by [15]:

$$\frac{dx}{d\sigma} = -\sigma \nabla_x \log p_\sigma(x), \quad \sigma \in [\sigma_{\text{min}}, \sigma_{\text{max}}] \tag{1}$$

Here, $p_\sigma(x)$ represents the perturbed data distribution obtained by adding Gaussian noise with zero mean and standard deviation σ to the original data distribution $p_{\text{data}}(x)$. The term $\nabla_x \log p_\sigma(x)$ is known as the score function, which plays a crucial role in score-based generative models [16–18]. The probability flow ODE establishes a bijective mapping between a noisy data sample $x_\sigma \sim p_\sigma(x)$ and $x_{\sigma_{\text{min}}} \sim p_{\sigma_{\text{min}}}(x) \approx x \sim p_{\text{data}}(x)$. This mapping, denoted as $f(x_\sigma, \sigma) \mapsto x_{\sigma_{\text{min}}}$, is termed the consistency function, which satisfies the boundary condition $f(x_{\sigma_{\text{min}}}, \sigma_{\text{min}}) = x_{\sigma_{\text{min}}}$. A consistency model $f_\theta(x_\sigma, \sigma)$ is a neural network trained to approximate the consistency function $f(x_\sigma, \sigma)$. To meet the boundary condition, consistency models are parameterised as:

$$f_\theta(x_\sigma, \sigma) = c_{\text{skip}}(\sigma) x_\sigma + c_{\text{out}}(\sigma) F_\theta(x_\sigma, \sigma) \tag{2}$$

where $F_\theta(x_\sigma, \sigma)$ is a free-form neural network, and $c_{\text{skip}}(\sigma)$ and $c_{\text{out}}(\sigma)$ are differentiable functions such that $c_{\text{skip}}(\sigma_{\text{min}}) = 1$ and $c_{\text{out}}(\sigma_{\text{min}}) = 0$.

Consistency models can be trained using either consistency distillation (CD) or consistency training (CT). CD requires pre-training a diffusion model to estimate the score function $\nabla_x \log p_\sigma(x)$ via score matching [19]. CT, on the other hand, allows training consistency models in isolation and is the method that is considered in this work.

3.2 Consistency Training

In consistency training, the probability flow ODE is discretised using a sequence of noise levels $\sigma_{\text{min}} = \sigma_1 < \sigma_2 < \cdots < \sigma_N = \sigma_{\text{max}}$. The consistency model $f_\theta(x_\sigma, \sigma)$ is then trained by minimising the following consistency training loss over θ:

$$\mathcal{L}_{\text{CT}} = \mathbb{E}\left[\lambda(\sigma_i, \sigma_{i+1}) d\left(f_\theta(x_{\sigma_{i+1}}, \sigma_{i+1}), f_{\theta^-}(x_{\sigma_i}, \sigma_i)\right)\right] \tag{3}$$

where $d(x, y)$ is a metric function such as mean squared error and $\lambda(\sigma_i, \sigma_{i+1})$ is a noise level-dependent loss scaling. In the above equations, f_θ and f_{θ^-} are referred to as the student network and the teacher network respectively. The teacher's parameters θ^- are obtained by applying a stop-gradient operation to the student's parameters θ during training:

$$\theta^- \leftarrow \text{stopgrad}(\theta) \tag{4}$$

After training, the consistency model $f_\theta(x, \sigma)$ can directly generate a sample x by starting with $z \sim \mathcal{N}(0, I)$ and computing $x = f_\theta(\sigma_{\text{max}} z, \sigma_{\text{max}})$. This enables efficient one-step sampling, a key advantage of consistency models over diffusion models.

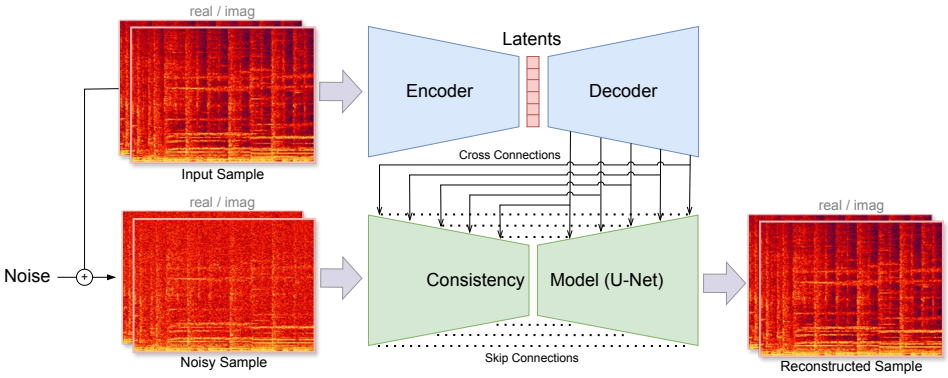

Figure 1. Training process of Music2Latent. The input sample is first encoded into a sequence of latent vectors. The latents are then upsampled with a decoder model. The consistency model is trained via consistency training, with an additional information leakage coming from the cross connections.

4. MUSIC2LATENT

In the following sections, we provide a detailed explanation of the audio representation, architecture, and training framework underlying Music2Latent.

4.1 Audio Representation

Music2Latent utilises complex-valued STFT spectrograms as the representation of waveform audio. This choice is motivated by several factors. First, previous works [20, 21] have demonstrated the effectiveness of complex spectrograms in capturing the intricate structure of audio signals and enabling the generation of high-fidelity audio. Second, 2-dimensional spectrograms allow for the direct application of UNet architectures [22] that have been successfully used in the image domain with diffusion and consistency models. However, the distribution of values across different frequencies in a STFT spectrogram can vary significantly, with substantially higher magnitudes in low frequencies compared to high frequencies. This can hinder the ability of the model to accurately reconstruct all frequency components, as the learning signal for high frequencies may be overshadowed by the stronger signal from lower frequencies. To address this issue, we apply the amplitude transformation proposed in [23] and later used in [24] which scales up lower energy components in the spectrogram:

$$\tilde{c} = \beta |c|^{\alpha} e^{i \angle(c)} \tag{5}$$

where c is the original complex STFT coefficient, $\tilde{c}$ is the transformed coefficient, $\alpha \in (0, 1]$ is a compression exponent that emphasizes lower-energy frequency components, $\angle(c)$ represents the phase angle of c, and $\beta \in \mathbb{R}^{+}$ is a scaling factor to normalize amplitudes within a desired range (e.g., $[0, 1]$). This transformation ensures that the model receives a more balanced representation of the audio signal, facilitating accurate reconstruction across all frequencies. We consider the complex STFT spectrogram as a 2-channel representation, with each channel representing real and imaginary components respectively.

4.2 Architecture

The architecture of Music2Latent consists of an encoder, a decoder, and a consistency model.

Encoder: The encoder receives as input the audio sample in the form of an STFT spectrogram with real and imaginary components in each channel. It then gradually downsamples the feature maps along the time axis and outputs a sequence of latent vectors with dimensionality d_{lat}. Instead of being trained with a VAE objective [10, 25] to keep the distribution of latent values under control, the latent encodings of the model are kept in the $(-1, 1)$ range using a $tanh$ activation function, which was proven to be a successful approach in previous works for downstream latent generative modeling tasks [1, 12].

Decoder: The decoder mirrors the encoder architecture but performs upsampling instead of downsampling. The decoder takes as input a sequence of latent vectors from the encoder and progressively upsamples them to match the dimensionality of the feature maps of the consistency model. The only purpose of the decoder is to ensure that the conditioning information from the latent encodings is available to the consistency model at all levels of its architecture (the reason for this architectural choice is provided in the next section).

Consistency Model: The consistency model uses a UNet architecture with a downsampling branch and an upsampling branch connected via additive skip connections. The output of the decoder at each upsampling layer is also added to the corresponding layer of the consistency model. This provides cross connections that allow the consistency model to directly access the conditioning information about the sample it is attempting to reconstruct at all levels of its architecture. This design choice is crucial for single-step reconstruction, as it ensures that the model has access to the necessary information to accurately reconstruct the target sample from the very beginning of the UNet architecture.

Adaptive Frequency Scaling: The distribution of values along the frequency axis in the input spectrograms changes

significantly with respect to the noise level σ. Specifically, when σ is close to σ_{min}, the magnitudes at low frequencies are on average much higher than the ones at high frequencies, while with σ approaching σ_{max}, there is an equal distribution of values across all frequencies since the sample is pure noise. To address this, we introduce a frequency-wise scaling mechanism that adaptively scales the input and output of the consistency model based on the current noise level. Specifically, we employ a Multi-Layer Perceptron (MLP) that takes as input the noise level σ in the form of a sinusoidal embedding [26] and outputs a scaling factor for each frequency bin:

$$s_f(\sigma) = \text{MLP}(\sigma), \tag{6}$$

where $s_f(\sigma) \in \mathbb{R}^F$ is a vector of scaling factors, one for each of the F frequency bins of the noisy spectrogram. We calculate different scaling factors to scale both the input x_σ and the output of the consistency model $F_\theta(x_\sigma)$ as follows:

$$\tilde{x}_\sigma = x_\sigma \odot s_{f,\text{in}}(\sigma) \quad \tilde{F}_\theta(x_\sigma) = F_\theta(x_\sigma) \odot s_{f,\text{out}}(\sigma) \tag{7}$$

where $\odot$ denotes element-wise multiplication.

Frequency-wise self-attention: To capture long-range dependencies within the frequency domain while keeping a memory footprint that scales linearly with the time axis, Music2Latent employs frequency-wise self-attention. This mechanism allows the model to attend to information from all frequency bins at a given time step, enabling it to learn complex relationships between different frequency components. Considering that only the time dimension of the input can vary at inference time, using frequency-wise attention compared to full self-attention does not incur in a memory requirement that scales quadratically with time. After computing the query Q, key K, and value V via linear projections of the input features, we calculate the attention matrix A by performing an outer product on individual timesteps t:

$$A_t = \text{softmax}\left(\frac{Q_t K_t^T}{\sqrt{d}}\right) \tag{8}$$

where d is the channel dimension, and after concatenating the attention weights from all timesteps together we have $A \in \mathbb{R}^{T\times F\times F}$. The softmax operation is then applied across the frequency dimension, ensuring that the attention weights for each frequency bin sum to one.

4.3 Training Process

Music2Latent is trained using the consistency training (CT) objective [5, 6]. As described in Sec. 3.2, the objective minimizes the discrepancy between the outputs of the consistency model at adjacent noise levels σ_i and σ_{i+1}. As for the distance metric in the consistency training loss function (Eq. 3), we use the Pseudo-Huber loss function [27] which smoothly transitions from the ℓ_1 to the squared ℓ_2 metrics:

$$d(x, y) = \sqrt{|x - y|^2 + c^2} - c, \tag{9}$$

where c is a hyperparameter that controls the transition. In [6], it was shown that for image generation with consistency models, this loss provides smoother gradients during training and performs substantially better compared to the more common squared ℓ_2 loss. The consistency model is parameterised as described in Eq. 2, with the exception that in addition to providing as input the noisy sample x_σ, we allow for information leakage of the clean sample x through the features $\boldsymbol{y}_x$ provided by the decoder via cross connections:

$$\begin{gathered}\text{lat}_x = \text{Enc}_\theta(x) \quad \boldsymbol{y}_x = \text{Dec}_\theta(\text{lat}_x) \\ f_\theta(x_\sigma, \sigma, \boldsymbol{y}_x) = c_{\text{skip}}(\sigma)x_\sigma + c_{\text{out}}(\sigma)F_\theta(x_\sigma, \sigma, \boldsymbol{y}_x)\end{gathered} \tag{10}$$

which results in the following consistency loss that is used to train the system fully end-to-end:

$$\mathcal{L} = \mathbb{E}\left[\lambda(\sigma_i, \sigma_{i+1}) d\left(f_\theta(x_{\sigma_{i+1}}, \sigma_{i+1}, \boldsymbol{y}_x), f_{\theta^-}(x_{\sigma_i}, \sigma_i, \boldsymbol{y}_x)\right)\right] \tag{11}$$

With respect to the noise level-dependent loss scaling $\lambda(\sigma_i, \sigma_{i+1})$, we follow [6] and use:

$$\lambda(\sigma_i, \sigma_{i+1}) = \frac{1}{\sigma_{i+1} - \sigma_i} \tag{12}$$

which assigns a higher weight to the loss when there is a small gap between consecutive noise levels. We also adopt the lognormal sampling of σ introduced by [28] and adopted for consistency training by [6] to focus training on a more relevant range of noise levels.

Continuous Noise Levels: Unlike the formulation presented in previous consistency model literature [5, 6], which use a discrete set of noise levels for training, Music2Latent employs a continuous noise schedule. This change is inspired by recent state-of-the-art diffusion models which notably sample noise levels from a continuous distribution [28]. Parallel work on improving the performance of consistency models also demonstrates how employing a continuous noise schedule improves results compared to the original discrete schedule [29]. Specifically, we use an exponential schedule during training to determine the step size between consecutive noise levels used for the consistency loss:

$$\Delta t_k = \Delta t_0^{\frac{k}{K}(e_K - 1)+1} \tag{13}$$

where Δt_k is the step size at training iteration k, Δt_0 is the initial step size at iteration 0, and e_K is the exponent at final iteration K. This schedule ensures that the step size decreases exponentially as training progresses, allowing the model to gradually learn finer details of the data distribution. In order to calculate σ_i and σ_{i+1}, we first sample a timestep $t_{i+1} \in [0, 1]$ with the sampling weights given by the lognormal distribution, and calculate the adjacent timestep $t_i = \max(t_{i+1} - \Delta t_k, 0)$. Finally we calculate σ_i using the time step-to-noise level mapping from [28]:

$$\sigma_i = \left(\sigma_{\text{min}}^{\frac{1}{\rho}} + t_i\left(\sigma_{\text{max}}^{\frac{1}{\rho}} - \sigma_{\text{min}}^{\frac{1}{\rho}}\right)\right)^\rho \tag{14}$$

where $\rho = 7$. We use the same mapping to calculate σ_{i+1}.

	MagnaTagATune		Beatport		TinySOL-pitchclass		TinySOL-instrument	
	AUC-ROC	AUC-PR	Micro Acc.	Macro Acc.	Micro F1	Macro F1	Micro F1	Macro F1
Musika	84.8	32.9	45.2	41.0	93.5	93.4	93.3	84.5
LatMusic	85.9	34.9	37.4	30.2	88.9	88.8	92.6	80.7
Moûsai_v2	86.2	35.4	48.2	42.0	95.1	95.1	82.8	68.6
Moûsai_v3	85.8	34.5	39.8	31.9	95.5	95.6	93.1	82.3
Music2Latent	88.6	40.0	**65.5**	**60.1**	**99.8**	**99.8**	92.6	81.0
MusiCNN-MSD	87.6	37.5	13.5	7.3	17.2	15.7	68.2	60.8
CLMR	89.9	42.6	13.9	7.8	16.8	16.2	93.5	89.7
MERT-v1-95M	**90.8**	**44.9**	50.7	44.3	98.3	98.3	**97.1**	**95.8**

Table 1. Downstream task performance on MagnaTagATune (autotagging), Beatport (key estimation), TinySOL (pitch and instrument classification). Best results among autoencoder baselines are underlined.

4.4 Implementation Details

With respect to the UNet architecture of the consistency model, we use the NCSN++ architecture introduced in [17], which consists of convolutional residual blocks with 3x3 kernels, Swish activation function [30] and Group Normalisation layers. The same residual blocks are used in both the encoder and decoder. We use sinusoidal embeddings to encode the noise level, using $\frac{\log(\sigma)}{4}$ as the input. The skip connections between the downsampling and the upsampling branches of the UNet are added instead of being concatenated, as recent works on diffusion models [31] show that addition provides better performance. Consequently, the cross connections from the decoder are also added to the corresponding UNet features, following a linear projection layer. In the encoder, before the final bottleneck layer with a $tanh$ activation function, used to constrain the latent encodings to the (-1,1) range, the 2D features are reshaped into 1D features by flattening the frequency dimension into the channel dimension, and a series of 4 residual blocks with 1D convolutions with kernel size of 3 are used. We choose $d_{lat} = 64$, which results in a $4096x$ time compression ratio and a $64x$ total compression ratio. The decoder perfectly mirrors the architecture of the encoder, while not receiving any incoming skip connections, since all the information necessary to reconstruct the clean input sample must be contained in the latent encodings. For the consistency model and encoder/decoder models we use 5 levels corresponding to 4 upsampling/downsampling operations, and in each level we use 2 residual blocks for the consistency model, and 1 residual block for the encoder and decoder. The base channels for all models are set to 64 and the channel multiplier for each of the 5 levels is set to $[1, 2, 4, 4, 4]$ for all models. We use 512 channels for the 1D convolutional blocks in the encoder and decoder. We use frequency-wise self-attention layers with 4 heads in the 3 last levels for all models, in order not to use it with higher frequency dimensions. The channels used for sinusoidal embeddings and the MLPs used for both noise level embeddings and frequency scalings are set to 256. The model has ~ 58 million parameters. The consistency training framework follows the same implementation of [6] with respect to the scaling factors $c_{\text{in}}, c_{\text{skip}}, c_{\text{out}}$, the parameter c for the pseudo-Huber loss function, the minimum and maximum noise parameters $\sigma_{\text{min}}, \sigma_{\text{max}}$, the standard deviation of the data samples σ_{data}, and the lognormal distribution values of $P_{\text{mean}}, P_{\text{std}}$. Regarding the input STFT spectrograms, we extract them using hop $= 512$, window $= 4 \cdot$ hop and we transform them using the formula presented in Sec. 4.1, with $\alpha = 0.65, \beta = 0.35$. Regarding the step size schedule for the continuous noise levels, we choose $\Delta t_0 = 0.1$ and $e_K = 3$. We train the model on waveforms of $34,304$ samples, which correspond to ~ 0.78 s of 44.1 kHz audio. The model thus produces latent representations of 44.1 kHz audio at a sampling rate of ~ 11 Hz. We use a batch size of 16 and train for $K = 800k$ iterations using the RAdam optimizer [32] with $\text{lr}_0 = 1e^{-4}, \beta_1 = 0.9, \beta_2 = 0.999$. We use a cosine learning rate decay with $\text{lr}_K = 1e^{-6}$ and we keep an Exponential Moving Average (EMA) of the parameters of all models with a momentum of 0.9999. Training takes ~ 5 days on a single RTX 3090 GPU.

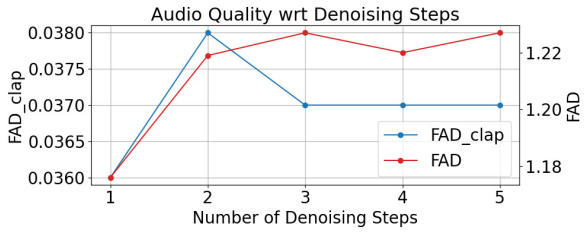

Figure 2. Audio quality of reconstructed samples with respect to the number of denoising steps of the consistency model.

5. EXPERIMENTS AND RESULTS

5.1 Datasets

We train the model on MTG Jamendo [33] and on the clean speech segments from DNS Challenge 4 [34], sampling from each dataset with equal probability. We keep the original sample rates of 44.1 kHz and 48 kHz. We include speech in the training data to both improve the reconstruction of vocal content in music samples, and to make Music2Latent useful also for speech-related tasks. We use MusicCaps [35] as our evaluation dataset.

5.2 Baselines

We compare Music2Latent to different audio autoencoders that encode audio samples into a continuous latent space to enable downstream latent generative modeling. We include the autoencoder introduced in Musika [1] and the

autoencoder introduced by [36] to train a latent diffusion model for music accompaniment generation (we name this model LatMusic in our comparison). Both models encode audio samples with the same compression ratio of $64x$ as Music2Latent. We also include the diffusion autoencoder introduced in Moûsai [12], which has a compression ratio of $32x$ (Moûsai_v3), and a different autoencoder model that is made available by the authors of Moûsai [2] with a comparable compression ratio of $64x$ (Moûsai_v2).

5.3 Audio Compression and Quality

	SI-SDR ↑	ViSQOL ↑	FAD_{clap} ↓	FAD ↓
Musika	-25.81	3.80	0.103	2.308
LatMusic	-27.32	**3.95**	0.050	1.630
Moûsai_v2	-21.44	2.36	0.731	4.687
Moûsai_v3	-17.47	2.28	0.647	4.473
Music2Latent	**-3.85**	3.84	**0.036**	**1.176**
DAC	*9.48*	*4.21*	*0.041*	*0.966*

Table 2. Audio compression and quality results.

We adopt the same objective evaluation metrics as in [3] and use Scale-Invariant Signal-to-Noise Ratio (SI-SDR) [37] and ViSQOL [38–40]. SI-SDR is a distance calculated between input and reconstructed waveforms, while ViSQOL estimates a MOS-like score on perceptual quality from the difference between the two signals. Considering that Music2Latent is trained as a generative model, we also use Frechét Audio Distance (FAD [41]) to evaluate the general audio quality of reconstructed samples without relying on paired samples. In addition to the original FAD implementation, we also evaluate on FAD_{clap} using CLAP [42] features, which was shown to correlate significantly better with perceived audio quality [43]. In Tab. 2 we show that Music2Latent is competitive with respect to ViSQOL to Musika and LatMusic, while vastly outperforming all baselines on the remaining metrics. Note that all four baselines discard phase information from the input of the autoencoder, which may explain the poor SI-SDR performance. DAC, while not being directly comparable, scores favourably in reconstruction metrics, while matches Music2Latent in terms of audio quality. In Fig. 2 we also show that the audio quality of reconstructions remains almost constant when using more than a single denoising step. We provide audio samples and additional supplementary material on the accompanying website [3].

5.4 Ablation Study

	FAD_{clap} ↓	FAD ↓
Base Model	0.0563	1.808
+ Freq-wise Attention	0.0547	1.710
+ Adaptive Freq Scaling	**0.0537**	**1.665**

Table 3. Ablation study. *Base Model* is trained without frequency-wise attention and adaptive frequency scaling.

[2] https://github.com/archinetai/archisound

[3] https://sonycslparis.github.io/music2latent-companion/

To demonstrate the effectiveness of both frequency-wise attention and learned frequency scaling, we perform an ablation study and report the FAD and FAD_{clap} results in Table 3. With respect to the model with no attention and no scaling, we use channel multipliers $[1, 2, 4, 4, 5]$ to roughly match the number of parameters that are lost. All ablated models are trained for $200k$ iterations. The remaining training details are the ones described in Sec. 4.4.

5.5 Downstream Performance

Since training representation learning models on compressed audio representations instead of raw data was shown to be a promising approach [44–47], our goal is to investigate whether there are well disentangled audio features in the feature space of audio autoencoders. We evaluate downstream performance on MagnaTagATune [48] for auto-tagging, Beatport [49] for key estimation, and TinySOL [50] for instrument and pitch class classification. For each dataset, we extract the encoder features from the layer with the highest number of output channels from each of the models (after flattening the 2D features for Music2Latent and before the last linear layer for the remaining models), average them along the time axis, and train a 2-layer MLP with $[256, 128]$ units. We also show the results obtained by performing the same evaluation on features from the classification model MusiCNN-MSD [51] and well-established representation learning models CLMR [52] and MERT-v1-95M [47] (with averaged features from layers 9 to 12). We extract features from these models following [53] and perform all evaluations using the `mir_ref` library [4] [54]. In Tab. 1 we show how Music2Latent outperforms autoencoder baselines in almost all tasks, and in the case of key and pitch classification it even outperforms state-of-the-art representation learning models. We hypothesize that the loss is more sensitive to pitch information than timbre content (explaining the weak comparison on TinySOL-instrument to the representation learning models).

6. CONCLUSION

In this work we introduced Music2Latent, a consistency autoencoder that efficiently compresses high-dimensional audio waveforms into a continuous latent space. By leveraging consistency training, Music2Latent achieves high-fidelity single-step reconstruction, and enables efficient downstream latent generative modeling. We propose a learned frequency scaling mechanism to handle varying frequency distributions across diffusion noise levels. Experiments show Music2Latent matches or outperforms baselines in reconstruction accuracy and audio quality, while having comparable or better performance on downstream tasks. To our knowledge, Music2Latent represents the first successful end-to-end consistency autoencoder. Future work could explore extensions to other modalities and higher compression ratios. Overall, we believe Music2Latent is a significant contribution to audio generative modeling and representation learning.

[4] https://github.com/chrispla/mir_ref

7. ACKNOWLEDGEMENTS

This work is supported by the EPSRC UKRI Centre for Doctoral Training in Artificial Intelligence and Music (EP/S022694/1) and Sony Computer Science Laboratories Paris.

8. REFERENCES

[1] M. Pasini and J. Schlüter, "Musika! Fast Infinite Waveform Music Generation," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022, Bengaluru, India, December 4-8, 2022*, 2022, pp. 543–550.

[2] N. Zeghidour, A. Luebs *et al.*, "SoundStream: An End-to-End Neural Audio Codec," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 30, pp. 495–507, 2022.

[3] A. Défossez, J. Copet *et al.*, "High Fidelity Neural Audio Compression," Oct. 2022, arXiv:2210.13438 [cs, eess, stat].

[4] R. Kumar, P. Seetharaman *et al.*, "High-Fidelity Audio Compression with Improved RVQGAN," Jun. 2023, arXiv:2306.06546 [cs, eess].

[5] Y. Song, P. Dhariwal *et al.*, "Consistency Models," May 2023, arXiv:2303.01469 [cs, stat].

[6] Y. Song and P. Dhariwal, "Improved techniques for training consistency models," *arXiv preprint arXiv:2310.14189*, 2023.

[7] A. van den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems 30*, Dec. 2017, pp. 6306–6315.

[8] A. Razavi, A. van den Oord *et al.*, "Generating diverse high-fidelity images with VQ-VAE-2," in *Advances in Neural Information Processing Systems 32*, Dec. 2019, pp. 14 837–14 847.

[9] P. Esser, R. Rombach *et al.*, "Taming transformers for high-resolution image synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Computer Vision Foundation / IEEE, Jun. 2021, pp. 12 873–12 883.

[10] R. Rombach, A. Blattmann *et al.*, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[11] K. Preechakul, N. Chatthee *et al.*, "Diffusion autoencoders: Toward a meaningful and decodable representation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022, pp. 10 609–10 619.

[12] F. Schneider, Z. Jin *et al.*, "Mo\^usai: Text-to-Music Generation with Long-Context Latent Diffusion," Jan. 2023, arXiv:2301.11757 [cs, eess].

[13] S. Luo, Y. Tan, L. Huang, J. Li, and H. Zhao, "Latent consistency models: Synthesizing high-resolution images with few-step inference," *arXiv preprint arXiv:2310.04378*, 2023.

[14] Z. Ye, W. Xue *et al.*, "Comospeech: One-step speech and singing voice synthesis via consistency model," in *Proceedings of the 31st ACM International Conference on Multimedia, MM 2023, Ottawa, ON, Canada, 29 October 2023- 3 November 2023*. ACM, 2023, pp. 1831–1839.

[15] J. Song, C. Meng *et al.*, "Denoising Diffusion Implicit Models," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[16] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019, pp. 11 895–11 907.

[17] Y. Song, J. Sohl-Dickstein *et al.*, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[18] Y. Song and S. Ermon, "Improved techniques for training score-based generative models," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[19] Y. Song, S. Garg *et al.*, "Sliced Score Matching: A Scalable Approach to Density and Score Estimation," in *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*, ser. Proceedings of Machine Learning Research, vol. 115. AUAI Press, 2019, pp. 574–584.

[20] J. Nistal, S. Lattner *et al.*, "DRUMGAN: synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, Oct. 2020, pp. 590–597.

[21] ——, "Comparing representations for audio synthesis using generative adversarial networks," in *28th European Signal Processing Conference (EUSIPCO)*. IEEE, Jan. 2020, pp. 161–165.

[22] O. Ronneberger, P. Fischer *et al.*, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted*

Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III, ser. Lecture Notes in Computer Science, vol. 9351. Springer, 2015, pp. 234–241.

[23] J. Richter, S. Welker *et al.*, "Speech enhancement and dereverberation with diffusion-based generative models," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 31, pp. 2351–2364, 2023.

[24] G. Zhu, Y. Wen *et al.*, "Edmsound: Spectrogram based diffusion models for efficient and high-quality audio synthesis," *arXiv preprint arXiv:2311.08667*, 2023.

[25] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations (ICLR)*, Apr. 2014.

[26] A. Vaswani, N. Shazeer *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, Dec. 2017, pp. 5998–6008.

[27] P. Charbonnier, L. Blanc-Feraud *et al.*, "Deterministic edge-preserving regularization in computed imaging," *IEEE Transactions on Image Processing*, vol. 6, no. 2, pp. 298–311, 1997.

[28] T. Karras, M. Aittala *et al.*, "Elucidating the Design Space of Diffusion-Based Generative Models," Oct. 2022, arXiv:2206.00364 [cs, stat].

[29] Z. Geng, W. Luo *et al.*, "Consistency models made easy," 2024.

[30] P. Ramachandran, B. Zoph *et al.*, "Searching for activation functions," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.

[31] P. Esser, S. Kulal *et al.*, "Scaling rectified flow transformers for high-resolution image synthesis," *arXiv preprint arXiv:2403.03206*, 2024.

[32] L. Liu, H. Jiang *et al.*, "On the variance of the adaptive learning rate and beyond," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[33] D. Bogdanov, M. Won *et al.*, "The mtg-jamendo dataset for automatic music tagging," in *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States, 2019.

[34] H. Dubey, V. Gopal *et al.*, "Icassp 2022 deep noise suppression challenge," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*. IEEE, 2022, pp. 9271–9275.

[35] A. Agostinelli, T. I. Denk *et al.*, "MusicLM: Generating Music From Text," Jan. 2023, arXiv:2301.11325 [cs, eess].

[36] M. Pasini, M. Grachten *et al.*, "Bass accompaniment generation via latent diffusion," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 1166–1170.

[37] J. L. Roux, S. Wisdom *et al.*, "SDR - half-baked or well done?" in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*. IEEE, 2019, pp. 626–630.

[38] A. Hines, J. Skoglund *et al.*, "Visqol: an objective speech quality model," *EURASIP J. Audio Speech Music. Process.*, vol. 2015, p. 13, 2015.

[39] C. Sloan, N. Harte *et al.*, "Objective assessment of perceptual audio quality using visqolaudio," *IEEE Trans. Broadcast.*, vol. 63, no. 4, pp. 693–705, 2017.

[40] M. Chinen, F. S. C. Lim *et al.*, "Visqol v3: An open source production ready objective speech and audio metric," in *Twelfth International Conference on Quality of Multimedia Experience, QoMEX 2020, Athlone, Ireland, May 26-28, 2020*. IEEE, 2020, pp. 1–6.

[41] K. Kilgour, M. Zuluaga *et al.*, "Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms," in *20th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Sep. 2019, pp. 2350–2354.

[42] Y. Wu, K. Chen *et al.*, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," in *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*. IEEE, 2023, pp. 1–5.

[43] M. Tailleur, J. Lee *et al.*, "Correlation of fr\'echet audio distance with human perception of environmental audio is embedding dependant," *arXiv preprint arXiv:2403.17508*, 2024.

[44] R. Castellon, C. Donahue *et al.*, "Codified audio language modeling learns useful representations for music information retrieval," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, 2021, pp. 88–96.

[45] L. Pepino, P. Riera *et al.*, "Encodecmae: Leveraging neural codecs for universal audio representation learning," *arXiv preprint arXiv:2309.07391*, 2023.

[46] P. Alonso-Jiménez, L. Pepino *et al.*, "Leveraging pretrained autoencoders for interpretable prototype learning of music audio," *arXiv preprint arXiv:2402.09318*, 2024.

[47] Y. Li, R. Yuan *et al.*, "Mert: Acoustic music understanding model with large-scale self-supervised training," 2023.

[48] D. Wolff, S. Stober *et al.*, "A systematic comparison of music similarity adaptation approaches," in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*. FEUP Edições, 2012, pp. 103–108.

[49] Ángel Faraldo, "Beatport edm key dataset," Jan. 2018.

[50] C. Emanuele, D. Ghisi *et al.*, "TinySOL: an audio dataset of isolated musical notes," Jan. 2020.

[51] J. Pons and X. Serra, "musicnn: Pre-trained convolutional neural networks for music audio tagging," *arXiv preprint arXiv:1909.06654*, 2019.

[52] J. Spijkervet and J. A. Burgoyne, "Contrastive learning of musical representations," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, 2021, pp. 673–681.

[53] C. Plachouras, "Beyond Benchmarks: A Toolkit for Music Audio Representation Evaluation," Ph.D. dissertation, Universitat Pompeu Fabra, Sep. 2023.

[54] C. Plachouras, P. Alonso-Jiménez *et al.*, "mir_ref: A representation evaluation framework for music information retrieval tasks," in *37th Conference on Neural Information Processing Systems (NeurIPS), Machine Learning for Audio Workshop*, New Orleans, LA, USA, 2023.