LONG-FORM MUSIC GENERATION WITH LATENT DIFFUSION

Zach Evans **Julian D. Parker** **CJ Carr**
Zack Zukowski **Josiah Taylor** **Jordi Pons**

Stability AI

ABSTRACT

Audio-based generative models for music have seen great strides recently, but so far have not managed to produce full-length music tracks with coherent musical structure from text prompts. We show that by training a generative model on long temporal contexts it is possible to produce long-form music of up to 4m 45s. Our model consists of a diffusion-transformer operating on a highly downsampled continuous latent representation (latent rate of 21.5 Hz). It obtains state-of-the-art generations according to metrics on audio quality and prompt alignment, and subjective tests reveal that it produces full-length music with coherent structure.

1. INTRODUCTION

Generation of musical audio using deep learning has been a very active area of research in the last decade. Initially, efforts were primarily directed towards the unconditional generation of musical audio [1,2]. Subsequently, attention shifted towards conditioning models directly on musical metadata [3]. Recent work has focused on adding natural language control via text conditioning [4–7], and then improving these architectures in terms of computational complexity [8–11], quality [12–15] or controlability [16–19].

Existing text-conditioned models have generally been trained on relatively short segments of music, commonly of 10-30s in length [4–7] but in some cases up to 90s [14]. These segments are usually cropped from longer compositions. Although it is possible to generate longer pieces using (e.g., autoregressive [8]) models trained from short segments of music, the resulting music shows only local coherence and does not address long-term musical structure (see Table 4, MusicGen-large-stereo results). Furthermore, the analysis of a dataset of metadata from 600k popular music tracks [1] (Figure 1) confirms that the majority of songs are much longer than the lengths addressed by previous works. Therefore, if we want to produce a model that can understand and produce natural musical structure, it is likely necessary to train and generate on a longer time window. We identify 285s (4m 45s) as a target length, as it is short enough to be within reach of current deep learning architectures, can fit into the VRAM of modern GPUs, and covers a high percentage of popular music.

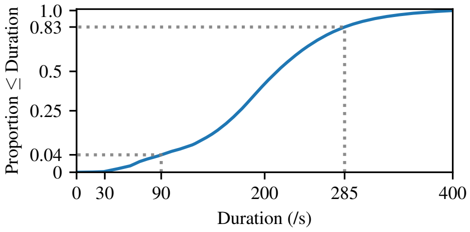


Figure 1: Cumulative histogram showing the proportion of music that is less than a particular length, for a representative sample of popular music[1]. Dotted lines: proportion associated with the max generation length of our model (285s) and of previous models (90s). The vertical axis is warped with a power law for greater readability.

In previous works [4, 20] it has been hypothesized that "semantic tokens enable long-term structural coherence, while modeling the acoustic tokens conditioned on the semantic tokens enables high-quality audio synthesis" [20]. Semantic tokens are time-varying embeddings derived from text embeddings, aiming to capture the overall characteristics and evolution of music at a high level. This intermediate representation is practical because it operates at low temporal resolution. Semantic tokens are then employed to predict acoustic embeddings, which are later utilized for waveform reconstruction.[2] Semantic tokens are commonly used in autoregressive modeling to provide guidance on what and when to stop generating [4, 20].

Another line of work [14] implicitly assumes that conditioning on semantic tokens is unnecessary for long-form music structure to emerge. Instead, it assumes that structure can emerge by training end-to-end without semantic tokens. This involves generating the entire music piece at once (full-context generation), rather than generating audio autoregressively guided by semantic tokens [4, 20]. This approach has the potential to simplify the pipeline from four stages[2] to three (text→text-embedding→acoustic-token→waveform) or even one (text→waveform). While the single-stage approach represents the closest approxi-

[1] www.kaggle.com/yamaerenay/spotify-tracks-dataset-19222021

[2] [4, 20] are typically conformed by four stages (denoted here as →): text→text-embedding→semantic-token→acoustic-token→waveform.

Attribution: Z. Evans, J. D. Parker, CJ Carr, Z. Zukowski, J. Taylor and J. Pons, "Long-form music generation with latent diffusion", in *Proc. of the 25th Int. Society for Music Information Retrieval Conf.,* San Francisco, United States, 2024.

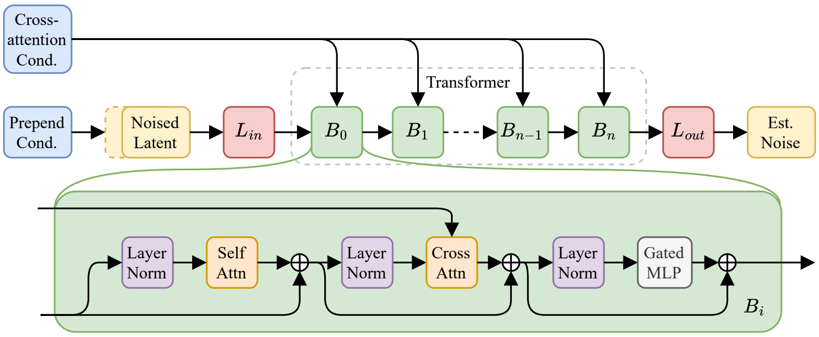


Figure 2: Architecture of the diffusion-transformer (DiT). Cross-attention includes timing and text conditioning. Prepend conditioning includes timing conditioning and also the signal conditioning on the current timestep of the diffusion process.

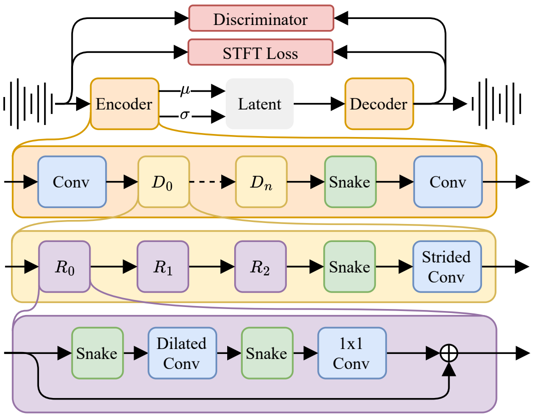


Figure 3: Architecture of the autoencoder.

mation to end-to-end learning, its may be challenging to implement due to the VRAM limitations of current GPUs. Our model consists of three stages able to generate an entire music piece of 4m 45s at once without semantic tokens.

Most music generation works rely on autoencoders to condense the long waveforms into compact latent representations (acoustic tokens or embeddings). Prominent examples utilize residual-vector-quantizers to provide discrete acoustic tokens [21–23] for autoregressive or masked token modeling [8–10, 16]. Another prominent line of work focuses on variational autoencoders to provide a continuous and normalized acoustic embedding [5, 7, 12, 14] for latent diffusion modelling. Our work relies on latent diffusion modeling to generate music from text prompts. Yet, and differently from prior works operating with latent rates of 40Hz to 150Hz [14, 23, 24], our autoencoder relies on a highly downsampled latent operating at 21.5Hz (Table 5). We argue that maintaining perceptual quality at low latent rates can be essential for training generative models on long temporal contexts, enabling the creation of long-form music without the need to rely on semantic tokens.

In our work we scale a generative model to operate over the 285s (4m 45s) time interval. This is achieved by using a highly compressed continuous latent, and a generative model relying on latent diffusion (Sections 2 and 3). The resulting model obtains state-of-the-art results in terms of audio quality and text-prompt coherence (Section 4.1), and is also capable of generating long-form music with coherent structure (Sections 4.2 and 4.4) in 13s on a GPU.

Code to reproduce our model [3] and demos [4] are online.

2. LATENT DIFFUSION ARCHITECTURE

Our model generates variable-length (up to 4m 45s) stereo music at 44.1kHz from text prompts. It comprises three main components: an autoencoder that compresses waveforms into a manageable sequence length, a contrastive text-audio embedding model based on CLAP [25, 26] for text conditioning, and a transformer-based diffusion model that operates in the latent space of the autoencoder. Check their exact parametrizations online in our code repository.[3]

2.1 Autoencoder

We employ an autoencoder structure that operates on raw waveforms (Figure 3). The encoder section processes these waveforms by a series of convolutional blocks, each of which performs downsampling and channel expansion via strided convolutions. Before each downsampling block, we employ a series of ResNet-like layers using dilated convolutions and Snake [27] activation functions for further processing. All convolutions are parameterized in a weight-normalised form. The decoder is almost identical to the encoder structure, but employs transposed strided convolutions for upsampling and channel contraction at the start of each upsampling block. The encoder and decoder structures are similar to that of DAC [23], but with the addition of a trainable β parameter in the Snake activation, which controls the magnitude of the periodicity in the activation. We also remove the $tanh()$ activation used in DAC at the output of the decoder, as we found it introduced harmonic distortion into the signal. The bottleneck of the autoencoder is parameterized as a variational autoencoder.

We train it using a variety of objectives. First, the reconstruction loss, consisting of a perceptually weighted multi-resolution STFT [28] that deals with stereo audio as follows: the STFT loss is applied to the mid-side (M/S) representation of the stereo audio, as well as the left and right (L/R) channels separately. The L/R component is weighted by 0.5 compared to the M/S one, and exists to mitigate potential ambiguity around L/R placement. Second, an adversarial loss term with feature matching, utilizing 5 con-

[3] https://github.com/Stability-AI/stable-audio-tools/
[4] https://stability-ai.github.io/stable-audio-2-demo/

	DiT	AE	CLAP	Total
Parameters	1.1B	157M	125M	1.3B

Table 1: Number of learnable parameters of our models.

volutional discriminators [22] with hyperparameters consistent with previous work [14], but with channel count scaled to give ≈4 times the parameter count. And third, the KL divergence loss term that is weighted by $\times 10^{-4}$.

2.2 Diffusion-transformer (DiT)

Instead of the widely used convolutional U-Net structure [5–7, 12], we employ a diffusion-transformer (DiT). This approach has seen notable success in other modalities [29], and has recently been applied to musical audio [30]. The used transformer (Figure 2) follows a standard structure with stacked blocks consisting of serially connected attention layers and gated multi-layer perceptrons (MLPs), with skip connections around each. We employ layer normalization at the input to both the attention layer and the MLP. The key and query inputs to the attention layer have rotary positional embedding [31] applied to the lower half of the embedding. Each transformer block also contains a cross-attention layer to incorporate conditioning. Linear mappings are used at the input and output of the transformer to translate from the autoencoder latent dimension to the embedding dimension of the transformer. We utilize efficient block-wise attention [32] and gradient checkpointing [33] to reduce the computational and memory impact of applying a transformer architecture over longer sequences. These techniques are crucial to viable training of model with this context length.

The DiT is conditioned by 3 signals: *text* enabling natural language control, *timing* enabling variable-length generation, and *timestep* signaling the current timestep of the diffusion process. Text CLAP embeddings are included via cross-attention. Timing conditioning [3, 14] is calculated using sinusoidal embeddings [34] and also included via cross-attention. Timing conditioning is also prepended before the transformer, along with a sinusoidal embedding describing the current timestep of the diffusion process.

2.3 Variable-length music generation

Given that the nature of long-form music entails varying lengths, our model also allows for variable-length music generation. We achieve this by generating content within a specified window length (e.g., 3m 10s or 4m 45s) and relying on the timing condition to fill the signal up to the length specified by the user. The model is trained to fill the rest of the signal with silence. To present variable-length audio outputs shorter than the window length to end-users, one can easily trim the appended silence. We adopt this strategy, as it has shown its effectiveness in previous work [14].

2.4 CLAP text encoder

We rely on a contrastive model trained from text-audio pairs, following the structure of CLAP [26]. It consists of a HTSAT-based [35] audio encoder with fusion and a RoBERTa-based [36] text encoder, both trained from scratch on our dataset with a language-audio contrastive loss. Following previous work [14], we use as text features the next-to-last hidden layer of the CLAP text encoder.

3. TRAINING SETUP

Training the model is a multi-stage process and was conducted on a cluster of NVIDIA A100 GPUs. Firstly, the autoencoder and CLAP model are trained. The CLAP model required approximately 3k GPU hours [5] and the autoencoder 16k GPU hours[4]. Secondly, the diffusion model is trained. To reach our target length of 4m 45s, we first pre-train the model for 70k GPU hours[4] on sequences corresponding to a maximum of 3m 10s of music. We then take the resulting model and fine-tune it on sequences of up to 4m 45s for a further 15k GPU hours[4]. Hence, the diffusion model is first pre-trained to generate 3m 10s music (referred to as the *pre-trained* model), and then fine-tuned to generate 4m 45s music (the *fully-trained* model).

All models are trained with the AdamW optimiser, with a base learning rate of $1e-5$ and a scheduler including exponential ramp-up and decay. We maintain an exponential moving average of the weights for improved inference. Weight decay, with a coefficient of 0.001, is also used. Parameter counts for the networks are given in Table 1, and the exact hyperparameters we used are detailed online[3].

The DiT is trained to predict a noise increment from noised ground-truth latents, following the v-objective [37]. We sample from our model using DPM-Solver++ [38] (100 steps), with classifier-free guidance [39] (scale of 7.0).

3.1 Training data and prompt preparation

Our dataset consists of 806,284 files (19,500h) containing music (66% or 94%) [6], sound effects (25% or 5%)[5], and instrument stems (9% or 1%)[5]. This audio is paired with text metadata that includes natural-language descriptions of the audio file's contents, as well as metadata such as BPM, genre, moods, and instruments for music tracks. All of our dataset (audio and metadata) is available online [7] for consultation. This data is used to train all three components of the system from scratch: the CLAP text encoder, the autoencoder and the DiT. The 285s (4m 45s) target temporal context encompasses over 90% of the dataset.

During the training of the CLAP text encoder and the DiT, we generate text prompts from the metadata by concatenating a random subset of the metadata as a string. This allows for specific properties to be specified during inference, while not requiring these properties to be present at all times. For half of the samples, we include the metadata-type (e.g., Instruments or Moods) and join them with a delimiting character (e.g., Instruments: Guitar, Drums, Bass Guitar|Moods: Uplifting, Energetic). For the other half, we do not include the metadata-type and join the

[5] GPU hours represent one hour of computation on a single GPU. The training process was distributed across multiple GPUs for efficency.

[6] Percentages: number of files or GBs of content, respectively.

[7] https://www.audiosparx.com/

properties with a comma (e.g., Guitar, Drums, Bass Guitar, Uplifting, Energetic). For metadata-types with a list of values, we shuffle the list. Hence, we perform a variety of random transformations of the resulting string, including two variants of delimiting character (“,” and “|”), shuffling orders and transforming between upper and lower case.

4. EXPERIMENTS

4.1 Quantitative evaluation

We evaluate a corpus of generated music using previously established metrics [14], as implemented in *stable-audio-metrics.* [8] Those include the Fréchet distance on OpenL3 embeddings [40], KL-divergence on PaSST tags [41], and distance in LAION-CLAP space [26, 42] [9] .

We set MusicGen-large-stereo (MusicGen) [8] as baseline, since it is the only publicly available model able to generate music at this length in stereo. This autoregressive model can generate long-form music of variable length due to its sequential (one-sample-at-a-time generation) sampling. However, note that MusicGen is not conditioned on semantic tokens that ensure long-term structural coherence, and it was not trained to generate such long contexts.

The prompts and ground-truth audio used for the quantitative study are from the Song Describer Dataset [43]. We select this benchmark, with 2m long music, because other benchmarks contain shorter music segments [4] and are inappropriate for long-form music evaluation. As vocal generation is not our focus and MusicGen is not trained for this task either, we opted to ensure a fair evaluation against MusicGen by curating a subset of 586 prompts that exclude vocals. [10] This subset, referred to as the Song Describer Dataset (no-singing), serves as our benchmark for comparison. We assess 2m generations to remain consistent with the ground-truth and also evaluate our models at their maximum generation length—which is 3m 10s for the pre-trained model or 4m 45s for the fully-trained one (Tables 2 and 3, respectively). For each model and length we study, we generate one render per prompt in our benchmark. This results in 586 generations per experiment.

Our model is first pre-trained to generate 3m 10s music (pre-trained model) and then fine-tuned to generate 4m 45s music (fully-trained model). Tables 2 and 3 show the quantitative results for both models and inference times. Comparing metrics between the pre-trained model and the fully-trained one shows no degradation, confirming the viability of extending context length via this mechanism. The proposed model scores better than MusicGen at all lengths while being significantly faster.

4.2 Qualitative evaluation

We evaluate the corpus of generated music qualitatively, with a listening test developed with webMUSHRA [44]. Mixed in with our generated music are generations from MusicGen and also ground-truth samples from the Song Describer Dataset (no-singing). Generations of our fully-trained model are included at both 4m 45s and 2m long, whilst ground-truth is only available at 2m. We selected two samples from each use case that were competitive for both models. For MusicGen it was difficult to find coherently structured music, possibly because it is not trained for long-form music generation. For our model, we found some outstanding generations that we selected for the test. Test material is available on our demo page.

Test subjects were asked to rate examples on a number of qualities including audio quality, text alignment, musical structure, musicality, and stereo correctness. We report mean opinion scores (MOS) in the following scale: *bad* (1), *poor* (2), *fair* (3), *good* (4), *excellent* (5). We observed that assessing stereo correctness posed a significant challenge for many users. To address this, we streamlined the evaluation by seeking for a binary response, correct or not, and report percentages of stereo correctness. All 26 test subjects used studio monitors or headphones, and self-identified as music producers or music researchers. In order to reduce test time and maximise subject engagement, we split the test into two parts. Each participant can choose one of the parts, or both, depending on their available time.

Results in Table 4 indicate that the generations from our system are comparable to the ground-truth in most aspects, and superior to the existing baseline. Our model obtains *good* (4) MOS scores across the board and stereo correctness scores higher than 95%, except for 2m long generations where its musical structure is *fair* (3). Differently from our quantitative results in Table 3, qualitative metrics show that 2m long generations are slightly worse than the 4m 45s generations (specially musical structure). We hypothesize that this could be due to the relative scarcity of full-structured music at this length in our dataset, since most music at this length might be repetitive loops. These results confirm that semantic tokens are not strictly essential for generating music with structure, as it can emerge through training with long contexts. Note, however, that the temporal context must be sufficiently long to obtain structured music generation. It was not until we scaled to longer temporal contexts (4m 45s), that we observed music with *good* strcuture, reflecting the inherent nature of the data. It is also noteworthy that the perceptual evaluation of structure yields to a wide diversity of responses, as indicated by the high standard deviations in Table 4. This highlights the challenge of evaluating subjective musical aspects. Finally, MusicGen achieves a stereo correctness rate of approximately 60%. This may be attributed to its tendency to generate mixes where instruments typically panned in the center (such as bass or kick) are instead panned to one side, creating an unnaturally wide mix that was identified as incorrect by the music producers and researchers participating in our test.

[8] https://github.com/Stability-AI/stable-audio-metrics

[9] https://github.com/LAION-AI/CLAP

[10] Prompts containing any of those words were removed: speech, speech synthesizer, hubbub, babble, singing, male, man, female, woman, child, kid, synthetic singing, choir, chant, mantra, rapping, humming, groan, grunt, vocal, vocalist, singer, voice, and acapella.

	channels/sr	output length	FD$_{openl3}$ ↓	KL$_{passt}$ ↓	CLAP$_{score}$ ↑	inference time
MusicGen-large-stereo [8]	2/32kHz	2m	204.03	0.49	0.28	6m 38s
Ours (pre-trained)	2/44.1kHz	2m[†]	78.70	0.36	0.39	8s
MusicGen-large-stereo [8]	2/32kHz	3m 10s	213.76	0.50	0.28	9m 32s
Ours (pre-trained)	2/44.1kHz	3m 10s	89.33	0.34	0.39	8s

Table 2: *Song Describer Dataset (no-singing subset):* results of the 3m 10s pre-trained model. [†]Our pre-trained model generates 3m 10s outputs, but during inference it can generate 2m outputs by relying on the timing conditioning. We trim audios to 2m (discarding the end silent part) for a fair quantitative evaluation against the state-of-the-art (see Section 2.3).

	channels/sr	output length	FD$_{openl3}$ ↓	KL$_{passt}$ ↓	CLAP$_{score}$ ↑	inference time
MusicGen-large-stereo [8]	2/32kHz	2m	204.03	0.49	0.28	6m 38s
Ours (fully-trained)	2/44.1kHz	2m[†]	79.09	0.35	0.40	13s
MusicGen-large-stereo [8]	2/32kHz	4m 45s	218.02	0.50	0.27	12m 53s
Ours (fully-trained)	2/44.1kHz	4m 45s	81.96	0.34	0.39	13s

Table 3: *Song Describer Dataset (no-singing subset):* results of the 4m 45s fully-trained model. [†]Our fully-trained model generates 4m 45s outputs, but during inference it can generate 2m outputs by relying on the timing conditioning. We trim audios to 2m (discarding the end silent part) for a fair quantitative evaluation against the state-of-the-art (see Section 2.3).

Results with the fully-trained model:	2m long			4m 45s long	
	Stable Audio 2	MusicGen-large-stereo	ground truth	Stable Audio 2	MusicGen-large-stereo
Audio Quality	4.0±0.6	2.8±0.8	4.6±0.4	4.5±0.4	2.8±0.8
Text Alignment	4.3±0.7	3.1±0.8	4.6±0.5	4.6±0.4	2.9±1.0
Structure	3.5±1.3	2.4±0.7	4.3±0.8	4.0±1.0	2.1±0.7
Musicality	4.0±0.8	2.7±0.9	4.6±0.5	4.3±0.7	2.6±0.7
Stereo correctness	96%	61%	96%	100%	57%

Table 4: *Qualitative results.* Top: mean opinion score ± standard deviation. Bottom: percentages.

	sampling rate	STFT distance ↓	MEL distance ↓	SI-SDR ↑	latent rate	latent (channels)
DAC [23]	44.1kHz	0.96	0.52	10.83	86 Hz	discrete
AudioGen [24]	48kHz	1.17	0.64	9.27	50 Hz	discrete
Encodec [8,22]	32kHz	1.82	1.12	5.33	50 Hz	discrete
AudioGen [24]	48kHz	1.10	0.64	8.82	100 Hz	continuous (32)
Stable Audio [14]	44.1kHz	1.19	0.67	8.62	43 Hz	continuous (64)
Ours	44.1kHz	1.19	0.71	7.14	21.5 Hz	continuous (64)

Table 5: *Autoencoder reconstructions on the Song Describer Dataset (all the dataset).* Although different autoencoders operate at various sampling rates, the evaluations are run at 44.1kHz bandwidth for a fair comparison. Sorted by latent rate.

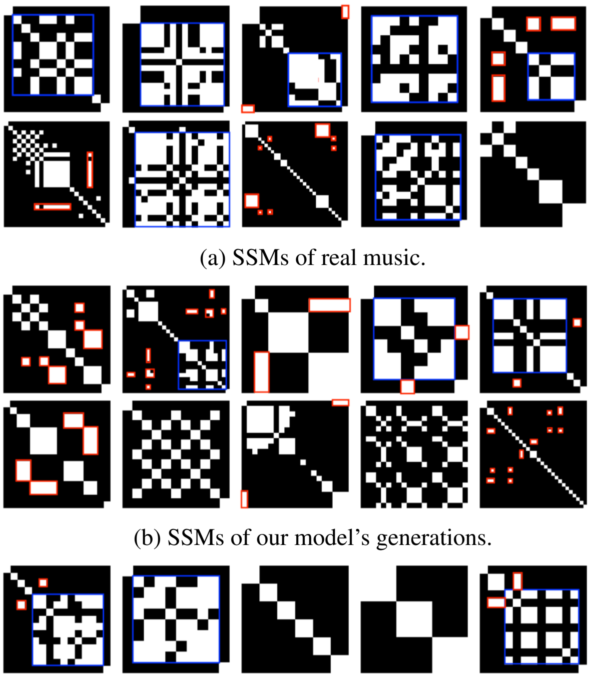
4.3 Autoencoder evaluation

We evaluate the audio reconstruction quality of our autoencoder in isolation, as it provides a quality ceiling for our system. We achieve this by comparing ground-truth and reconstructed audio via a number of established audio quality metrics [22, 23]: STFT distance, MEL distance and SI-SDR (as in AuraLoss library [28], with its default parameters). The reconstructed audio is obtained by encoding-decoding the ground-truth audio from the Song Describer Dataset (all the dataset, 706 tracks) through the autoencoder. As a comparison, we calculate the same metrics on a number of publicly available neural audio codecs including Encodec [22], DAC [23] and AudioGen [24]. Encodec and DAC are selected because are widely used for generative music modeling [4, 8, 10]. We select the Encodec 32kHz variant because our MusicGen baseline relies on it, and DAC 44.1kHz because its alternatives operate at 24kHz and 16kHz. Further, since our autoencoder relies on a continuous latent, we also compare against AudioGen, a state-of-the-art autoencoder with a continuous latent. Notably, AudioGen presents both continuous and discrete options, and we report both for completeness. All neural audio codecs are stereo, except DAC 44.1kHz and Encodec 32 kHz. In those cases, we independently project left and right channels and reconstruct from those.

The results in Table 5 show that the proposed autoencoder is comparable or marginally worse in raw reconstruction quality with respect to the other available baselines, whilst targeting a significantly larger amount (2x-5x) of temporal downsampling, and hence a lower latent rate. Our results are not strictly comparable against discrete neural audio codecs, but are included as a reference. For a qualitative assessment of our autoencoder's reconstruction quality, listen to some examples on our demo site.

4.4 Musical structure analysis

We explore the plausibility of the generated structures by visualizing the binary self-similarity matrices (SSMs) [45] of randomly chosen generated music against real music of the same genre. Real music is from the Free Music Archive (FMA) [46]. Similarly to real music, our model's generations can build structure with intricate shifts, including repetition of motives that were introduced at the first section. Red marks in Figure 4 show late sections that are similar to early sections. In MusicGen examples, early sections rarely repeat (e.g., see diagonal lines in Figure 4c) or music gets stuck in a middle/ending section loop (repetitive/loop sections are marked in blue in Figure 4). Note that our model's middle sections can also be repetitive, while still maintaining an intro/outro. We omit MusicGen's second row because most of its SSMs exhibit a similar behaviour.



(a) SSMs of real music.

(b) SSMs of our model's generations.

(c) SSMs of MusicGen-large-stereo generations.

Figure 4: Each column shows the SSMs of different genres (left to right): rock, pop, jazz, hip-hop, and classical.

4.5 Memorization analysis

Recent works [47,48] examined the potential of generative models to memorize training data, especially for repeated elements in the training set. Further, musicLM [4] conducted a memorization analysis to address concerns on the potential misappropriation of creative content. Adhering to principles of responsible model development, we also run a comprehensive study on memorization [4,47,48].

Considering the increased probability of memorizing repeated music within the dataset, we start by studying if our training set contains repeated data. We embed all our training data using the LAION-CLAP[8] audio encoder to select audios that are close in this space based on a manually set threshold. The threshold is set such that the selected audios correspond to exact replicas. With this process, we identify 5566 repeated audios in our training set.

We compare our model's generations against the training set in LAION-CLAP[8] space. Generations are from 5566 prompts within the repeated training data (in-distribution), and 586 prompts from the Song Describer Dataset (no-singing, out-of-distribution). We then identify the top-50 generated music that is closest to the training data and listen. We extensively listened to potential memorization candidates, and could not find memorization. We even selected additional outstanding generations, and could not find memorization. The most interesting memorization candidates, together with their closest training data, are online for listening on our demo page.

4.6 Additional creative capabilities

Besides text-conditioned long-form music generation, our model exhibits capabilities in other applications. While we do not conduct a thorough evaluation of these, we briefly describe those and showcase examples on our demo page.

Audio-to-audio — With diffusion models is possible to perform some degree of style-transfer by initializing the noise with audio during sampling [15,49]. This capability can be used to modify the aesthetics of an existing recording based on a given text prompt, whilst maintaining the reference audio's structure (e.g., a beatbox recording could be style-transfered to produce realistic-sounding drums). As a result, our model can be influenced by not only text prompts but also audio inputs, enhancing its controllability and expressiveness. We noted that when initialized with voice recordings (such as beatbox or onomatopoeias), there is a sensation of control akin to an instrument. Examples of audio-to-audio are on our demo page.

Vocal music — The training dataset contains a subset of music with vocals. Our focus is on the generation of instrumental music, so we do not provide any conditioning based on lyrics. As a result, when the model is prompted for vocals, the model's generations contains vocal-like melodies without intelligible words. Whilst not a substitute for intelligible vocals, these sounds have an artistic and textural value of their own. Examples are given on our demo page.

Short-form audio generation — The training set does not exclusively contain long-form music. It also contains shorter sounds like sound effects or instrument samples. As a consequence, our model is also capable of producing such sounds when prompted appropriately. Examples of short-form audio generations are also on our demo page.

5. CONCLUSIONS

We presented an approach to building a text-conditioned music generation model, operating at long enough context lengths to encompass full musical tracks. To achieve this we train an autoencoder which compresses significantly more in the temporal dimension than previous work. We model full musical tracks represented in the latent space of this autoencoder via a diffusion approach, utilizing a diffusion-transformer. We evaluate the trained model via qualitative and quantitative tests, and show that it is able to produce coherent music with state-of-the-art results over the target temporal context of 4m 45s.

6. ETHICS STATEMENT

Our technology represents an advancement towards aiding humans in music production tasks, facilitating the creation of variable-length, long-form stereo music based on textual input. This advancement greatly enhances the creative repertoire available to artists and content creators. However, despite its numerous advantages, it also brings inherent risks. A key concern lies in the potential reflection of biases inherent in the training data. Additionally, the nuanced context embedded within music emphasizes the necessity for careful consideration and collaboration with stakeholders. In light of these concerns, we are dedicated to ongoing research and collaboration with those stakeholders, including artists and data providers, to navigate this new terrain responsibly. Adhering to best practices in responsible model development, we conducted an exhaustive study on memorization. Employing our methodology, we found no instances of memorization.

7. REFERENCES

[1] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv*, 2016.

[2] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," *arXiv*, 2016.

[3] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv*, 2020.

[4] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank, "MusicLM: Generating music from text," *arXiv*, 2023.

[5] F. Schneider, Z. Jin, and B. Schölkopf, "Moûsai: Text-to-music generation with long-context latent diffusion," *arXiv*, 2023.

[6] Q. Huang, D. S. Park, T. Wang, T. I. Denk, A. Ly, N. Chen, Z. Zhang, Z. Zhang, J. Yu, C. Frank, J. Engel, Q. V. Le, W. Chan, Z. Chen, and W. Han, "Noise2music: Text-conditioned music generation with diffusion models," *arXiv*, 2023.

[7] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. Mandic, W. Wang, and M. D. Plumbley, "AudioLDM: Text-to-audio generation with latent diffusion models," *arXiv*, 2023.

[8] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, "Simple and controllable music generation," *arXiv*, 2023.

[9] H. F. Garcia, P. Seetharaman, R. Kumar, and B. Pardo, "VampNet: Music generation via masked acoustic token modeling," *arXiv*, 2023.

[10] A. Ziv, I. Gat, G. L. Lan, T. Remez, F. Kreuk, A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "Masked audio generation using a single non-autoregressive transformer," *arXiv*, 2024.

[11] M. W. Y. Lam, Q. Tian, T. Li, Z. Yin, S. Feng, M. Tu, Y. Ji, R. Xia, M. Ma, X. Song, J. Chen, Y. Wang, and Y. Wang, "Efficient neural music generation," *arXiv*, 2023.

[12] H. Liu, Q. Tian, Y. Yuan, X. Liu, X. Mei, Q. Kong, Y. Wang, W. Wang, Y. Wang, and M. D. Plumbley, "AudioLDM 2: Learning holistic audio generation with self-supervised pretraining," *arXiv*, 2023.

[13] G. Cideron, S. Girgin, M. Verzetti, D. Vincent, M. Kastelic, Z. Borsos, B. McWilliams, V. Ungureanu, O. Bachem, O. Pietquin, M. Geist, L. Hussenot, N. Zeghidour, and A. Agostinelli, "MusicRL: Aligning music generation to human preferences," *arXiv*, 2024.

[14] Z. Evans, C. Carr, J. Taylor, S. Hawley, and J. Pons, "Fast timing-conditioned latent audio diffusion," *arXiv*, 2024.

[15] S. Pascual, G. Bhattacharya, C. Yeh, J. Pons, and J. Serrà, "Full-band general audio synthesis with score-based diffusion," *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2023.

[16] J. Parker, J. Spijkervet, K. Kosta, F. Yesiler, B. Kuznetsov, J.-C. Wang, M. Avent, J. Chen, and D. Le, "StemGen: A music generation model that listens," *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2024.

[17] Z. Novack, J. McAuley, T. Berg-Kirkpatrick, and N. J. Bryan, "Ditto: Diffusion inference-time t-optimization for music generation," *arXiv*, 2024.

[18] G. Mariani, I. Tallini, E. Postolache, M. Mancusi, L. Cosmo, and E. Rodolà, "Multi-source diffusion models for simultaneous music generation and separation," *arXiv*, 2023.

[19] M. Pasini, M. Grachten, and S. Lattner, "Bass accompaniment generation via latent diffusion," *arXiv*, 2024.

[20] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, D. Grangier, M. Tagliasacchi *et al.*, "AudioLM: a language modeling approach to audio generation," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 2023.

[21] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 2021.

[22] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *arXiv*, 2022.

[23] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[24] AudiogenAI, "Audiogenai/agc: Audiogen codec." [Online]. Available: https://github.com/AudiogenAI/agc

[25] B. Elizalde, S. Deshmukh, M. A. Ismail, and H. Wang, "Clap: Learning audio concepts from natural language supervision," *arXiv*, 2022.

[26] Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, and S. Dubnov, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2023.

[27] L. Ziyin, T. Hartwig, and M. Ueda, "Neural networks fail to learn periodic functions and how to fix it," *arXiv*, 2020.

[28] C. J. Steinmetz and J. D. Reiss, "auraloss: Audio focused loss functions in PyTorch," in *Digital Music Research Network One-day Workshop (DMRN+15)*, 2020.

[29] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proc. of the IEEE/CVF Int. Conf. on Comp. Vision (ICCV)*, 2023.

[30] M. Levy, B. Di Giorgi, F. Weers, A. Katharopoulos, and T. Nickson, "Controllable music production with diffusion models and guidance gradients," *arXiv*, 2023.

[31] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *arXiv*, 2023.

[32] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," *arXiv*, 2022.

[33] T. Chen, B. Xu, C. Zhang, and C. Guestrin, "Training deep nets with sublinear memory cost," *arXiv*, 2016.

[34] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *arXiv*, 2020.

[35] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-Kirkpatrick, and S. Dubnov, "HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection," in *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2022.

[36] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv*, 2019.

[37] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," *arXiv*, 2022.

[38] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "DPM-solver++: Fast solver for guided sampling of diffusion probabilistic models," *arXiv*, 2022.

[39] S. Lin, B. Liu, J. Li, and X. Yang, "Common diffusion noise schedules and sample steps are flawed," *IEEE/CVF Winter Conf. on Applications of Comp. Vision*, 2024.

[40] A. L. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," *IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2019.

[41] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *Conf. of the Int. Speech Comm. Assoc. (INTERSPEECH)*, 2022.

[42] R. Huang, J. Huang, D. Yang, Y. Ren, L. Liu, M. Li, Z. Ye, J. Liu, X. Yin, and Z. Zhao, "Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models," *arXiv*, 2023.

[43] I. Manco, B. Weck, S. Doh, M. Won, Y. Zhang, D. Bogdanov, Y. Wu, K. Chen, P. Tovstogan, E. Benetos, E. Quinton, G. Fazekas, and J. Nam, "The Song Describer Dataset: a corpus of audio captions for music-and-language evaluation," *arXiv*, 2023.

[44] M. Schoeffler, S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, "web-MUSHRA—a comprehensive framework for web-based listening tests," *Journal of Open Research Software*, 2018.

[45] J. Serra, M. Müller, P. Grosche, and J. L. Arcos, "Unsupervised music structure annotation by time series structure features and segment similarity," *IEEE Trans. on Multimedia*, 2014.

[46] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," *arXiv*, 2016.

[47] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, B. Balle, D. Ippolito, and E. Wallace, “Extracting training data from diffusion models,” in *USENIX Security Symposium*, 2023.

[48] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel *et al.*, “Scaling rectified flow transformers for high-resolution image synthesis,” *arXiv*, 2024.

[49] S. Rouard and G. Hadjeres, “CRASH: Raw audio score-based generative modeling for controllable high-resolution drum sound synthesis,” *arXiv*, 2021.