Fast Timing-Conditioned Latent Audio Diffusion

Zach Evans [1]   **CJ Carr** [1]   **Josiah Taylor** [1]   **Scott H. Hawley** [2]   **Jordi Pons** [1]

Abstract

Generating long-form 44.1kHz stereo audio from text prompts can be computationally demanding. Further, most previous works do not tackle that music and sound effects naturally vary in their duration. Our research focuses on the efficient generation of long-form, variable-length stereo music and sounds at 44.1kHz using text prompts with a generative model. Stable Audio is based on latent diffusion, with its latent defined by a fully-convolutional variational autoencoder. It is conditioned on text prompts as well as timing embeddings, allowing for fine control over both the content and length of the generated music and sounds. Stable Audio is capable of rendering stereo signals of up to 95 sec at 44.1kHz in 8 sec on an A100 GPU. Despite its compute efficiency and fast inference, it is one of the best in two public text-to-music and -audio benchmarks and, differently from state-of-the-art models, can generate music with structure and stereo sounds.

1. Introduction

The introduction of diffusion-based generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020) has lead to rapid improvements in the quality and controllability of generated images (Podell et al., 2023), video (Blattmann et al., 2023), and audio (Rouard & Hadjeres, 2021; Liu et al., 2023a).

One challenge is that diffusion models working within the raw signal space tend to be computationally demanding during both training and inference. Diffusion models working in the latent space of a pre-trained autoencoder, termed "latent diffusion models" (Rombach et al., 2022), are significantly more compute efficient. Working with a heavily downsampled latent representation of audio allows for much faster inference times compared to raw audio, and also allows generating long-form audio (e.g., 95 sec).

Another challenge with audio diffusion models is that those are usually trained to generate fixed-size outputs (Huang et al., 2023a), e.g., a model trained on 30 sec audio chunks will generate 30 sec outputs. This is an issue when training on and trying to generate audio of varying lengths, as is the case when generating full songs or sound effects. Hence audio diffusion models are commonly trained on randomly cropped chunks from longer audios, cropped or padded to fit the diffusion model's training length. With music, e.g., this causes the model to generate arbitrary sections of a song, which may start or end in the middle of a musical phrase.

Stable Audio is based on a latent diffusion model for audio conditioned on a text prompt as well as timing embeddings, allowing for control over the content and length of the generated music and sound effects. This additional timing conditioning allows us to generate audio of a specified (variable) length up to the training window length. Due to the compute efficient nature of latent diffusion modeling, it can generate long-form content in short inference times. It can render up to 95 sec (our training window length) of stereo audio at 44.1kHz in 8 sec on an A100 GPU (40GB VRAM).

The commonly used metrics for generative audio are designed to evaluate short-form mono signals at 16kHz (Kilgour et al., 2018). Yet, our work focuses on generating long-form full-band stereo signals. We propose: (i) a Fréchet Distance based on OpenL3 embeddings (Cramer et al., 2019) to evaluate the plausibility of the generated long-form full-band stereo signals, (ii) a Kullback-Leibler divergence to evaluate the semantic correspondence between lengthy generated and reference audios up to 32kHz, and (iii) a CLAP score to evaluate how long-form full-band stereo audios adhere to the given text prompt. We also conduct a qualitative study, assessing audio quality and text alignment, while also pioneering the assessment of musicality, stereo correctness, and musical structure. We show that Stable Audio can obtain state-of-the-art results on long-form full-band stereo music and sound effects generation from text and timing inputs. We also show that, differently from previous works, Stable Audio is also capable to generate structured music (with intro, development, outro) and stereo sound effects.

Code to reproduce our model/metrics and demos is online[1].

---

[1]Stability AI. [2]Belmont University, work done while at Stability AI. Correspondence to: Zach Evans <zach@stability.ai>.

---

[1]Model: https://github.com/Stability-AI/stable-audio-tools. Metrics: https://github.com/Stability-AI/stable-audio-metrics.

2. Related Work

Autoregressive models — WaveNet (Oord et al., 2016) autoregressively models quantized audio samples, but is slow during inference because it operates with waveforms. Recent autoregressive models addressed this by operating on a quantized latent space, enabling faster processing. Jukebox (Dhariwal et al., 2020) relies on a multi-scale approach to encode music into a sequence of quantized latents and subsequently models them using autoregressive transformers. Recent work such as MusicLM (Agostinelli et al., 2023) and MusicGen (Copet et al., 2023) utilize a similar approach and also autoregressively model quantized latent sequences. However, unlike Jukebox, such models are conditioned on text prompts rather than on artist, genre, and/or lyrics. Autoregressive models similar to MusicLM (AudioLM) and MusicGen (AudioGen) have also been used for sound synthesis (Borsos et al., 2023; Kreuk et al., 2022) and for generating music accompaniments from singing (Donahue et al., 2023). Our work is not based on autoregressive modeling.

Non-autoregressive models — Parallel WaveNet (Oord et al., 2018) and adversarial audio synthesis (Donahue et al., 2018; Pasini & Schlüter, 2022) were developed to tackle the computational inefficiencies inherent in autoregressive modeling. Recent works like VampNet (Garcia et al., 2023), StemGen (Parker et al., 2024) and MAGNeT (Ziv et al., 2024) are based on masked token modeling (Chang et al., 2022). These are for creating musical variations, generating additional stems for a given song, and to efficiently synthesize music and sounds, respectively. Flow-matching generative modeling (Vyas et al., 2023) was also recently introduced for speech and sounds synthesis. Our work is not based on any of the non-autoregressive models above.

End-to-end diffusion models — CRASH (Rouard & Hadjeres, 2021) was proposed for unconditional drums synthesis, DAG (Pascual et al., 2023) for class-conditional sounds synthesis, Noise2Music (Huang et al., 2023a) for text-conditional music synthesis, and Mariani et al. (2023) built an end-to-end diffusion model capable of both music synthesis and source separation. Our work is also based on diffusion, albeit not in an end-to-end fashion. Rather, it involves latent diffusion due to its computational efficiency.

Spectrogram diffusion models — Riffusion (Forsgren & Martiros, 2022) fine-tuned Stable Diffusion to generate spectrograms from text prompts, Hawthorne et al. (2022) addressed MIDI-to-spectrogram generation, and CQT-Diff (Moliner et al., 2023) relied on CQT spectrograms for bandwidth extension, inpainting, and declipping. An additional step is required to render waveforms from magnitude spectrograms. Our work is also based on diffusion, albeit it does not rely on spectrogram-based synthesis.

Demo: https://stability-ai.github.io/stable-audio-demo.

Latent diffusion models — Moûsai (Schneider et al., 2023) and AudioLDM (Liu et al., 2023a) pioneered using latent diffusion for text-to-music and -audio. Their main difference being that Moûsai decodes latents onto waveforms through a diffusion decoder, while AudioLDM decodes latents onto spectrograms which are then inverted to waveforms with HiFi-GAN (Kong et al., 2020). AudioLDM2 (Liu et al., 2023b) extends AudioLDM to also synthesize speech by using a shared representation for music, audio, and speech to condition the latent diffusion model. JEN-1 (Li et al., 2023) is an *omnidirectional* latent diffusion model trained in a multitask fashion. JEN-1 Composer (Yao et al., 2023) is its extension for multi-track music generation. Levy et al. (2023) explored sampling-time guidance for both end-to-end and latent diffusion models. All previous works constrain the latent to be normalized, often with a variational autoencoder (VAE). The exceptions being JEN-1, which runs over a dimensionality reduced latent that is normalized based on the mean and covariance, and Moûsai that simply uses a tanh. Our work is also based on latent diffusion, and we normalize latents by using a VAE. Appendix D includes further discussion on related latent diffusion models.

High sampling rate and stereo generation — Moûsai and JEN-1 generate 48kHz stereo music. AudioLDM2 can generate 48kHz mono music. Levy et al. (2023) generates 44.1kHz stereo music. No other prior works generate music up to the standard specifications of commercial music (44.1kHz stereo). DAG and AudioLDM2 generate 48kHz mono sounds, and we are not aware of prior works tackling stereo sound synthesis. Our work focuses on generating 44.1kHz stereo music and sounds from text prompts.

Text embeddings — CLAP (Wu et al., 2023) and T5-like (Raffel et al., 2020; Ghosal et al., 2023) text embeddings are commonly used because of their open-source nature. CLAP relies on a contrastive (multimodal) language-audio pretraining, and T5 is a large language model. Further, MusicLM uses MuLan (Huang et al., 2022), that is also based on contrastive language-audio pretraining but on their private dataset. Our work relies on a CLAP-based model trained in a contrastive language-audio fashion on our dataset.

Fast generation of variable-length, long-form audio — Autoregressive models can generate long-form audio of variable length due to their sequential (one-sample-at-a-time generation) nature, but are slow at inference time. Previous non-autoregressive models were trained to generate up to 20 sec long music (Parker et al., 2024). Previous end-to-end and latent diffusion models were trained to generate up to 30 sec long music (Huang et al., 2023a; Levy et al., 2023; Lam et al., 2024), with the exception of Moûsai that was trained to generate 44 sec. Hence, previous works are either slow at inference time (autoregressive models) or cannot generate variable-length, long-form audio (the rest). Our

work relies on latent diffusion to generate long-form (up to 95 sec), variable-length (controlled by the timing condition) stereo signals at 44.1kHz in 8 sec on an A100 GPU (40GB VRAM).

Timing conditioning — The use of learned embeddings to condition music generation models on timing information was introduced by Jukebox (Dhariwal et al., 2020), an autoregressive model conditioned with timing information on: (i) song duration, (ii) starting time of the training/generated audio sample within the song, and (iii) how much fraction of the song has elapsed. We are not aware of previous works using timing conditioning for conditioning (latent) diffusion models. Our work employs timing conditioning to control the length of the generations, enabling our latent diffusion models to generate variable-length outputs.

Evaluation metrics — The commonly used quantitative audio metrics were developed for evaluating short-form mono audio generations at 16kHz (Kilgour et al., 2018; Copet et al., 2023). Yet, our work focuses on generating long-form full-band stereo signals. Only Pascual et al. (2023) explored quantitative metrics for evaluating full-band audio, although their focus was short-form mono signals. Our work explores new quantitative metrics to evaluate long-form full-band stereo generations. Qualitative metrics assessing audio quality and text alignment are also prevalent in the literature (Dong et al., 2023; Copet et al., 2023; Ziv et al., 2024). Our work also explores additional qualitative metrics to evaluate musicality, stereo correctness, and musical structure.

Multitask generative modeling — While generative models have traditionally focused on specific tasks like speech, music or sound synthesis, recent works showed success in addressing all these tasks simultaneously (Yang et al., 2023; Liu et al., 2023b). Our work relies on one model to generate both music and sound (not speech) given a text prompt.

3. Architecture

Stable Audio is based on a latent diffusion model consisting of a variational autoencoder (Section 3.1), a conditioning signal (Section 3.2), and a diffusion model (Section 3.3).

3.1. Variational autoencoder (VAE)

The VAE (Kingma & Welling, 2013) compresses 44.1kHz stereo audio into an invertible (lossy) latent encoding that enables faster generation and training time compared to working with raw audio samples. To allow for arbitrary-length audio encoding and decoding, we use a fully-convolutional architecture (133M parameters) that follows the Descript Audio Codec (Kumar et al., 2023) encoder and decoder (without the quantizer). We found that the Snake activations (Ziyin et al., 2020) in the Descript Audio Codec architecture improved audio reconstruction at high compression ratios

compared to alternatives such as EnCodec (Défossez et al., 2022), at the expense of increased VRAM consumption. The VAE is trained from scratch on our dataset and downsamples the input stereo audio sequence by a factor of 1024, with the resulting latent sequence having a channel dimension of 64 (i.e., maps a $2 \times L$ input into $64 \times L/1024$ latent). This results in an overall data compression ratio of 32.

3.2. Conditioning

Text encoder — To condition on text prompts, we use a CLAP text encoder trained from scratch on our dataset. We use the actual setup recommended by the CLAP authors: (i) a HTSAT-based audio encoder with fusion having 31M parameters, and (ii) a RoBERTa-based text encoder of 110M parameters, both trained with a language-audio contrastive loss. We use CLAP embeddings (instead of the also commonly used T5 embeddings) because its multimodal nature (language-audio) allows the text features to contain some information about the relationships between words and audio. Further, in Section 6.2 we empirically note that the CLAP embeddings trained from scratch on our dataset can outperform the open-source CLAP and T5 embeddings. As shown by NovelAI (2022) when using CLIP (Radford et al., 2021) text features for Stable Diffusion (Rombach et al., 2022), the text features in the next-to-last layer of the text encoder can provide a better conditioning signal than the text features from the final layer. Because of this, we use the text features from the next-to-last hidden layer of the CLAP text encoder. These text features are provided to the diffusion U-Net through cross-attention layers.

Timing embeddings — We calculate two properties when gathering a chunk of audio from our training data: the second from which the chunk starts (termed *seconds_start*) and the overall number of seconds in the original audio file (termed *seconds_total*), see Figure 2. For example, if we take a 95 sec chunk from an 180 sec audio file with the chunk starting at 14 sec, then *seconds_start* is 14 and *seconds_total* is 180 (see Figure 2, Left). These values are then translated into per-second discrete learned embeddings[2] and concatenated along the sequence dimension with the text features from the prompt conditioning before being passed into the U-Net's cross-attention layers. For training with audio files shorter than the training window (see Figure 2, Right), we pad with silence up to the training window length. During inference, *seconds_start* and *seconds_total* are also provided as conditioning, allowing the user to specify the overall length of the output audio. For example, given our 95 sec model, setting *seconds_start* to 0 and *seconds_total* to 30 will create an output with 30 sec of audio followed by 65 sec of silence. This method allows the user generating variable-length music and sound effects.

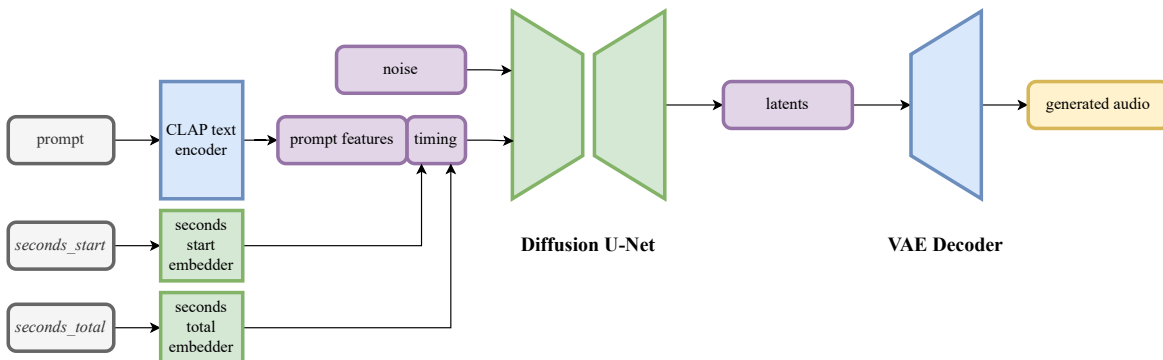[2]We have a learnt, continuous timing embedding per second.

Figure 1. Stable Audio. Blue: frozen pre-trained models. Green: parameters learnt during diffusion training. Purple: signals of interest.
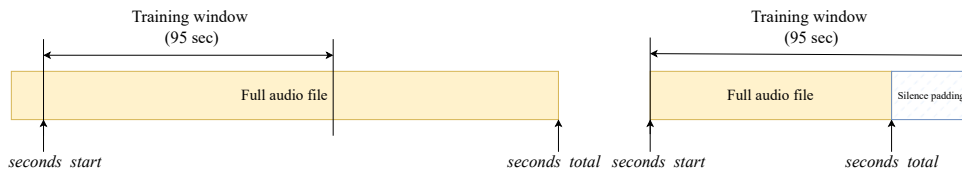

Figure 2. Timing embeddings examples. Left: Audio file longer than training window. *Right*: Audio file shorter than training window.

3.3. Diffusion model

Based on a U-Net (907M parameters) inspired by Moûsai's architecture (Schneider et al., 2023), it consists of 4 levels of symmetrical downsampling encoder blocks and upsampling decoder blocks, with skip connections between the encoder and decoder blocks providing a residual path at the same resolution. The 4 levels have channel counts of 1024, 1024, 1024, and 1280, and downsample by factors of 1 (no downsampling), 2, 2, and 4 respectively. After the final encoder block, there is a 1280-channel bottleneck block. Each block consists of 2 convolutional residual layers followed by a series of self-attention and cross-attention layers. Each encoder or decoder block has three of these attention layers, except for those in the first U-Net level, which only have one. We rely on a fast and memory-efficient attention implementation (Dao et al., 2022), to allow the model to scale more efficiently to longer sequence lengths. The diffusion timestep conditioning is passed in through FiLM layers (Perez et al., 2017) to modulate the model activations based on the noise level. The prompt and timing conditioning information is passed in to the model through cross-attention layers. Further implementation details are in Appendix F.

3.4. Inference

Our sampling strategy during inference is based on the DPM-Solver++ (Lu et al., 2022), and we use classifier-free guidance (with a scale of 6) as proposed by Lin et al. (2024). We use 100 diffusion steps during inference, see Appendix A to know more on how the number of steps was chosen. Stable Audio is designed for variable-length, long-form music and sound generation. This is achieved by generating content within a specified window length (95 sec), and relying on

the timing condition to fill the signal up to the length specified by the user and fill the rest with silence. To present variable-length audios (shorter than window length) to the end user, one can simply trim the silence. In Section 6.3 we note that the timing conditioning is very reliable, showing the robustness of the proposed silence-trimming strategy.

4. Training

4.1. Dataset

Our dataset consists of 806,284 audios (19,500 hours) containing music (66% or 94%)[3], sound effects (25% or 5%)[3], and instrument stems (9% or 1%)[3], with the corresponding text metadata from the stock music provider AudioSparx. Our dataset (audio and metadata) is online [4] for consultation.

4.2. Variational autoencoder (VAE)

It was trained using automatic mixed precision for 1.1M steps with an effective batch size of 256 on 16 A100 GPUs. After 460,000 steps the encoder was frozen and the decoder was fine-tuned for an additional 640,000 steps. To ensure a consistent stereo reconstruction, we use a multi-resolution sum and difference STFT loss designed for stereo signals (Steinmetz et al., 2020). To that end, we apply A-weighting (Fletcher & Munson, 2005) before the STFT and use window lengths of 2048, 1024, 512, 256, 128, 64, and 32. We also employ adversarial and feature matching losses using a multi-scale STFT discriminator modified to accept stereo audio (Défossez et al., 2022). The discriminators (set with

---

[3] Percentages: number of files or GBs of content, respectively.
[4] https://www.audiosparx.com

2048, 1024, 512, 256, and 128 STFT window lengths) use a complex STFT representation of the real and reconstructed audio, and a patch-based discriminative objective using the hinge loss (Défossez et al., 2022). Each loss is weighted as follows: 1.0 for spectral losses, 0.1 for adversarial losses, 5.0 for the feature matching loss, and 1e-4 for the KL loss.

4.3. Text encoder

The CLAP model was trained for 100 epochs on our dataset from scratch, with an effective batch size of 6,144 with 64 A100 GPUs. We use the setup recommended by CLAP authors and train it with a language-audio contrastive loss.

4.4. Diffusion model

It was trained using exponential moving average and automatic mixed precision for 640,000 steps on 64 A100 GPUs with an effective batch size of 256. The audio was resampled to 44.1kHz and sliced to 4,194,304 samples (95.1 sec). Files longer than this length were cropped from a random starting point, while shorter files were padded at the end with silence. We implemented a v-objective (Salimans & Ho, 2022) with a cosine noise schedule and continuous denoising timesteps. We apply dropout (10%) to the conditioning signals to be able to use classifier-free guidance. The text encoder is frozen while training the diffusion model.

4.5. Prompt preparation

Each audio file in our dataset is accompanied by text metadata describing the audio file. This text metadata includes natural-language descriptions of the audio file's contents, as well as domain-specific metadata such as BPM, genre, moods, and instruments for music tracks. During the training of the text encoder and the diffusion model, we generate text prompts from this 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 the | 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 properties with a comma (e.g. *Guitar, Drums, Bass Guitar, Uplifting, Energetic*). For metadata-types with a list of values, we shuffle the list.

5. Methodology

5.1. Quantitative metrics

FD_{openl3} — The Fréchet Distance (FD) is utilized to evaluate the similarity between the statistics of a generated audio set and a reference audio set in a feature space. A

low Fréchet Distance implies that the generated audio is plausible and closely matches the reference audio (Kilgour et al., 2018; Copet et al., 2023). While most previous works project the audio into the VGGish feature space (Hershey et al., 2017), we propose projecting it into the Openl3[5] feature space (Cramer et al., 2019). Importantly, Openl3 accepts signals of up to 48kHz while VGGish operates at 16kHz. With this modification, our FD is not limited to evaluate downsampled 16kHz audio but it can evaluate the full bandwidth of the generated audios. Since we focus on generating 44.1kHz audio, we resample all the evaluation audios to 44.1kHz. Finally, we also extend the FD to evaluate stereo signals. To that end, we project left- and right-channel audios into Openl3 features independently, and concatenate them to obtain the stereo features. If the evaluation audio is mono, we concatenate copied Openl3 (mono) features to obtain the desired stereo features. Hence, we propose a novel FD_{openl3} metric to study the plausibility of the generated variable-length, full-band stereo signals.

KL_{passt} — We use PaSST, a state-of-the-art audio tagger trained on AudioSet (Koutini et al., 2022), to compute the Kullback–Leibler (KL) divergence over the probabilities of the labels between the generated and the reference audio (Copet et al., 2023). The generated audio is expected to share similar semantics (tags) with the reference audio when the KL is low. While most previous works focus on generating short snippets, our work focuses on generating long-form audio. For this reason, we modify the KL to evaluate audios of varying and longer lengths. This adaptation involves segmenting the audio into overlapping analysis windows[6]. Subsequently, we calculate the mean (across windows) of the generated logits and then apply a softmax. Finally, PaSST operates at 32kHz. To evaluate our 44.1kHz models, we resample all the evaluation audios from 44.1kHz to 32kHz. Hence, we propose a novel KL_{passt} metric capable to evaluate the semantic correspondence between lengthy generated and reference audios up to 32kHz.

CLAP_{score} — The cosine similarity is computed between the CLAP_{LAION} text embedding of the given text prompt and the CLAP_{LAION} audio embedding of the generated audio (Wu et al., 2023; Huang et al., 2023b). A high CLAP_{score} denotes that the generated audio adheres to the given text prompt. Differently from previous works, that evaluate 10 sec inputs, we use the 'feature fusion' variant of CLAP_{LAION} to handle longer audios. It is based on 'fusing' (concatenating) inputs at various time-scales: a global input (downsampled to be of 10 sec) is concatenated to 3 random crops (of

---

[5]The Openl3 settings we use: mel256 input, 44.1kHz, 'music' or 'env' content type depending if we evaluate music or audio, embedding size of 512, and hop size of 0.5 sec.

[6]PaSST model was originally trained with 10 sec inputs, and we utilize an analysis window of 10 sec (to match PaSST training) with a 5 sec overlap (50% overlap, for compute efficiency).

10 sec) from the first, middle, and last parts of the audio. CLAP$_{\text{LAION}}$ audio embeddings are computed from 48kHz audio. To evaluate our 44.1kHz models, we resample all the evaluation audios from 44.1kHz to 48kHz. Hence, we propose a novel CLAP$_{score}$ to evaluate how 48kHz audios longer than 10 sec adhere to a given text prompt.

In short, we adapted established metrics to assess the more realistic use case of long-form full-band stereo generations. All quantitative metrics can deal with variable-length inputs.

5.2. Qualitative metrics

Audio quality — We evaluate whether the generated audio is of low-fidelity with artifacts or high-fidelity.

Text alignment — We evaluate how the generated audio adheres to the given text prompt.

Musicality (music only) — We evaluate the capacity of the model to articulate melodies and harmonies.

Stereo correctness (stereo only) — We evaluate the appropriateness of the generated spatial image.

Musical structure (music only) — We evaluate if the generated song contains intro, development, and/or outro.

We collect human ratings for the metrics above and report mean opinion scores for audio quality, text alignment, and musicality in the following scale: bad (0), poor (1), fair (2), good (3) and excellent (4). 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: either stereo correctness or not. Similarly, we adopted a binary approach for evaluating musical structure. We ask users to determine whether the generated music exhibits some common structural elements of music (intro, development, outro) or not. For those binary responses (stereo correctness and musical structure) we report percentages. Note that musicality and musical structure are only evaluated for music signals. For non-music (audio) signals we evaluate audio quality, text alignment and stereo correctness. Also note that stereo correctness is only evaluated for stereo signals. We relied on webMUSHRA (Schoeffler et al., 2018) to run our perceptual experiments. We are not aware of previous works that qualitatively assess musicality, stereo correctness, and/or musical structure.

5.3. Evaluation data

Quantitative experiments — We rely on the standard MusicCaps (Agostinelli et al., 2023) and AudioCaps (Kim et al., 2019) benchmarks. MusicCaps contains 5,521 music segments from YouTube, each with 1 caption (5,434 audios were available for download). AudioCaps test set contains

979 audio segments from YouTube, each with several captions (881 audios were available for download, and it includes 4,875 captions). For every model to evaluate, we generate an audio per caption. This results in 5,521 generations for the MusicCaps evaluations and 4,875 generations for the AudioCaps ones. While these benchmarks are not typically used for evaluating full-band stereo signals, the original data is predominantly stereo and full-band (Appendix B). We rely on the original data resampled to 44.1kHz to meet the target bandwidth of Stable Audio. Finally, since the standard MusicCaps and AudioCaps segments are of 10 sec, we also looked into the full-length audios to consider variable-length long-form evaluation content. Yet, captions do not hold consistently throughout the whole (long) audio, as they only accurately represent the intended 10 sec segment. As a result, reference audios are of 10 sec while generated audios range from 10 to 95 sec (Tables 1 and 2). Hence, in addition to modifying the established metrics to evaluate full-band stereo generations, it was also crucial to adapt the standard datasets to align with our evaluation criteria.

Qualitative experiments — Prompts for qualitative evaluation were randomly picked from MusicCaps and AudioCaps. We avoided prompts including "low quality" (or similar) to focus on high-fidelity synthesis, avoided ambient music because users found challenging to evaluate musicality, and avoided speech-related prompts since it is not our focus.

5.4. Baselines

Direct comparisons with some models (e.g., Moûsai or JEN1) is infeasible as their weights are not accessible. For this reason, we benchmark against AudioLDM2, MusicGen, and AudioGen. These are state-of-the-art open-source models representative of the current literature: latent diffusion models (AudioLDM2) or autoregressive models (MusicGen, AudioGen), that can be stereo (MusicGen-stereo) or mono, and at various sampling rates (see Table 1 and 2). The AudioLDM2 variants we evaluate are: 'AudioLDM2-48kHz' that was trained to generate full-band mono sounds and music, 'AudioLDM2-large' to generate 16kHz mono sounds and music, and 'AudioLDM2-music' that was trained on music only to generate 16kHz mono music (checkpoints[7]). The MusicGen variants we evaluate are: 'MusicGen-small' that is a compute-efficient autoregressive model for music generation, 'MusicGen-large' that is its large variant, and 'MusicGen-large-stereo' that is its stereo version. However, MusicCaps includes vocal-related prompts and MusicGen models are not trained to generate vocals. In Appendix E we also benchmark against MusicGen without vocal prompts. We also evaluate 'AudioGen-medium', the only open-source autoregressive model available for sound synthesis.

---

[6]Used checkpoint: '630k-audioset-fusion-best'.

[7]The used checkpoints are 'audioldm_48k', 'audioldm2-full-large-1150k' and 'audioldm2-music-665k', respectively.

	channels/sr	output length	$\text{FD}_{openl3} \downarrow$	$\text{KL}_{passt} \downarrow$	$\text{CLAP}_{score} \uparrow$	inference time
Training data (upper bound)	2/44.1kHz	full songs	101.47	-	-	-
Autoencoded training data	2/44.1kHz	full songs	117.52	-	-	-
Stable Audio w/ CLAP$_{\text{ours}}$	2/44.1kHz	23 sec	<u>118.09</u>	<u>0.97</u>	<u>0.44</u>	4 sec
Stable Audio w/ CLAP$_{\text{LAION}}$	2/44.1kHz	23 sec	123.30	1.09	0.43	4 sec
Stable Audio w/ T5	2/44.1kHz	23 sec	126.93	1.06	0.41	4 sec
AudioLDM2-music	1/16kHz	95 sec	354.05	1.53	0.30	38 sec
AudioLDM2-large	1/16kHz	95 sec	339.25	1.46	0.30	37 sec
AudioLDM2-48kHz	1/**48kHz**	95 sec	299.47	2.77	0.22	242 sec
MusicGen-small	1/32kHz	95 sec	205.65	0.96	0.33	126 sec
MusicGen-large	1/32kHz	95 sec	197.12	0.85	0.36	242 sec
MusicGen-large-stereo	**2**/32kHz	95 sec	216.07	1.04	0.32	295 sec
Stable Audio	**2**/44.1kHz	95 sec	**108.69**	**0.80**	**0.46**	**8** sec

Table 1. Quantitative results on MusicCaps. Top: autoencoder audio fidelity study, discussed in Section 6.1. Middle: text encoder ablation study, discussed in Section 6.2. Bottom: comparing Stable Audio against the state-of-the-art, see Section 6.4. Different experiments (top, middle, bottom sections of the table) are not strictly comparable due to different output lengths. <u>Underlines</u> denote the best results in the middle section of the table, and **bold** indicates the best results in the bottom section.

	channels/sr	output length	$\text{FD}_{openl3} \downarrow$	$\text{KL}_{passt} \downarrow$	$\text{CLAP}_{score} \uparrow$	inference time
Training data (upper bound)	2/44.1kHz	full-length audio	88.78	-	-	-
Autoencoded training data	2/44.1kHz	full-length audio	106.13	-	-	-
Stable Audio w/ CLAP$_{\text{ours}}$	2/44.1kHz	23 sec	<u>114.25</u>	<u>2.57</u>	0.16	4 sec
Stable Audio w/ CLAP$_{\text{LAION}}$	2/44.1kHz	23 sec	119.29	2.73	<u>0.19</u>	4 sec
Stable Audio w/ T5	2/44.1kHz	23 sec	119.28	2.69	0.11	4 sec
AudioLDM2-large	1/16kHz	10 sec	170.31	1.57	0.41	14 sec
AudioLDM2-48kHz	1/**48kHz**	10 sec	**101.11**	2.04	0.37	107 sec
AudioGen-medium	1/16kHz	10 sec	186.53	**1.42**	**0.45**	36 sec
Stable Audio	**2**/44.1kHz	95 sec †	**103.66**	2.89	0.24	**8** sec

Table 2. Quantitative results on AudioCaps. Top: autoencoder audio fidelity study, discussed in Section 6.1. Middle: text encoder ablation study, discussed in Section 6.2. Bottom: comparing Stable Audio against the state-of-the-art, see Section 6.4. Different experiments (top, middle, bottom sections of the table) are not strictly comparable due to different output lengths. † Stable Audio was trained to generate 95 sec outputs, but during inference it can generate variable-length outputs by relying on the timing conditioning. Despite Stable Audio generating 95 sec outputs and the rest of state-of-the-art models generating 10 sec outputs, it is still significantly faster. We trim audios to 10 sec (discarding the end silent part) for a fair quantitative evaluation against the state-of-the-art (see Section 3.4 for inference details).

	MusicCaps				AudioCaps		
	Stable Audio	MusicGen large	MusicGen stereo	AudioLDM2 48kHz	Stable Audio	AudioGen medium	AudioLDM2 48kHz
Audio Quality	**3.0**±0.7	2.1±0.9	2.8±0.7	1.2±0.5	**2.5**±0.8	1.3±0.4	2.2±0.9
Text Alignment	**2.9**±0.8	2.4±0.9	2.4±0.9	1.3±0.6	2.7±0.9	2.5±0.9	**2.9**±0.8
Musicality	**2.7**±0.9	2.0±0.9	**2.7**±0.9	1.5±0.7	-	-	-
Stereo correctness	**94.7**%	-	86.8%	-	57%	-	-
Structure: intro	**92.1**%	36.8%	52.6%	2.6%	-	-	-
Structure: development	65.7%	68.4%	**76.3**%	15.7%	-	-	-
Structure: outro	**89.4**%	26.3%	15.7%	2.6%	-	-	-

Table 3. Qualitative results. Top: mean opinion score ± standard deviation. Bottom: percentages. 19 users participated in this study.

6. Experiments

6.1. How does our autoencoder impact audio fidelity?

To understand the reconstruction capabilities of our latent space, we project a subset of training data (5,521 and 4,875 audios in Table 1 and 2, respectively) through our autoencoder to obtain the latents and reconstruct from them. Then, we compare the FD_{openl3} of the real and the autoencoded training data with respect to the MusicCaps and AudioCaps evaluation audio (Tables 1 and 2). In both cases, the autoencoded training data yields slightly inferior results compared to the real training data. This indicates a marginal degradation, yet informal listening suggests that the impact is fairly transparent (examples available in our demo website).

6.2. Which text encoder performs the best?

Various text encoders are prevalent in the literature, including: the open-source CLAP (Wu et al., 2023) denoted here as $CLAP_{LAION}$, privately trained CLAP-like models denoted here as $CLAP_{ours}$ (trained as in Section 4.3), and the open-source T5 embeddings. An ablation study is conducted in Tables 1 and 2 to determine which text encoder performs the best. In this study, we train the base diffusion model in Section 4.4 for 350k steps with different text encoders and evaluate them using our qualitative metrics both on Musiccaps and AudiocCaps. The text encoders are frozen during training. Results indicate comparable performance, with $CLAP_{ours}$ exhibiting a slight superiority, leading us to choose it for further experimentation. The utilization of a privately trained CLAP guarantees the use of text embeddings trained on the same dataset as our diffusion model. This approach ensures consistency across all components of the model, mitigating distribution or vocabulary mismatches between the text embeddings and the diffusion model.

6.3. How accurate is the timing conditioning?

The timing condition is evaluated by generating audios of variable lengths (length controlled by the timing condition) to note its behavior across different length values (Figure 3). We compare the expected length (provided by the timing conditioning) against the measured one, aiming for a diagonal in Figure 3. We measure the length of the audio by detecting when the signal becomes silence with a simple energy threshold—because, e.g., a model with a 30 sec timing condition is expected to fill the 95 sec window with 30 sec of signal plus 65 sec of silence. In Figure 3 we note that the model is consistently generating audios of the expected length, with more errors around 40-60 sec. This error might be caused because there is less training data of this duration. Also, note that some of the shortest measured lengths (seen in gray) may be false positives resulting from the simplistic silence detector we use. Appendix C includes more results.
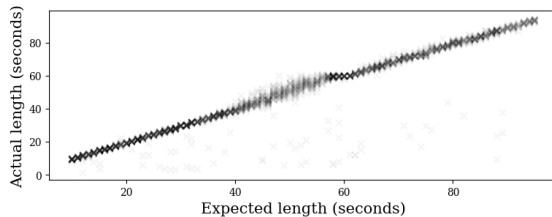

Figure 3. Comparing the actual length (measured in the signal) against the expected length (provided by the timing conditioning).

6.4. How does it compare with the state-of-the-art?

This section discusses Tables 1, 2, and 3. Stable Audio can outperform the state-of-the-art in audio quality and also improves text alignment in MusicCaps. Yet, text alignment is slightly worse in AudioCaps possibly due to the small amount of sound effects in our training set (Section 4.1). It is also very competitive at musicality and at generating correct stereo music signals. It's interesting, though, its low stereo correctness score in AudioCaps. It might be caused because the randomly selected prompts did not require much stereo movement, resulting in renders that are relatively non-spatial (see in our demo website). Despite this difficulty, the stereo render remained consistent without artefacts, leading to a stereo correctness score of 57%. Our demo website includes more stereo sound examples. Finally, Stable Audio is also capable to generate structured music: with intro, some degree of development, and outro. Note that state-of-the-art models are not consistent at generating a coherent structure, since they are mainly capable of developing musical ideas.

6.5. How fast is it?

We compare inference times using one A100 GPU and a batch size of 1. First, note that latent diffusion (AudioLDM2 and Stable Audio) is significantly faster than autoregressive modeling, as outlined in the introduction. Second, note that Stable Audio (operating at stereo 44.1kHz) is also faster than AudioLDM2-large and -music (operating at mono 16kHz). Stable Audio's speedup is even more significant when compared to AudioLDM2-48kHz (operating at mono 48kHz)[8].

7. Conclusions

Our latent diffusion model enables the rapid generation of variable-length, long-form stereo music and sounds at 44.1kHz from textual and timing inputs. We explored novel qualitative and quantitative metrics for evaluating long-form full-band stereo signals, and found Stable Audio to be a top contender, if not the top performer, in two public benchmarks. Differently from other state-of-the-art models, ours can generate music with structure and stereo sound effects.

---

[8] AudioLDM2-large and -music are implemented with Diffusers, 3x faster than the native implementation of the 48kHz one. AudioLDM2 runs use the setup recommended by the authors.

Acknowledgments

Impact Statement

Our technology represents a significant improvement in assisting humans with audio production tasks, offering the capability to generate variable-length, long-form stereo music and sound effects based on text descriptions. This innovation expands the toolbox available to artists and content creators, enriching their creativity. However, alongside its potential benefits, also confronts several inherent risks. One prominent concern lies in the reflection of biases present in the training data. This raises questions about the appropriateness of the technology for cultures underrepresented in the training dataset. Moreover, the contextual nature embedded in audio recordings and music emphasize the importance of careful consideration and collaboration with stakeholders. In light of these considerations, we commit to continued research and collaboration with stakeholders (like artists and data providers) to navigate the complex landscape of AI-based audio production responsibly.

References

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

Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. Align your latents: High-resolution video synthesis with latent diffusion models. *arXiv*, 2023.

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

Chang, H., Zhang, H., Jiang, L., Liu, C., and Freeman, W. T. Maskgit: Masked generative image transformer. *CCVPR*, 2022.

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

Cramer, A. L., Wu, H.-H., Salamon, J., and Bello, J. P. Look, listen, and learn more: Design choices for deep audio embeddings. *ICASSP*, 2019.

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. Flashat-tention: Fast and memory-efficient exact attention with io-awareness. *arXiv*, 2022.

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

Donahue, C., McAuley, J., and Puckette, M. Adversarial audio synthesis. *arXiv*, 2018.

Donahue, C., Caillon, A., Roberts, A., Manilow, E., Esling, P., Agostinelli, A., Verzetti, M., Simon, I., Pietquin, O., Zeghidour, N., et al. Singsong: Generating musical accompaniments from singing. *arXiv*, 2023.

Dong, H.-W., Liu, X., Pons, J., Bhattacharya, G., Pascual, S., Serrà, J., Berg-Kirkpatrick, T., and McAuley, J. Clipsonic: Text-to-audio synthesis with unlabeled videos and pretrained language-vision models. *arXiv*, 2023.

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

Fletcher, H. and Munson, W. A. Loudness, Its Definition, Measurement and Calculation. *The Journal of the Acoustical Society of America*, 2005.

Forsgren, S. and Martiros, H. Riffusion - stable diffusion for real-time music generation. 2022. URL https://github.com/riffusion/riffusion.

Garcia, H. F., Seetharaman, P., Kumar, R., and Pardo, B. Vampnet: Music generation via masked acoustic token modeling. *arXiv*, 2023.

Ghosal, D., Majumder, N., Mehrish, A., and Poria, S. Text-to-audio generation using instruction-tuned llm and latent diffusion model. *arXiv*, 2023.

Hawthorne, C., Simon, I., Roberts, A., Zeghidour, N., Gardner, J., Manilow, E., and Engel, J. Multi-instrument music synthesis with spectrogram diffusion. *arXiv*, 2022.

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., et al. Cnn architectures for large-scale audio classification. *ICASSP*, 2017.

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

Huang, Q., Jansen, A., Lee, J., Ganti, R., Li, J. Y., and Ellis, D. P. Mulan: A joint embedding of music audio and natural language. *ISMIR*, 2022.

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

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

Kilgour, K., Zuluaga, M., Roblek, D., and Sharifi, M. Fréchet audio distance: A metric for evaluating music enhancement algorithms. *arXiv*, 2018.

Kim, C. D., Kim, B., Lee, H., and Kim, G. Audiocaps: Generating captions for audios in the wild. *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv*, 2013.

Kong, J., Kim, J., and Bae, J. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 2020.

Koutini, K., Schlüter, J., Eghbal-zadeh, H., and Widmer, G. Efficient training of audio transformers with patchout. *Interspeech*, 2022.

Kreuk, F., Synnaeve, G., Polyak, A., Singer, U., Défossez, A., Copet, J., Parikh, D., Taigman, Y., and Adi, Y. Audiogen: Textually guided audio generation. *arXiv*, 2022.

Kumar, R., Seetharaman, P., Luebs, A., Kumar, I., and Kumar, K. High-fidelity audio compression with improved rvqgan. *arXiv*, 2023.

Lam, M. W., Tian, Q., Li, T., Yin, Z., Feng, S., Tu, M., Ji, Y., Xia, R., Ma, M., Song, X., et al. Efficient neural music generation. *NeurIPS*, 2024.

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

Li, P., Chen, B., Yao, Y., Wang, Y., Wang, A., and Wang, A. Jen-1: Text-guided universal music generation with omnidirectional diffusion models. *arXiv*, 2023.

Lin, S., Liu, B., Li, J., and Yang, X. Common diffusion noise schedules and sample steps are flawed. *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024.

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

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

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

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

Moliner, E., Lehtinen, J., and Välimäki, V. Solving audio inverse problems with a diffusion model. *ICASSP*, 2023.

NovelAI. Novelai improvements on stable diffusion, Oct 2022. URL https://shorturl.at/wW034.

Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., et al. Parallel wavenet: Fast high-fidelity speech synthesis. *ICML*, 2018.

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

Parker, J., Spijkervet, J., Kosta, K., Yesiler, F., Kuznetsov, B., Wang, J.-C., Avent, M., Chen, J., and Le, D. Stemgen: A music generation model that listens. *ICASSP*, 2024.

Pascual, S., Bhattacharya, G., Yeh, C., Pons, J., and Serrà, J. Full-band general audio synthesis with score-based diffusion. *ICASSP*, 2023.

Pasini, M. and Schlüter, J. Musika! fast infinite waveform music generation. *arXiv*, 2022.

Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer. *arXiv*, 2017.

Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., and Rombach, R. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv*, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. *arXiv*, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 2020.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. *arXiv*, 2022.

Rouard, S. and Hadjeres, G. Crash: Raw audio score-based generative modeling for controllable high-resolution drum sound synthesis. *arXiv*, 2021.

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

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

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

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv*, 2015.

Steinmetz, C. J., Pons, J., Pascual, S., and Serrà, J. Automatic multitrack mixing with a differentiable mixing console of neural audio effects. *arXiv*, 2020.

Vyas, A., Shi, B., Le, M., Tjandra, A., Wu, Y.-C., Guo, B., Zhang, J., Zhang, X., Adkins, R., Ngan, W., et al. Audiobox: Unified audio generation with natural language prompts. *arXiv*, 2023.

Wu, Y., Chen, K., Zhang, T., Hui, Y., Berg-Kirkpatrick, T., and Dubnov, S. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. *ICASSP*, 2023.

Yang, D., Tian, J., Tan, X., Huang, R., Liu, S., Chang, X., Shi, J., Zhao, S., Bian, J., Wu, X., et al. Uniaudio: An audio foundation model toward universal audio generation. *arXiv*, 2023.

Yao, Y., Li, P., Chen, B., and Wang, A. Jen-1 composer: A unified framework for high-fidelity multi-track music generation. *arXiv*, 2023.

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

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

A. Inference diffusion steps

In diffusion generative modeling, a critical consideration revolves around the trade-off between the quality of the generated outputs and the number of inference steps employed (quality vs inference-speed trade-off). Our results in Figure 4 show that a significant portion of the overall improvement in output quality is achieved within the initial 50 inference steps, suggesting diminishing returns with additional computational effort. Given that, we choose to set the total inference steps to 100. This decision is undertaken with a precautionary approach, ensuring a sufficient number of time steps to guarantee certain quality in the generated outputs. Nevertheless, this implies the possibility of being more aggressive with the number of diffusion steps to significantly accelerate our inference times without compromising much quality.
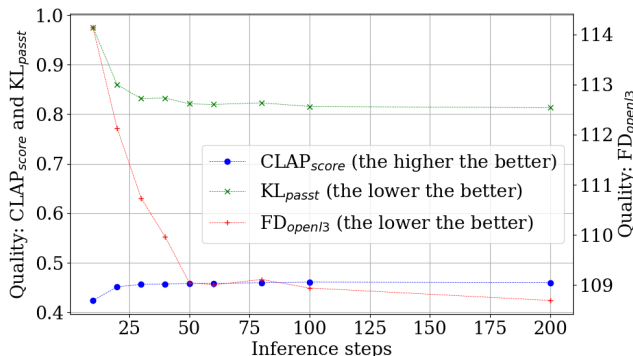

Figure 4. Quality metrics vs Inference diffusion steps (trade-off).

B. MusicCaps and AudioCaps: the original data from Youtube

MusicCaps and AudioCaps benchmarks are not commonly used for evaluating full-band stereo signals, since most researchers typically use mono versions at 16kHz of those datasets. However, the original data is predominantly stereo and full-band (see Figures 5 and 6). Provided that this data is easily available, we rely on the original data resampled at 44.1kHz to meet the target bandwidth of Stable Audio. This approach ensures that our evaluation encompasses the richness inherent in the original stereo full-band signals, providing a more accurate representation of the model's performance under conditions reflective of real-world data.
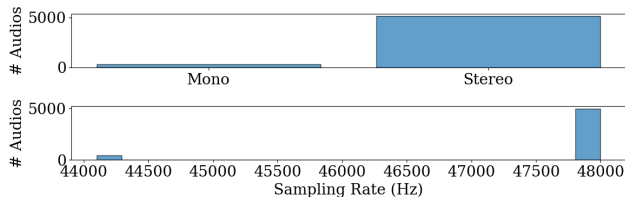

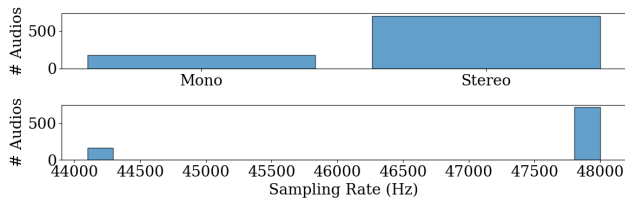*Figure 5.* Statistics of the MusicCaps original data.

Figure 6. Statistics of the AudioCaps original data.

C. Timing conditioning: additional evaluation

In Section 6.3 we observed that Stable Audio adheres to the timing conditioning to generate signals of the specified length.

We further study its behaviour by generating MusicCaps prompts at various lengths: 30, 60 and 90 sec. In Figure 7 we depict the histogram of the measured lengths, clustered by the specified lengths (blue 30 sec, red 60 sec, and green 90 sec). As in Section 6.3, we measure the length of the audio by detecting when the signal becomes silence with a simple energy threshold. In this experiment we note that the timing conditioning is fairly precise, generating audios of the expected length, with an error of a few seconds with a slight bias towards generating shorter audios. This means that the audio tends to finish right before the expected length, making it very appropriate to cut out the signal at the expected length. Also, note that some of the shortest measured lengths may be attributed to false positives resulting from the simplistic silence detector we use.
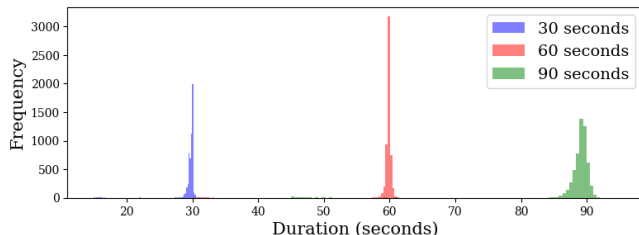

Figure 7. Histogram depicting the measured lengths of MusicCaps captions.

D. Related work: additional discussion on latent diffusion models

Moûsai and JEN-1 are closely related to our work. Both target high-fidelity stereo music synthesis with latent diffusion models. Our work, however, differs from Moûsai in several key aspects:

- Moûsai's latent is based on a spectrogram-based encoder and a diffusion decoder that requires 100 decoding steps, while ours in a fully-convolutional end-to-end VAE. This distinction is crucial in achieving our fast inference times.

- Moûsai's realtime factor is of $\times 1$, while ours is of $\times 10$.

- Moûsai uses a form of timing conditioning based on information about chunked audio files in the prompts (e.g. *Electro House, 3 of 4*), but we include explicit timing conditioning that allows for variable-length audio generation.

Our work differs from JEN-1 in the following aspects:

- JEN-1 relies on a masked autoencoder with a dimensionality reduced latent, and also on a *omnidirectional* latent diffusion model trained in a multitask fashion. In contrast, Stable Audio is inherently simpler with no *omnidirectional* constraints, no dimensionality reduction, and no multitask training. Our approach allows for an easier implementation and training while still being able to deliver state-of-the-art results.

- JEN-1 is trained to generate 10 sec of music, not to generate variable-length, long-form music and sound effects.

Note that Moûsai and JEN-1 target music synthesis while we target both music and sounds synthesis with a single model.

AudioLDM2 is also closely related to our work. It is a latent diffusion model capable to generate mono speech, sound effects, and music up to 48kHz. Although the original AudioLDM2 was designed to operate at 16kHz, a recent release operates at 48kHz. Our work, however, differs from AudioLDM2 in several key aspects:

- AudioLDM2 relies on a shared representation for music, audio, and speech to condition the latent diffusion model. This representation is shared between an audio masked auto encoder (audioMAE) and a GPT-2 that takes audio, text, speech transcripts, and images. As Stable Audio was not trained for speech generation or image-to-audio, there's no need to incorporate the intricacies of GPT-2. Instead, we opt for a CLAP text encoder.

- Stable Audio is faster and outperforms AudioLDM2 in audio quality and on text alignement for music generation. Yet, AudioLDM2 outperforms Stable Audio on text alignement for sound effects generation (see Tables 1, 2, and 3).

Moûsai, JEN-1, and AudioLDM2 use the open-source, pretrained T5 or FLAN-T5 text embeddings, and we use CLAP text embeddings trained on the same dataset as our diffusion model. Our approach ensures consistency across all components of the model, eliminating distribution (or vocabulary) mismatches between the text embeddings and the diffusion model.

E. Additional MusicCaps results: quantitative evaluation without singing-voice prompts

MusicCaps includes vocal-related prompts but MusicGen's released weights (used for benchmarking) are not trained to generate vocals[9]. To allow for a fair evaluation against MusicGen, we also evaluate the models in Table 1 with a subset of 2184 prompts that do not include vocals[10]. In Table 4, we observe results akin to those in Table 1: Stable Audio consistently obtains better results than the rest (with the exception of MusicGen-large that obtains comparable KL_{passt} scores to ours).

	channels/sr	output length	$FD_{openl3} \downarrow$	$KL_{passt} \downarrow$	$CLAP_{score} \uparrow$	inference time
AudioLDM2-music	1/16kHz	95 sec	354.37	1.66	0.32	38 sec
AudioLDM2-large	1/16kHz	95 sec	349.67	1.66	0.32	37 sec
AudioLDM2-48kHz	1/**48kHz**	95 sec	296.46	3.15	0.22	242 sec
MusicGen-small	1/32kHz	95 sec	186.28	1.02	0.34	126 sec
MusicGen-large	1/32kHz	95 sec	176.54	**0.86**	0.37	242 sec
MusicGen-large-stereo	**2**/32kHz	95 sec	196.66	1.08	0.33	295 sec
Stable Audio	**2**/44.1kHz	95 sec	**100.29**	**0.87**	**0.44**	**8** sec

Table 4. Quantitative results on MusicCaps without singing-voice prompts. Scores highlighted in **bold** indicate which are the best results.

F. Implementation details

Code to reproduce Stable Audio can be found online: https://github.com/Stability-AI/stable-audio-tools.

The configuration file used for training and defining our VAE autoencoder is available online.

The configuration file used for training and defining our latent diffusion model is available online.

The provided configuration files offer compact descriptions of the architecture and implementation of Stable Audio. These configurations serve as additional resources for comprehensively understanding the underlying implementation of Stable Audio.

Code to reproduce our metrics can be found online: https://github.com/Stability-AI/stable-audio-metrics.

We relied on the code shared by the CLAP authors (Wu et al., 2023) to train our text encoder with our private dataset:

https://github.com/LAION-AI/CLAP

---

[9] Vocals were removed from their training data source using the corresponding tags, and then using a music source separation method.

[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.