Diffusion-Based Sound Synthesis in Music Production

Pierre-Louis Wolfgang Léon Suckrow
Berlin University of the Arts
Berlin, Germany
Technical University of Berlin
Berlin, Germany
p.suckrow@udk-berlin.de

Christoph Johannes Weber
University of Television and Film Munich
Munich, Germany
LMU Munich
Munich, Germany
c.weber@hff-muc.de

Sylvia Rothe
University of Television and Film Munich
Munich, Germany
s.rothe@hff-muc.de

Abstract

In this paper, we explore the usability of generative artificial intelligence in music production through the development of a digital instrument that incorporates diffusion-based sound synthesis in its sound generation. Current text-to-audio models offer a novel method of defining sounds, which we aim to render utilizable in a music-production environment. Selected pretrained latent diffusion models, enable the synthesis of playable sounds through textual descriptions, which we incorporated into a digital instrument that integrates with standard music production tools. The resultant user interface not only allows generating but also modifying the sounds by editing model and instrument-specific parameters. We evaluated the applicability of current diffusion models with their parameters as well as the fitness of possible prompts for music production scenarios. Adapting published diffusion model pipelines for integration into the instrument, we facilitate experimentation and exploration of this innovative sound synthesis method. Our findings show that despite facing some limitations in the models' responsiveness to specific music production contexts and the instrument's functionality, the tool allows the development of novel and intriguing soundscapes. The instrument and code is published under https://github.com/suckrowPierre/WaveGenSynth.

CCS Concepts: • **Applied computing** → **Sound and music computing**; • **Computing methodologies** → *Artificial intelligence*; • **Human-centered computing** → *Text input*; *User studies.*

Keywords: sound generation, text-to-sound, user interface, user study

FARM '24, September 2, 2024, Milan, Italy

ACM ISBN 979-8-4007-1099-5/24/09
https://doi.org/10.1145/3677996.3678289

ACM Reference Format:
Pierre-Louis Wolfgang Léon Suckrow, Christoph Johannes Weber, and Sylvia Rothe. 2024. Diffusion-Based Sound Synthesis in Music Production. In *Proceedings of the 12th ACM SIGPLAN International Workshop on Functional Art, Music, Modelling, and Design (FARM '24), September 2, 2024, Milan, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3677996.3678289

1 Introduction

Recent advancements in artificial intelligence and generative models have significantly enhanced and simplified the process of creating new sounds. Traditionally, artists and sound engineers relied on large and comprehensive sound libraries to meet their creative and technical needs. While these libraries provide quick access to a broad spectrum of sounds, various problems can be identified.

Recorded sounds, bound by distinct licenses and copyrights, often limit sound usability and adaptation to safeguard creators, yet these restrictions can impede the creative process. AI-generated content emerges as a solution to these legal hurdles, though the status of AI authorship and copyright is currently under debate. Another point is that the complexity and utility of digital sound libraries increase with their size and chosen compression, facilitating the discovery of specific sounds but complicating their search as the collection expands. This challenge becomes more pronounced beyond a certain threshold, where the addition of new sounds entails the inclusion of variations on existing themes, necessitating that artists compare multiple similar sounds to find the one that precisely meets their requirements. AI's capability to generate any sound from text descriptions enables artists to efficiently create unique sounds, circumventing traditional library constraints.

Musicians and composers have always explored new methods for creating and performing music, from traditional instruments, analog synthesizers to digital computation [27]. The evolution of computing technology, driven partly by the quest to innovate and preserve sounds, underscores a historical symbiosis between binary computing and musical advancement [27]. This relationship, rooted in the ancient connection between mathematics and music theory [16],

highlights the continuous impact of technological progress on musical creativity. Generative models represent the latest advancement in a series of innovations that have transformed music production and sound design. Thus we formulate the following research question:

RQ: Are diffusion models suitable for use as a sound source in a music production process?

This paper investigates the implementation of a synthesizer leveraging novel *latent diffusion* [26] technology. We present a digital instrument called *WaveGenSynth*[1], which allows us to incorporate pretrained music generation models and generate sounds with natural language. We also analyzed the quality of the generated audio samples and their correspondence with the input prompt.

Both the resulting digital instrument and the potential advantages and limitations of this implementation are examined. In addition, the potential of the *diffusion* models used in the context of musical applications is assessed. Our realization aims to empower users to generate, modify, and play sounds according to their individual preferences. The primary objective of this work is to conduct a comprehensive analysis of various implementation strategies as well as the results of the digital instrument and the *diffusion* models used. This involves identifying their advantages and limitations, and evaluating their effectiveness as a novel tool for musical expression. Additionally, our paper seeks to design the instrument in a way that facilitates its smooth integration into modern music production workflows, thereby offering musicians novel avenues for sound exploration and synthesis techniques.

2 Related Work

Synthesizing data through generative AI is rendered possible under the premise that examined data emanates from a distinct distribution, which Machine-Learning models strive to approximate, thereby enabling the extraction of novel data points through distribution sampling [6, 13, 28]. The models employed in this work synthesize sound through a *diffusion* process. As stated by Haohe Liu[2] utilizing *diffusion* [4, 7, 18, 30] as a means to synthesize sound offers numerous benefits, including the potential to democratize the sector by reducing dependency on expensive recorded sounds, giving users enhanced control over timbres and sound properties, and from a creative standpoint, enable the shaping of unique sounds that may inspire artists.

In the *diffusion* process, drawing on statistical thermodynamics [11] and sequential Monte Carlo methods [17], a neural network aims to iteratively denoise data in the backward process by learning from distortions typically introduced via Gaussian noise [29] in the forward process. This generative model demonstrates enhanced efficiency over similar models [18, 30]. The denoising of a data point makes it appear as though new data is generated from random noise. Building upon this concept, *latent diffusion* [26] allows the generation of images from textual descriptions by shifting the *diffusion* process from operating on raw pixel values to acting within a latent space created by a *variational autoencoder* (*VAE*) [12]. This innovation not only minimizes computational requirements for broader hardware compatibility but also supports diverse media conditioning. It optimizes semantic information retention, enhancing generated image quality [26].

Inspired by *Contrastive Language-Image Pretraining* (*CLIP*) [22], Large-Scale *Contrastive Language-Audio Pretraining* (*CLAP*) [34] develops a unified latent representation for audio and text through the training on audio-text pairs, effectively mapping their shared information into corresponding embeddings. This latent space enables tasks like text extraction from audio and audio generation from text descriptions. [34]

Music, as an art form and cultural expression, involves the structured organization of sound over time, distinguished from noise by its structured harmonies pleasing to the human ear [19, 31]. Sound, at its physical level, is a wave generated by vibration and includes a fundamental tone accompanied by resonating overtones, giving rise to various harmonics that define its timbre [19, 31]. The analysis of sound focuses on the relationships among tones, classifying the lowest dominant frequency as the fundamental tone and higher frequencies as overtones, which are considered harmonic when their frequencies are integer multiples of the fundamental tone, and inharmonic otherwise, leading to the unique sonic qualities of different instruments [19, 27, 33]. The sound's texture, timbre and characteristics, are described through the relationship of its underlying tones and their amplitude and frequency and can be visualized and distinguished over time in a spectrogram [24]. This foundational understanding supports the use of spectrograms in representing audio as images, facilitating the extraction of information or generation of audio signals through image-based computational intelligence methods.

Applying this methodology of using image based architecture on audio is the generative model *AudioLDM* [14] allowing for the synthesis and manipulation of audio through a natural language input. Unlike previous efforts such as *Diffsound* [35], *AudioLDM* does not directly use text-audio pairs in training, which could limit the scope of available data, and instead relies only on audio data by incorporating *CLAP* into its architecture. Following [26], a *VAE* [12] is employed to convert mel-spectrogram audio into a latent space for

[1] https://github.com/suckrowPierre/WaveGenSynth
[2] https://www.youtube.com/watch?v=6qtL9_T8m3c

diffusion. The embeddings from *CLAP* are used to condition the model both during training and at inference, enabling it to extract information from the input prompts. The process culminates in the generation of a spectrogram that is subsequently transformed into an audio signal with a vocoder. [14]

Expanding on *AudioLDM*, the *AudioLDM2* [15] model introduces a novel architecture that promises enhanced outcomes in audio generation. It aims to overcome the constraints of prior audio synthesis models, which were often biased to specific tasks or domains, by integrating a novel *language of audio* (*LOA*) representation for conditioning. This representation aims to generalize across various domains and modalities such as text, audio, images and videos. The *diffusion* process occurs in the same latent space as proposed in [14]. The used neural network, has been further refined, amongst other things with attention mechanisms [32]. The model employs the self-supervised *AudioMAE* [10] for converting audio into *LOA* and *GPT-2* [23] for other modalities. The former playing a role during training and fine-tuning, and the latter being employed during inference and the majority of fine-tuning. [15]

Possible methodologies and approaches for creating instruments that facilitate sound synthesis via artificial intelligence have been explored by the "Neural Audio Plugin"[3] competition, organized by *The Audio Programmer Ltd*[4] in the spring of 2023. Among the participants, the digital instrument *VroomAI*[5] stood out, offering the capability to generate playable sounds from textual descriptions using generative AI models. *VroomAI* offers two usable models: *AudioLDM* [14] and *AudioLM* [1], the former of which is also employed in our tool. The subsequent model, *AudioLDM2* [15], was not available at the time of *VroomAI*'s latest maintaining date, thus not being available as an option, unlike in our instrument. The architecture used in *VroomAI* mirrors that of our approach, with both initiatives utilizing the *Juce*[6] *C++* framework for digital instrument development. Unlike this project, *VroomAI* depends on the *AudioLDM* published *Gradio*[7] inference server[8] for sound generation. A process complicated by hardware limitations, particularly on devices lacking *NVIDIA CUDA*[9] support, leading us to the development of a bespoke inference server for our application using *FastAPI*[10] and the published *AudioLDM* endpoints on the Huggingface repository. In contrast to *VroomAI*, we enable the capability to adjust audio parameters such as envelope, gain, and tuning, as well as model parameters including the negative prompt, number of inference steps, guidance scale, audio length, and the seed directly within the instrument. Unlike our current version, it is possible to set looping point in *VroomAI*, a feature absent in our project but that we want to explore in later revisions of this first prototype. Lastly, the distribution process for *VroomAI* and it's *Gradio* server necessitates source code compilation by the user, a technical barrier that has been successfully eliminated in the deployment of our final product.

3 Methods

3.1 Application Design

To ensure the developed instrument exhibits minimal latency during use, the predominant choice for programming language in real-time audio applications C++ [3, 5] is being used in conjunction with the *JUCE* framework. Enriching the framework with the Module *PluginGuiMagic*[11] helped in the development of the GUI (see figure 1).

JUCE is an open-source C++ codebase, enabling the creation of standalone software across multiple platforms and different plugin formats. The framework provides an abstraction layer for audio processing and *MIDI* from native audio devices on each platform or a Host-DAW. The digital signal processing (DSP) library offered by *JUCE* facilitates rapid prototyping and implementation of various audio effects, filters, instruments, and generators. Additionally, *JUCE* offers a multitude of classes addressing common challenges in audio project development, including the management of graphics, sound, user interaction, and network communication. [25]

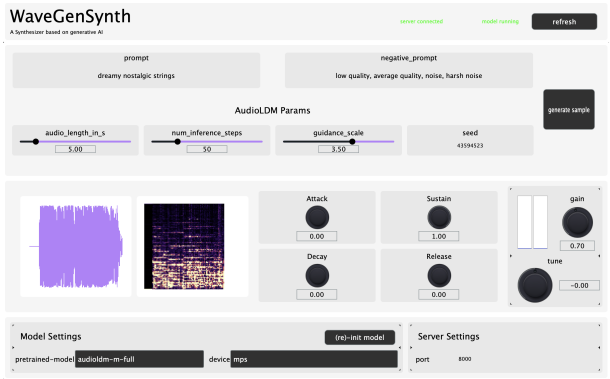

Figure 1. Graphical interface to connect to inference server, choose pretrained model, define prompt, model parameters, playback parameters and visualize generated waveform

[3] https://www.theaudioprogrammer.com/neural-audio
[4] https://www.theaudioprogrammer.com/
[5] https://vroomai.com/
[6] https://juce.com/
[7] https://www.gradio.app/
[8] https://github.com/haoheliu/AudioLDM/
[9] https://developer.nvidia.com/cuda-zone
[10] https://fastapi.tiangolo.com/
[11] https://foleysfinest.com/developer/pluginguimagic/

To integrate the *diffusion* models *AudioLDM* and *AudioLDM2* into the *JUCE* digital instrument framework, it is necessary to facilitate their inference directly from the instrument's C++ code. Oliver Larkin's suggestion[12] of generating binary representations in ONNX [13] or Torchscript [14] formats was not feasible due to the models' complex structures and dependencies on various sub-models. Instead, we employed a *FastAPI*[15] server to host instances of *AudioLDM* and *AudioLDM2* models, such as *audioldm-s-full-v2*, *audioldm-m-full*, *audioldm-l-full*, *audioldm2*, *audioldm2-large*, and *audioldm2-music*, published via HuggingFace[16,17]. These models are accessible via an API endpoint, allowing initialization with a specified *torch device*. For silicone architectures, "mps" is the recommended torch device, whereas "cuda" is suitable for NVIDIA GPUs, when supported, and "cpu" for remaining cases. Choosing model, device and initializing is facilitated in the Instruments GUI (see figure 2). Model inference is performed by defining parameters such as prompt, negative-prompt, audio length, inference steps, guidance scale and an optional 8-Byte large seed, also settable in the GUI (see figure 3), with the dedicated endpoint returning the generated audio as a *Base64* encoded signal for processing in the instrument's C++ code.

Once generated, the audio signal is visualized through

Figure 2. Options to set model, torch-device and initialize the model in the GUI

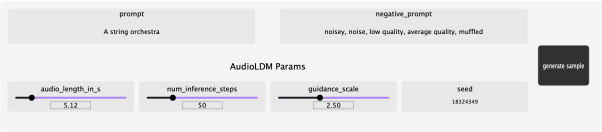

Figure 3. Options to set prompt, model parameters and seed in the GUI

its waveform and spectrogram and loaded into a sampler, which was implemented by augmenting the provided *Juce* class for continuous pitch shifting over an octave and also allows usage of the keyboard's pitch bender with a standard two-semitone range. During playback, the GUI (see figure 4) allows users to modify the sound's Envelope by adjusting

Figure 4. Options to change the envelope, gain, and tuning in the GUI

the Attack, Decay, Sustain, and Release parameters. It also provides options to adjust the gain and tune the instrument, ensuring the pitch of the generated audio aligns with the MIDI mapping.

For macOS, we generated a *.app* application, a *VST3*, and an *AU* file, which were collectively packaged into a *pkg* installer. The inference server was encapsulated into a standalone application using PyInstaller[18], simplifying its launch by bundling all necessary dependencies into a single package. Despite the original Silicon server script being only 5 KB, this packaging process increases the file size to 450 MB due to the inclusion of all required modules. The increase in storage requirement is substantial but indispensable for distributing and publishing the server and thus a usable instrument. For systems experiencing unstable performance or issues with the server application, running the server script within a specified Conda[19] environment is an alternative solution.

The digital instrument was tested both on macOS with Silicon, Intel chipset. When using the server application on Intel Macs with a mandatory torch-device "cpu", poor and unstable performance was observed. In this case, it is advisable to run the script for the server in the Conda environment with the specified Python modules.

The code for the instrument is published at *GitHub*[20]. The repository includes the source code for both the digital instrument and the *FastApi* server. Additionally, a plugin installer for Mac, executable server files for Silicon and Intel architectures on Mac, and a zip file are published[21]. The zip file contains the Python script for the server and the required Python packages for Conda, enabling the server to run without an executable file. The README file provides instructions for installing and running the code.

[12] https://www.youtube.com/watch?v=t662qg12f_Y
[13] https://onnx.ai/
[14] https://pytorch.org/docs/stable/jit.html
[15] https://fastapi.tiangolo.com/
[16] https://huggingface.co/cvssp/audioldm
[17] https://huggingface.co/cvssp/audioldm2
[18] https://pyinstaller.org/en/stable/
[19] https://www.anaconda.com/
[20] https://github.com/suckrowPierre/WaveGenSynth
[21] https://github.com/suckrowPierre/WaveGenSynth/releases/

3.2 Study Design

To assess the suitability of *AudioLDM* and *AudioLDM2* models for generating short audio samples in our application, we conducted a survey involving n=6 participants, some of whom are experienced in music production, making them apt for this evaluation. Participants rated their music production expertise on a 1-10 scale, with the average score being 5, indicating a moderate level of experience. The survey's design used Euler's square to ensure question distribution diversity, mitigating the risk of repetitive answer patterns due to convenience or habit. Our results were measured by posing questions related to both the quality of sound and how well the audio reflects the given input prompt (audio-text alignment) on a Likert scale of 1-5, where 1 represents the most unpleasant and 5 represents the most pleasant result.

All audio was generated with a fixed seed and following negative prompts: "low quality", "average quality", "noise", "high pitch", "artefacts". Our main goal was to investigate the following points:

Influence of device: Due to numerical precision and therefore small precision variations, parallelization and order of operations, and other reasons, results may sometimes show small differences when executed on different torch-devices. Therefore, as a first step, we decided to check whether the choice of device had a influence on the result of the generated sounds. The measurement was done for four prompts "Warm organ chord", "Ambient ocean waves", "Bright bell sound" and "Smooth synth lead" (see figure 5). The length of the audio has been adjusted to 5 seconds, with 50 inference steps and a guidance scale of 3.

Influence of guidance scale: *Diffusion* models use a guidance scale to tell the model how closely it should stick to the given prompt [8, 15]. Lower numbers allow for more diversity and creativity in the results, while higher numbers cause the model to adhere more strictly and thus more fittingly to the text prompt. We let the participants evaluate generated audio for the prompts "Gentle guitar strum", "Tribal African drum circle", "vocal harmonies". (see table 1).

Influence of inference steps: In *stable diffusion*, the term *inference steps*, also referred to as *reverse diffusion steps* refers to the number of passes the model makes over an image before it produces a final result [14, 15]. As a rule, more steps mean better results. However, beyond a certain threshold, the marginal benefit of increasing the number of steps diminishes [14]. We let participants rate the generated audio for different inference steps to determine if there was a sweet spot for generating audio with audio length adjusted to 5 seconds and the guidance scale set to 3. For this purpose, we generated audio for 5, 10, 20, 50, 100, 200, and 400 steps for the same prompts (see figure 6).

Influence of duration: To be usable for digital instruments, the sound quality and audio-text alignment of the sounds must be as good as possible for sounds of different lengths. If there is a loss of quality, it is important to know this to limit the generation of sounds to the range in which the quality is still sufficiently good. We have therefore analyzed the sound quality and the audio-text alignment of the sounds as a function of the audio length for the prompts "Chirping birds at dawn", "A choir pad", and "An ambient electronic pad". The guidance scale has been adjusted to 3 and the number of inference steps has been set to 50.

Influence of the prompt: The prompt has a decisive influence on the sound. Therefore, the participants were asked to rate 32 generated sounds based on their fidelity to the describing prompt. The length of the audio has been adjusted to 5 seconds, with 100 inference steps and a guidance scale of 3. (see appendix A.1 table 2 - 6)

Other observations: While this study primarily focused on specific variables and parameters, it also yielded complementary observations that, although not directly incorporated into the study design, provide valuable insights into the broader context of the research topic. Despite their anecdotal or tangential nature, these observations merit recognition and consideration of their potential implications. These observations were brought to the attention of the study participants and discussed verbally with us. The results of this will be discussed in the following chapter.

4 Results

The results of the present study are based on a small sample size (n=6), which leads to a limitation in terms of statistical significance. With so few participants, the statistical power of the ANOVA test is low [21], which affects the reliability of the results and the generalisability of the conclusions. Consequently, the results of this study should be treated with particular caution, as ANOVA tests with such a small sample are not powerful enough to draw reliable conclusions. It is recommended that these results be considered as preliminary and that further research be conducted with a more appropriate number of participants to confirm the findings and increase their validity.

Analysis of participant feedback reveals that model selection impacts outcomes more than the choice of computing device (see figure 5). Interestingly enough using consistent seeds and parameters, audio generation varied across different torch devices, with CPU-produced audio being less favorable. Among the tested devices, the base *AudioLDM2*

model was preferred for its superior general audio quality and accurate audio-text alignment. Surprisingly, despite its simplicity, the base *AudioLDM* model outperformed the more complex *audioldm2-large* model, indicating that a higher number of parameters does not necessarily equate to better performance in this context.

Model characteristics other than the larger parameter size of

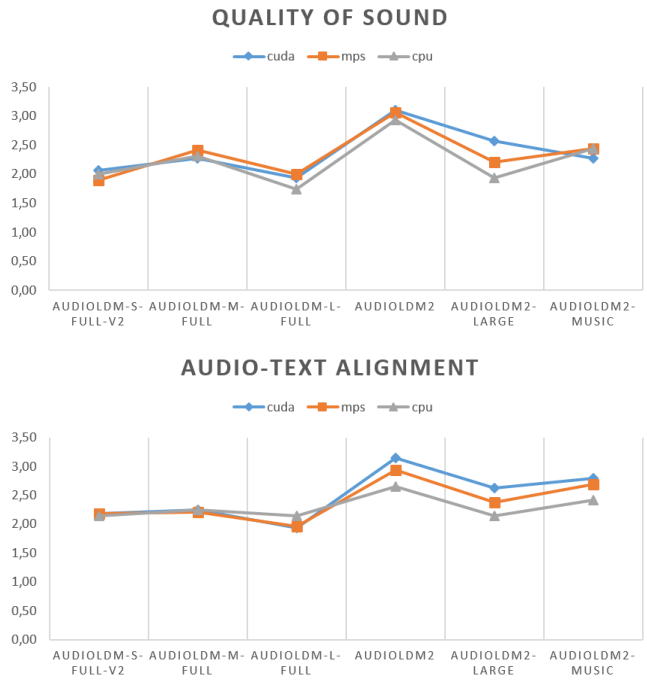

Figure 5. Impact of the choice of device type and model on the average perceived sound quality and the audio-text alignment on a Likert scale from 1-5.

the *AudioLDM2* models may also influence the results. The smaller training set of the *audioldm2-music* model compared to the other *Audioldm2* models, the specific music conditioning of *audioldm2-music*, or the fact that *audioldm-m-full* also performs the audio conditioning have different effects on the results depending on the task.

Lower guidance-scale values cause the model to follow the prompt less strictly. Therefore it is natural that the perceived audio-text alignment increases with higher values of the guidance scale. However, the perceived quality of the sound also seems to increase slightly. The results showed that optimum results can be expected for values from a guidance scale of 4-5 (see table 1)

When looking at the results of the inference steps, it is noticeable that perceived sound quality as well as audio-text alignment initially rises sharply, but then falls. Naturally,

Table 1. Guidance scale evaluation on perceived sound quality and audio-text alignment for all models on average, measured on a Likert scale from 1 to 5 while 1 means very poor and 5 is very high quality.

	1	2	3	4	5
Sound Quality	2,03	2,27	2,33	2,42	2,41
Audio-Text Alignment	2,10	2,34	2,33	2,46	2,47

one anticipates enhanced outcomes through increased iterations, as repeated refinement generally improves quality. Yet, encountering a deterioration or stagnation of results appears paradoxical initially. This phenomenon, common in *diffusion* generators, occurs because adding more information to a fully generated image yields no additional value and merely supersedes existing details.

The participants' ratings were approximately equal for the

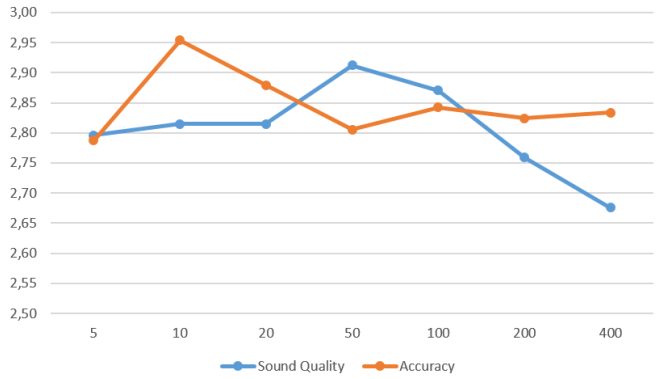

Figure 6. Effect of number of inference steps from 5-400 on the perceived audio quality and audio-text alignment on the average of all variants of *AudioLDM*, measured on a Likert scale from 1-5.

sound quality and audio-text alignment of the audio for all audio lengths between 5 and 30 seconds. In some of the longer generated audio a sort of pitch drifting was observed. This could be an undesired or appreciated side effect based on the use case and personal preference.

The expectation that prompts like "A kickdrum", "A snare", "Loud clap sound", and "A gong hit" would elicit a singular, non-repetitive audio event was not fully met (see appendix A.1 figure 2). The models often produced signals with repeated audio events, indicating a tendency towards creating rhythmic musical patterns rather than isolating a single sound event. A more precise prompt, such as "A single", seems to be more effective with *audioldm-m-full*. The *audioldm2-music* model tends to generate repetitive structures, presumably because its primary function is to produce

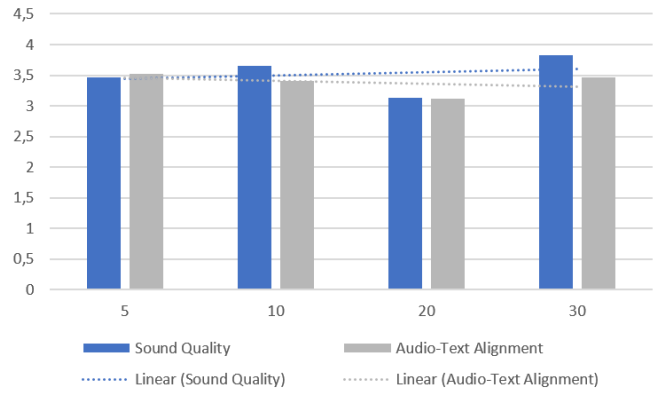

Figure 7. Impact of the duration of the generated audio (5 to 30 seconds) on the perceived audio quality and audio-text alignment, measured on a Likert scale from 1-5.

musical compositions rather than singular sound events.

Attempts to synthesize characteristic instrument sounds using prompts like "FM synthesis bells", "mellotron chords", "A bagpipe melody", "A guitar string", and "A piano chord" did not achieve the anticipated outcomes with *audioldm-m-full*, while *audioldm2* and *audioldm2-music* proved more adequate for this task. (see appendix A.1 figure 3). The sounds generated by *audioldm-m-full* were abstract and failed to accurately replicate the intended instruments. Additionally, it was noted that the less satisfactory outcomes from *AudioLDM2* exhibited a tendency to sound alike.

Adding emotional and descriptive elements to prompts, including "Dark pad sound", "An ethereal shimmering synth pad", "An angelic choir", "dreamy nostalgic strings", and "a sad violin solo", generally did not lead to favorable outcomes, (see appendix A.1) figure 4). It has been noted again that the outcomes from *AudioLDM2*, despite variating of prompts, show that less satisfactory results share similar sound structures.

The models appear capable of adequately responding to specific descriptions of musical and acoustic effects (see appendix A.1 figure 5). Interestingly, the *AudioLDM* models seem to perform better in this aspect than the *AudioLDM2* models, although one might expect the latter to achieve better results given their *LOA* representation.

Efforts to craft prompts incorporating specific sounds and terminology common in music production and pop culture resulted in unsatisfactory outcomes with *audioldm2* and achieved mediocre results with *audioldm2-music* (see appendix A.1 figure 6). Attempts to synthesize a 808-Kickdrum, prevalent in hip-hop, or a 909-Kickdrum, a staple in electronic music, resulted in timbres that failed to match the expected specifications. Similarly, generating the widely used Amen Break did not lead to the anticipated success. Only *audioldm-m-full* managed a resemblance to a 303 Baseline, with other models falling short. The prompts "A Juno-106 pad" and "Oberheimer OB-Xa string pads" yielded functional pads, implying sustained tones or chords, yet whether they authentically replicate the unique sounds of these instruments remains a matter of debate.

5 Discussion

The analysis indicates that existing models in music production may benefit from further refinement to improve their sensitivity to input prompts. Addressing these issues may involve training on a broader range of audio data from sources such as synthesizer or sampler manufacturers, audio libraries (e.g., *Splice*[22]), music studios, or producers, who often maintain personal sound libraries for their work. Fine-tuning models with more fitting music production data may lead to a more tailored and effective generative AI tool, as demonstrated by the customization of *AudioLDM* discussed in [20]. This approach aligns with the trend towards personalized *LORA*s [9] in other creative fields. Aggregating individual sound libraries into a unified database could provide a democratic solution to the lack of datasets in this area, offering a basis for more precise fine-tuning and training of models.

Employed models should aim to produce outputs at the sampling rate of 48kHz recommended by the *Audio Engineering Society* [2]. The models *AudioLDM* and *AudioLDM2* output at a rate of 16kHz, thus falling short of generating sounds in High Fidelity.

The failure to create an *ONNX* or *TorchScript* representation of the model for inference in C++ introduces complications in bundling and distributing the model. The creation of such a binary file for the utilized model would eliminate the need for a server and an API. Likewise, the requirement to use multiple programming languages for an implementation would no longer be necessary.

The current Sampler implementation lacks the ability to set start and end points for signal playback, limiting the selection of specific sound events from sequences. Additionally, it misses the capability to loop segments for continuous play, a standard feature in samplers since the 1990s. This brings forth the question of the necessity to develop a proprietary

[22] https://splice.com/

Sampler. Inspired by interfaces like *Text generation web UI*[23], *ComfyUI*[24], and *Stable Diffusion web UI*[25], a specialized interface could be created. This interface could integrating with current music production software while allowing detailed parameter adjustments, model fine-tuning, inpainting, style transfer and many other options. Assuming the model's effectiveness, such a platform could potentially make services like *Splice* redundant.

Incorporating machine learning into development of a custom Sampler could enhance realism by adjusting overtones for different pitches, a necessity beyond simple frequency modulation. This approach reflects the complexity of piano timbres, where each note's spectrum varies significantly [19]. A machine learning model could be trained to modify these subtle differences accurately, taking an audio signal as input and producing outputs for all possible pitches. This process necessitates a comprehensive dataset of audio signals from multiple instruments, encompassing all pitches.

Some musicians show a preference for hardware over a digital workflow due to its tactile feedback. The sound synthesis technique described in this study could be adaptable to dedicated hardware, utilizing deep learning solutions on microcontrollers, as seen with Google's *Coral*[26] and Nvidia's *Jetson*[27]. This would be a more accessible approach for musicians not wanting to invest time in setting up the software. Nonetheless, compatibility issues may arise, such as the requirement for models to be compatible with *TensorFlow Lite*[28] on *Coral* devices. We were able to run *AudioLDM2* successfully on Nvidia's *Jetson AGX Xavier Development Kit*, suggesting that the proposed methodology could extend to portable *Jetson* devices, transforming them into standalone, text-driven musical instruments with the addition of a soundcard and a MIDI keyboard for sound output and control.

The instrument exhibits several deficiencies that require optimization, such as the manual tuning process, which could benefit from automation. Additionally, it lacks the capability to save and recall generated sounds and settings, a functionality prevalent in numerous digital instruments. During the development of the initial prototype, the specification and verification of the source code were not conducted. Future implementations should include these processes to reduce the likelihood of bugs and errors in the software. Despite the identified limitations, the digital instrument, through *diffusion* models, has shown capability in producing playable and editable sounds based on user inputs. It also integrates seamlessly into contemporary music production and applications, pioneering AI-driven sound synthesis. Unlike other media, such as images, even suboptimal outputs from this instrument can be creatively repurposed, enhancing soundscapes through extensive editing and effects. This is illustrated by musician "Heinbach," who adopts Bob Ross's "Happy Little Accidents"[29] approach to refine unexpected sounds into novel timbres and soundscapes. Consequently, while not primarily designed for precise sound modeling, the instrument encourages exploration with innovative and unforeseen sounds.

6 Conclusions

Our research shows the capabilities and limitations of audio generation models within the framework of a digital instrument. Despite *AudioLDM* and *AudioLDM2* models not achieving perfection in the music-production context, and their complexity preventing binary conversion to simplify implementation and distribution, the trajectory of audio generation development is foreseeable. The evaluation of these models has uncovered limitations that mainly restrict their use in the discovery of novel, sometimes unforeseen sounds. Nonetheless, despite these limitations, there lies the potential to use these unforeseen and perhaps alien sounds as a basis for further manipulation and transformation. The modular design of our digital instrument notably facilitates the seamless replacement of one *diffusion* model with another, thereby ensuring adaptability and readiness for future enhancements. Future models with improved sound quality and fewer artifacts could challenge traditional instruments. Additionally, the potential for individual fine-tuning through one's sound libraries has been recognized, and an improvement in sampling rate to 48kHz has been advised.

The absence of features that allow for the adjustment of start and end points of the played sounds as well as the definition of loop points for sustaining a sound continuously limits the editing capabilities of the generated sound. Moreover, the instrument would benefit from additional optimizations such as an automatic tuning function and the ability to save and retrieve sounds.

Despite these deficiencies, it is undeniable that the work highlights the potential of musical applications of *diffusion* models, offering improved and more versatile integration possibilities into modern music production ecosystems compared to similar applications. Users can parameterize and generate sounds according to their preferences and modify the sound behavior during play. Input devices can be connected via MIDI.

[23] https://github.com/oobabooga/text-generation-webui
[24] https://github.com/comfyanonymous/ComfyUI
[25] https://github.com/AUTOMATIC1111/stable-diffusion-webui
[26] https://coral.ai/
[27] https://developer.nvidia.com/embedded/jetson-modules
[28] https://www.tensorflow.org/lite
[29] https://www.youtube.com/watch?v=Ibajy9A91ls

We hope that future research in this field will benefit from the discoveries made in this work and will continue to build upon them. Perspectives for further diverse potential integrations of generative Artificial Intelligence into the music production process have been highlighted.

Acknowledgments

We would like to express our gratitude to Daniel Walz of *Plugin GUI Magic*[30] for his invaluable assistance and guidance throughout the implementation of the graphical user interface in this research project. We also extend our thanks to the *AudioLDM* and *AudioLDM2* team for their significant contributions to the development and publication of their models. Additionally, we appreciate the support of *The Audio Programmer*[31] Community.

A Study Data

A.1 Prompt Evaluation

Table 2. Perceived audio-text alignment in describing a **single event** on Likert scale from 1 Orange to 5 Blue where 1 represents the most unpleasant and 5 represents the most pleasant result.

	Model		
Prompt	audioldm2	audioldm-m-full	audioldm2-music
A kickdrum	2.17	3.33	2.67
A single kickdrum	3.17	2.67	1.00
A snare	2.67	2.67	1.67
A single snare	2.25	3.67	1.50
A single Light triangle ting	3.17	3.67	4.00
Loud clap sound	1.00	1.17	1.33
A gong hit	2.17	3.50	2.50

Table 3. Perceived audio-text alignment in **describing an instrument** on Likert scale from 1 Orange to 5 Blue where 1 represents the most unpleasant and 5 represents the most pleasant result.

	Model		
Prompt	audioldm2	audioldm-m-full	audioldm2-music
FM synthesis bells	3.67	2.00	3.83
Mellotron chords	2.67	1.17	2.67
A bagpipe melody	4.67	2.00	2.17
A guitar string	1.67	1.00	3.17

Table 4. Perceived audio-text alignment for audios that should encode **emotions and adjectives in prompt** on Likert scale from 1 Orange to 5 Blue where 1 represents the most unpleasant and 5 represents the most pleasant result.

	Model		
Prompt	audioldm2	audioldm-m-full	audioldm2-music
Dark pad sound	3.50	1.50	3.67
An ethereal shimmering synth pad	2.50	2.67	2.00
An angelic choir	1.80	3.33	1.50
Dreamy nostalgic strings	1.67	1.50	1.83
A sad violin solo	1.67	3.67	2.00

Table 5. Perceived audio-text alignment in translating **audio effects** on Likert scale from 1 Orange to 5 Blue where 1 represents the most unpleasant and 5 represents the most pleasant result.

	Model		
Prompt	audioldm2	audioldm-m-full	audioldm2-music
Long sustain snare hit	1.67	2.33	2.00
A fluttering harp with crystal echoes	3.33	1.00	3.50
A synth with delay effect	3.50	3.50	2.67
Echoing synth stabs	2.33	3.80	3.50
A distorted synth	2.83	4.83	1.33
A detuned synth	2.17	3.00	2.50
Reverse cymbal	1.83	2.83	2.00
A kickdrum with a lot of reverb	2.80	2.00	2.80

Table 6. Perceived audio-text alignment in describing **music production specific prompts** on Likert scale from 1 Orange to 5 Blue where 1 represents the most unpleasant and 5 represents the most pleasant result.

	Model		
Prompt	audioldm2	audioldm-m-full	audioldm2-music
an 808 kickdrum	1.83	3.50	1.67
The amen break	1.83	2.17	2.17
a 909 snare	1.00	2.00	2.33
a 303 baseline	1.83	3.00	2.33
A jungle drum break	1.67	3.00	1.83
A Juno-106 pad	1.83	3.00	2.67
Oberheimer OB-Xa string pads	2.33	2.83	2.67

[30] https://foleysfinest.com/developer/pluginguimagic/
[31] https://www.theaudioprogrammer.com/

References

[1] Andrea Agostinelli, Timo I. Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matt Sharifi, Neil Zeghidour, and Christian Frank. 2023. MusicLM: Generating Music From Text. https://doi.org/10.48550/arXiv.2301.11325 arXiv:2301.11325 [cs.SD]

[2] Audio Engineering Society, Inc. 2018. *AES5-2018 (Rev. AES5-2008) - AES recommended practice for professional digital audio — Preferred sampling frequencies for applications employing pulse-code modulation.* Technical Report AES5-2018. Audio Engineering Society, Inc. https://www.aes.org/tmpFiles/aessc/20231014/aes05-2018-r2023-i.pdf

[3] Richard Charles Boulanger and Victor Lazzarini (Eds.). 2011. *The audio programming book* (Cambridge, Mass, 2011). MIT Press. OCLC: ocn503654559.

[4] Prafulla Dhariwal and Alex Nichol. 2021. Diffusion Models Beat GANs on Image Synthesis. https://doi.org/10.48550/arXiv.2105.05233 arXiv:2105.05233 [cs.LG]

[5] Timur Doumler. 2015. C++ in the Audio Industry. (2015). https://www.youtube.com/watch?v=boPEO2auJj4 CppCon 2015.

[6] Harshvardhan GM, Mahendra Kumar Gourisaria, Manjusha Pandey, and Siddharth Swarup Rautaray. 2020. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review* 38 (2020), 100285. https://doi.org/10.1016/j.cosrev.2020.100285

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. https://doi.org/10.48550/arXiv.2006.11239 arXiv:2006.11239 [cs.LG]

[8] Jonathan Ho and Tim Salimans. 2022. Classifier-Free Diffusion Guidance. https://doi.org/10.48550/arXiv.2207.12598 arXiv:2207.12598 [cs.LG]

[9] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. https://doi.org/10.48550/arXiv.2106.09685 arXiv:2106.09685 [cs.CL]

[10] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. 2023. Masked Autoencoders that Listen. https://doi.org/10.48550/arXiv.2207.06405 arXiv:2207.06405 [cs.SD]

[11] C. Jarzynski. 1997. Equilibrium free-energy differences from nonequilibrium measurements: A master-equation approach. *Physical Review E* 56, 5 (Nov. 1997), 5018–5035. https://doi.org/10.1103/physreve.56.5018

[12] Diederik P Kingma and Max Welling. 2022. Auto-Encoding Variational Bayes. https://doi.org/10.48550/arXiv.1312.6114 arXiv:1312.6114 [stat.ML]

[13] Alex Lamb. 2021. A Brief Introduction to Generative Models. https://doi.org/10.48550/arXiv.2103.00265 arXiv:2103.00265 [cs.LG]

[14] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D. Plumbley. 2023. AudioLDM: Text-to-Audio Generation with Latent Diffusion Models. arXiv:2301.12503 [cs.SD] https://arxiv.org/abs/2301.12503

[15] Haohe Liu, Qiao Tian, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D. Plumbley. 2023. AudioLDM 2: Learning Holistic Audio Generation with Self-supervised Pretraining. https://doi.org/10.48550/arXiv.2308.05734 arXiv:2308.05734 [cs.SD]

[16] Eli Maor. 2018. *Music by the Numbers: From Pythagoras to Schoenberg*. Princeton University Press.

[17] Radford M. Neal. 1998. Annealed Importance Sampling. https://doi.org/10.48550/arXiv.physics/9803008 arXiv:physics/9803008 [physics.comp-ph]

[18] Alex Nichol and Prafulla Dhariwal. 2021. Improved Denoising Diffusion Probabilistic Models. https://doi.org/10.48550/arXiv.2102.09672 arXiv:2102.09672 [cs.LG]

[19] Barry R. Parker. 2009. *Good vibrations: the physics of music*. Johns Hopkins University Press. OCLC: ocn320194527.

[20] Manos Plitsis, Theodoros Kouzelis, Georgios Paraskevopoulos, Vassilis Katsouros, and Yannis Panagakis. 2023. Investigating Personalization Methods in Text to Music Generation. https://doi.org/10.48550/arXiv.2309.11140 arXiv:2309.11140 [cs.SD]

[21] Patrick J Potvin and Robert W Schutz. 2000. Statistical power for the two-factor repeated measures ANOVA. *Behavior Research Methods, Instruments, & Computers* 32, 2 (2000), 347–356. https://doi.org/10.3758/BF03207805

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. https://doi.org/10.48550/arXiv.2103.00020 arXiv:2103.00020 [cs.CV]

[23] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. https://api.semanticscholar.org/CorpusID:160025533

[24] Hannes Raffaseder. 2010. *Audiodesign* (2., aktualisierte und erweiterte auflage ed.). Hanser.

[25] Martin Robinson. 2013. *Getting started with JUCE: leverage the power of the JUCE framework to start developing applications*. Packt Publishing. OCLC: 862386437.

[26] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. https://doi.org/10.48550/arXiv.2112.10752 arXiv:2112.10752 [cs.CV]

[27] André Ruschkowski. 2019. *Elektronische Klänge und musikalische Entdeckungen* (3., ergänzte auflage 2019 ed.). Number 19613 in Reclams Universal-Bibliothek. Reclam.

[28] Lars Ruthotto and Eldad Haber. 2021. An introduction to deep generative modeling. *GAMM-Mitteilungen* 44, 2 (2021), e202100008. https://doi.org/10.1002/gamm.202100008 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/gamm.202100008

[29] C.E. Shannon. 1949. Communication in the Presence of Noise. *Proceedings of the IRE* 37, 1 (1949), 10–21. https://doi.org/10.1109/JRPROC.1949.232969

[30] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. https://doi.org/10.48550/arXiv.1503.03585 arXiv:1503.03585 [cs.LG]

[31] Kinko Tsuji and Stefan C. Müller. 2021. *Physics and music: essential connections and illuminating excursions*. Springer Nature.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. https://doi.org/10.48550/arXiv.1706.03762 arXiv:1706.03762 [cs.CL]

[33] Harvey Elliott White and Donald H. White. 2014. *Physics and music: the science of musical sound*. Dover Publications, Inc. OCLC: 878661301.

[34] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2023. Large-scale Contrastive Language-Audio Pretraining with Feature Fusion and Keyword-to-Caption Augmentation. https://doi.org/10.48550/arXiv.2211.06687 arXiv:2211.06687 [cs.SD]

[35] Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. 2023. Diffsound: Discrete Diffusion Model for Text-to-sound Generation. https://doi.org/10.48550/arXiv.2207.09983 arXiv:2207.09983 [cs.SD]

Received 2024-06-02; accepted 2024-07-02