Latent Diffusion Model and "Language of Audio" in Generative Audio

Ethan Graybar
grayb031@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

Abstract

This paper aims to outline the advancements of audio generative technology, specifically *AudioLDM 2*, a model that creates audio given a set of inputs. As an overall summary of the work and working parts of AudioLDM 2, this paper will cover; latent diffusion models and their inefficiencies; the optimization of GPUs inside of LDMs; Language of Audio (LOA); the GPT-2 Language model; and the results of tests done on AudioLDM 2 through the scholarly paper "AudioLDM 2: Learning Holistic Audio Generation With Self-Supervised Pretraining" by Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D. Plumbley ([7]. With the evaluation and results presented later, this paper demonstrates the effectiveness of new and old systems in audio-generative AI, and aims to push computer science towards continued improvement in the field. Special attention will be drawn to Language of Audio as it is unique to AudioLDM 2.

Keywords: autoencoder, self-supervised pretrained, Mel spectrogram, patches, dimensional latent space

1 Introduction

Generative AI effectively accomplishes the tasks that are given to them, specifically text-to-text generation. As primarily text-generators, the AI that we are most familiar with excels at producing one type, or subdomain, of content. The question that this paper seeks to answer is: Does there exist an AI model that can produce different subdomains of quality audio content?

While some generative AI models have the capacity to create quality content in one or two subdomains, there is yet to exist a model that can generate quality content in several subdomains. For example, *AudioLDM* (the predecessor to *AudioLDM 2*) generates high quality audio in the text-to-audio and text-to-music subdomain, but cannot create satisfactory audio in the text-to-speech subdomain [7]. The speech generated by AudioLDM loosely resembles spoken word, but is mostly unintelligible. What AudioLDM 2 aims to do is create a wide variety of audio by paying more attention to the information that makes quality audio.

To demonstrate the advancements of *AudioLDM 2* compared to similar counterparts like *AudioLDM*, *AudioLDM 2* is tasked with generating three different subdomains of audio - general audio (sound effects), music, and speech. *AudioLDM 2* was given three different kinds of data during testing; for general audio, AudioCaps; music - MSD; and speech - LJSpeech. AudioCaps is a dataset including 50,000 general sound effects, each one labeled with an event description. The event descriptions provide information about the sound effects in the data, answering questions like "Where was the sound created?" and "What made that sound?" MSD, or Million Song Dataset, is a collection of over one million popular music songs. However, only 510,000 were used to train AudioMAE and GPT-2 during this paper's experiment. LJSpeech is a dataset composed of 13,100 audio samples of one person reading from a set of non-fiction books [7]. The results of these tests show that *AudioLDM 2* generates exemplary audio and music compared to other audio generators, and speech that is improved compared to *AudioLDM*. While generated speech may not be *AudioLDM 2's* specialty, the results show that the speech generated is certainly comparable to other speech generators, such as FastSpeech2.

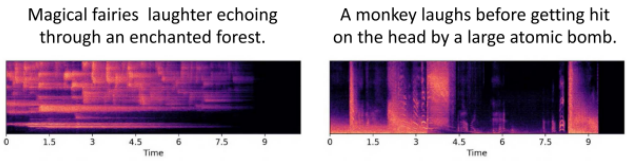

Figure 1. Audio generated by AudioLDM 2 given two different text prompts. Audio is represented using Mel spectrograms [7].

In this paper, we will dive into background information to better understand specialized terms, examine Language of Audio, go through both GPT-2 and latent diffusion models, and conclude with the results of experiments done on *AudioLDM 2*.

2 Background

In order to thoroughly understand the process of deep learning throughout *AudioLDM 2's* generation, we need to explore training. Training is the act of teaching a learning model, in this case *Audio LDM 2*. Learning models are given large amounts of training data, and with this data these models

gain knowledge so that they can create the most accurate predictions possible. *AudioLDM 2* is trained on large numbers of sample audio, ranging from something like contemporary rock music to something as random as animal noises. The immensity of data that *AudioLDM 2* is trained on allows the model to create accurate predictions on what a successful output would look like given varying kinds of inputs. The more accurate the prediction, the more accurate the output audio [4].

However, unlike most learning models, *AudioLDM 2* features deep learning architectures that are **self-supervised pretrained**. The "self-supervised" part of self-supervised pretrained models takes important information from unlabeled training and testing data. Other learning models require labels in the training data so that the models know what to focus on when learning information. *AudioLDM 2* is improved compared to other models because it does not require labeled data, which is often hard to find [4]. Using models that are self-supervised pretrained, such as AudioMAE and the core LDM (the audio generator in *AudioLDM 2*), allows *AudioLDM 2* to use unlabeled training and testing data. Unlabeled data is easier to find and is less costly than purchasing labeled information [4]. "Pretraining" functions similarly to self-supervision as it removes complexity in the training and testing phases. Pretraining is when an a component, in our case an autoencoder, is given information before being trained on conditioning information. This gives the autoencoder context BEFORE being trained further. AudioMAE and the LDM were pretrained on large amounts of varying audio data from datasets like AudioSet and GigaSpeech. Being that AudioMAE and the LDM have learned from such a large amount of varied data before training, they can be trained and tested on much smaller datasets with less descriptive labeling. For example, the LDM is tested on only a random 10-second clip from the thousands of audio samples provided by AudioCaps, MSD, and LJSpeech. The LDM in *AudioLDM 2* is able to take such limited testing information because it was first pretrained on a wider sample of data.

AudioMAE and VAE are learning and extraction models that take information from training and testing data and represent it in a more usable format, such as LOA. VAE stands for variational autoencoder, and it is the autoencoder inside of the LDM. What these models learn from data is semantic and acoustic information. The process of learning in AudioMAE, VAE, and LDMs is through an **autoencoder** architecture (Fig. 2). Autoencoders take in an input; encode it, compressing the information; then decode the compressed format to most closely resemble the input data [2]. Doing this gives us the acoustic and semantic information that is within the input conditioning information (information organized specifically for a systems purpose) that we need to generate audio.

AudioMAE takes in an audio sample and converts the sample into a **Mel spectrogram** - an image of the audio

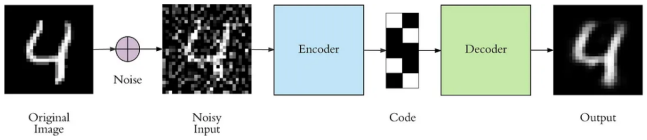

Figure 2. Example of an autoencoder architecture. Instead of images, AudioMAE takes audio as an input and the LDM takes in text prompts [2].

sample's waveform (Fig. 1). AudioMAE deconstructs the Mel spectrogram into individual chunks called **patches** that are used as the input for the AudioMAE autoencoder. The encoder adds noise to these patches, creating masked patches. AudioMAE then decodes these masked patches, learning the semantic and acoustic information that makes up LOA in the process. By adding noise to these patches, the encoder makes it harder for the decoder to find the useful information it needs to recreate the input. This challenge given to the decoder enforces the self-supervisory nature of the autoencoder, forcing the decoder to focus on only the important information. Imagine looking for a red balloon that is 200 yards away. You could keep your eyes wide open, taking in the entire landscape, or you could squint, focusing your eyes on the target and cutting out your peripherals. This is essentially how noising/masking works in autoencoders.

VAE (variational autoencoder) is used as a "feature compressor" [7] and a model that learns from the compressed audio representation (z) created by VAE during testing. Input z is the compressed testing information that VAE is given during the testing phase, which can be seen in the green box of Fig. 3. In this experiment, *AudioLDM 2's* VAE is given a text prompt that it will use to generate a relevant audio sample.

3 Language of Audio (LOA)

LOA is a representation of data that is easier to model than the initial input x. Our initial input x is conditioning information, specifically an audio sample. Important semantic and acoustic information is taken from our conditioning information and turned into our representational LOA using AudioMAE.

- **Semantic information:** the meaning of audio (or speech within audio)
- **Acoustic details:** frequency (pitch), amplitude (volume), etc.

LOA is accepted later on in the audio generation process as an input, eventually producing $\hat{x}$, the final output created by *AudioLDM 2*, is a sample of high quality audio, speech, or music. The process for creating audio using *AudioLDM 2* is shown in Fig. 3. The training phase includes the red and blue boxes, respectively, the AudioMAE and GPT-2 language models. These two boxes give us a prediction (LOA and LOA ground truth) of what the output should look like, and the

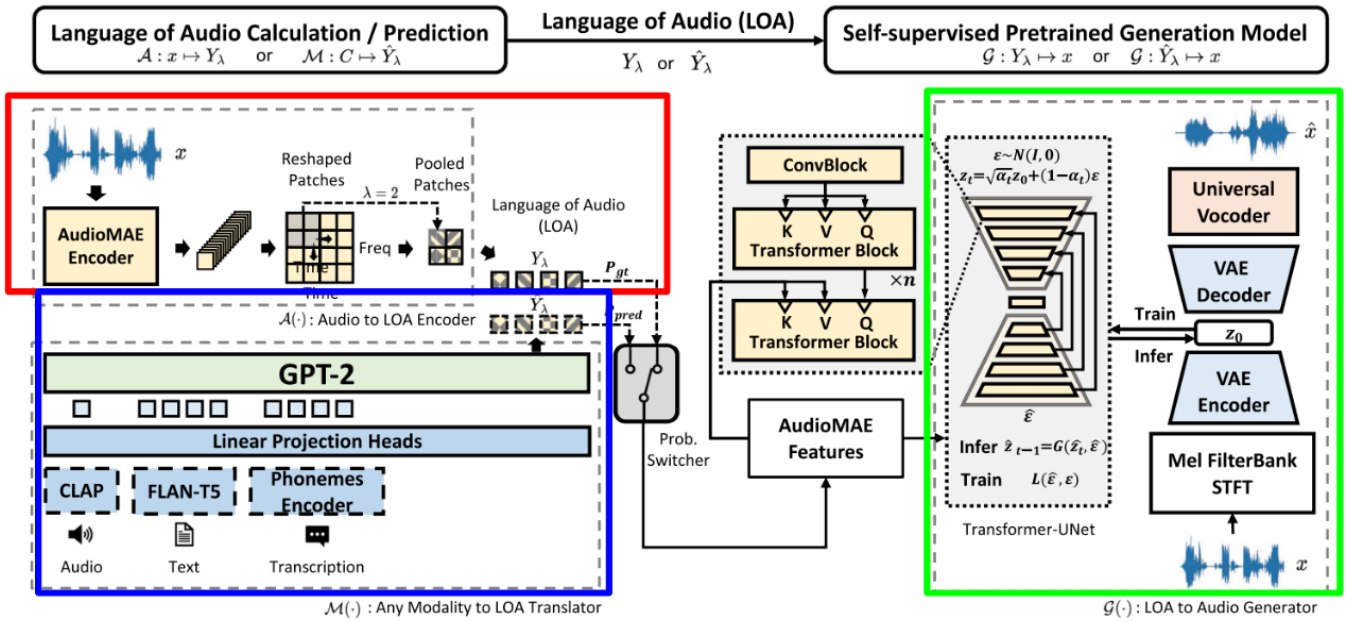

Figure 3. Three colored modules represent the AudioMAE, GPT-2 model, and LDM. The red and blue modules are used during testing to gather LOA. Green generates audio given testing data [7].

green box (latent diffusion model) uses that prediction to create an accurate output.

LOA gives *AudioLDM 2* the ability to accept lots of different inputs, essentially acting as a universal translator for the model. Without the additional step of creating LOA, *AudioLDM 2* could only receive labeled training and testing data, and all predictions would be made by the LDM. The addition of LOA adds a productive step to *AudioLDM 2's* framework that gives it a leg up against other audio generators.

4 GPT-2 Language Model

Language of Audio is generated from the input x, but what happens when we introduce an input outside of the traditional format that x comes in? If given training or testing data other than audio, the GPT-2 (Generative Pretrained Transformer) pretraining model is used. This model takes in conditioning information in the form of audio, text, and **transcription** (text derived from speech.)

GPT-2 language model creates a more accurate representation of conditioning information than AudioMAE that is used in the audio generation stage of AudioLDM 2. What the GPT-2 language model creates is called the ground truth LOA which is used as a comparison between our experimental LOA and the separate ground truth. Ground truth: an estimation of the true or correct output of the dataset [5]. The ground truth LOA is the most correct representation of data within LOA. In other words, the ground truth LOA is the semantic and acoustic details that relates most to the output audio.

The GPT-2 language model has the benefit of being able to transform multiple types of conditioning information; audio, text, and transcription. GPT-2 language model formats conditioning information into a representation that is usable by AudioMAE features [7]. In order to train on different types of input data than AudioMAE, GPT-2 uses separate encoders. **CLAP** (Contrastive Language and Audio Pretraining) is a pretrained text encoder that is the "default" when handling conditioning information. **FLAN-T5** is an aid to CLAP as CLAP oftentimes misses important semantic information in the conditioning input. FLAN-T5 is also pretrained, similarly to CLAP. Our last encoder in the GPT-2 model is the **Phoneme Encoder**, whose job is to find "the smallest units of sound in a language that can distinguish one word from another [7]". The Phoneme Encoder is not pretrained because its parameters are based off of another generative system, *Natural Speech*.

5 Latent Diffusion Model

The diffusion process is an example of an **autoencoder architecture** (Figure 2) which refines the quality of a model's output. More specifically, autoencoder architectures work to re-represent an output based on an input. Autoencoders do so by compressing an input into a **dimensional latent space**, or a simplified, coded version of our source information [2]. Taking an input and transforming it into a lower dimensional latent space is part of the encoding process. This lower dimensional latent space is the middle ground between the encoder and decoder. Once this simplified coded version of the input is established, the decoder does its best to create

another representation of the input based on the exclusive information given by the latent space. This output is often a crude version of the input, but with the output representation, we can get a clear idea of how close our model is to generating a *quality* output.

Another important step in the diffusion process is getting the model to notice important material more efficiently. This is done by adding noise, or a random distribution of data, to the input. Without the original input, the autoencoder architecture has nothing to reference, therefore we know it is doing its job by producing a noise free output [2]. This trains the model to pinpoint the important information and subtract unnecessary material.

5.1 GPU Optimization

Graphics Processing Units, or GPUs, are essential hardware components in generative artificial intelligence models (such as *AudioLDM 2*) that directly impact the efficiency of latent diffusion models. GPUs perform computation processes for generative AI, and are effective tools used to process large amounts of data and/or complex computations. However, during the complex compuatations involved with autoencoders and LDMs, GPUs tend to slow down. This leads to longer **training times** and an over-consumption of energy which can negatively affect the environment [1]. By finding inefficiencies in autoencoders and LDMs, we can both reduce training times and improve the energy efficiency of generative AI. Profiling and monitoring tools are used to find and subsequently reduce inefficiencies.

GPU inefficiencies are tracked using these three metrics:

- **GPU Utilization:** amount of time the GPU is processing data
- **GPU Memory Allocation:** the amount of memory assigned to a process
- **GPU Power Usage:** amount of energy used during processing

In terms of GPU utilization, higher utilization means more efficient GPU usage. Likewise, GPU memory allocation gauges the complexity of computations and the size of datasets used by the model as the GPU utilizes more memory.

Profiling tools analyze the performance of neural network processes such as execution runtime (the amount of time a process runs from beginning to end) and memory usage (the amount of memory a process uses). Through these metrics we can monitor how deep-learning algorithms interact with hardware, thus finding GPU inefficiencies. Some helpful profiling tools in the search for these inefficiencies are NSight by NVIDIA and PyTorch Profiler. These tools target the training process of diffusion models as this is the process that generates an output.

Optimizing GPUs is a difficult process in terms of generative AI, seeing that different models have different optimization needs. *AudioLDM* and *Stable Diffusion* are two models with contrasting optimization necessities. AudioLDM manages memory less efficiently than Stable Diffusion because AudioLDM processes more information and operates on a more complex architecture. Stable Diffusion struggled with complex computations, indicating that the GPUs used with Stable Diffusion could work on optimizing data utilization [1].

6 Evaluation Metrics

Two different types of metrics are used to rate the audio outputs of *AudioLDM 2* during this experiment: Quantitative and Subjective metrics. The CLAP, FAD, and KLD scores are all quantitative metrics, meaning they are numerically measurable metrics used to rate *AudioLDM 2's* outputs. The CLAP score is an independent metric compared to the FAD and KL scores which were created by AudioGen - another audio generative AI model. The CLAP score is used to measure how close the input text prompt is to the generated audio.

Frechet Audio Distance measures the mean distribution distance between the generated audio and a target, in this case, pre-recorded music. FAD is a **reference free evaluation metric**, meaning it doesn't need a sample of audio from outside of AudioLDM 2 to compare generated audio. In order to make these comparisons, FAD takes statistics from the generated audio (in this case music) and compares them with a set of statistics from clean, studio-recorded music. This gives us a measure of how close the generated audio is to quality music [6].

The next evaluation metric, Kullback-Leibler divergence, compares the similarity of the generated audio and a target with a baseline standard output from another audio generation model. KL divergence takes probability distributions from the generated and target audio, and the baseline audio and compares their drift - or how far a distribution is from the baseline. Think of a bar graph with a set of purple bars (reference distribution) and gold bars (baseline distribution) sitting next to each other. KL divergence measures the difference in size between the two sets of bars - the closer in size the reference distribution is to the baseline, the better [3].

Beyond comparing the semantics of generated audio, AudioLDM 2 includes subjective evaluation metrics in order to capture the qualitative accuracy of generated audio. A set of questions (labeled OVL, REL, and MOS) were given to 10 different raters per generated audio clip, and their answers were either given on a numeric rating scale or through multiple choice.

The OVL metric presented raters with the question *How would you rate the overall quality of this music? Consider its resemblance to real-world audio and its naturalness*, and asked them to rate their answers on a scale of *5-Excellent* to *1-Bad* [7]. REL proposed a similar question to OVL and used the same rating scale: *How would you rate the relevance of music to the text description?* Raters were given the following

question in MOS: *How natural does this recording sound? Take into account emotion, prosody, and other human-like details.* Raters in this group expressed their opinions using a range of answers ranging from *completely unnatural* to *speech* to *perfectly natural speech.*

7 Results

The results in Tables 1, 2, and 3 show that AudioLDM 2, with its usage of a latent diffusion model and LOA, matches the performance of other generative AI models (SoTa) in text-to-music and text-to-audio tasks [7]. AudioLDM 2 was tasked with generating audio through three different processes during testing: text-to-speech, text-to-audio, and text-to-music. Three different datasets were used to test the different sub-domain outputs of *AudioLDM 2*: AudioCaps, MSD, and LJSpeech. AudioCaps is a subset of AudioSet, and it features 46,000 ten-second clips of audio samples. MSD or Million Song Dataset contains 510,000 music clips including artists names, titles, and song labels. Lastly, LJSpeech offers small clips of an individual speaker, numbering at 13,100 audio samples [7]. In order to effectively measure how successful AudioLDM 2 is at generating audio, AudioLDM 2 was compared with these generative models: AudioLDM, AudioGen-Large, Make-an-Audio (both original and Make-an-Audio 2), and TANGO.

Table 1. Text-to-Audio (AudioCaps) Results Table

	FAD ↓	KL ↓	CLAP ↑	OVL ↑	REL ↑
AudioLDM	4.53	1.99	0.141	3.61	3.55
Make-an-Audio	2.05	1.27	0.173	3.68	3.62
TANGO	1.73	1.27	0.176	3.75	3.72
AudioLDM 2-AC	1.67	1.01	**0.249**	3.88	**3.90**
AudioLDM 2-AC-Large	**1.42**	**0.98**	0.243	**3.89**	3.87

Concluding text-to-audio tests (Table 1), *AudioLDM 2* performed better than all three comparison generators. AudioLDM is the predecessor to AudioLDM, AudioLDM 2-AC is AudioLDM 2 tested on AudioCaps, and AudioLDM 2-AC-Large is AudioLDM 2 tested on a scaled up version of AudioCaps. The most basic form of text-to-audio tested, *AudioLDM 2-AC*, received the highest CLAP score and rated the highest in the REL subjective metric. When *AudioLDM 2-AC* was given larger amounts of testing data, making it *AudioLDM 2-AC-Large*, it performed better in all other metrics compared to TANGO, Make-an-Audio 2, and AudioLDM. Of particular notice is *AudioLDM 2's* advancement in comparison to its predecessor AudioLDM-M. Both were models pretrained on large datasets, but *AudioLDM 2* performed far better.

Moving on to text-to-music generation (Table 2), we find that *AudioLDM 2* passed its competitors in test result scores. AudioLDM 2-MSD is AudioLDM 2 tested on the MSD dataset and AudioLDM 2-Full is AudioLDM 2 that generates both audio output and music output simultaneously. Our basic text-to-music model, *AudioLDM 2-MSD*, received the highest

Table 2. Text-to-Music (MSD) Results Table

	FAD ↓	KL ↓	CLAP ↑	OVL ↑	REL ↑
AudioLDM	3.20	1.29	**0.360**	3.03	3.25
MusicGen	3.40	1.23	0.320	3.37	3.38
AudioLDM 2-MSD	4.47	1.32	0.294	**3.41**	3.30
AudioLDM 2-Full	**3.13**	**1.20**	0.301	3.34	**3.54**

OVL score and *AudioLDM 2-Full* received the highest REL score. For reference, models with the suffix "Full" are given more training data and are capable of generating audio and music at the same time [7]. *AudioLDM 2-Full* earned higher FAD and KL divergence scores compared to MusicGen, AudioLDM, and MusicLM. This shows that *AudioLDM 2* performs text-to-music generation at such high efficiency that it can multitask.

Table 3. Text-to-Speech (LJSpeech) Results Table

	MOS ↑
GroundTruth	**4.63**
FastSpeech2	3.78
AudioLDM 2-LJS	3.65
AudioLDM 2-GIG	4.00

Unlike the previous two generated subdomains, text-to-speech (Table 3) was less successful. AudioLDM 2-LJS is AudioLDM 2 tested on the LJSpeech dataset and AudioLDM 2-GIG is the results of AudioLDM 2 tested on GigaSpeech. *AudioLDM 2-LJS* received the lowest MOS score compared to the other generators, coming in at 3.65. However, when *AudioLDM 2* was pretrained on GigaSpeech instead of LJSpeech it performed better than LJSpeech at a score of 4.00. GigaSpeech is a dataset with more audio samples in a larger variety than what LJSpeech offers, which shows the importance of diversifying the training data. Text-to-speech processes where only scored with the MOS scale because of the subjectivity of speech. Comparing Table 1 and Table 2 to our text-to-speech Table 3, we find that AudioLDM was not included. This is because AudioLDM - *AudioLDM 2's* predecessor - generated unintelligible audio, thus incomparable to the generated outputs of *AudioLDM 2* and FastSpeech2.

8 Conclusion

AudioLDM 2 is comparable to other audio generative AI in text-to-audio, text-to-music, and text-to-speech generation. However, *AudioLDM 2* excels at text-to-audio and text-to-music generation, even going as far as generating both subdomains simultaneously.

Throughout each sub-domain experiment, it was clear that without LOA, the efficiency and quality of the results would have been different. LOA changes the motivation of optimization from "How can we improve learning?" to "What datasets provide the best results?" Because LOA acts

as a universal audio translator, *AudioLDM 2* and other audio generative AI have the ability to focus on different areas of improvement.

References

[1] Bradley Aldous and Ahmed M. Abdelmoniem. 2024. Comparative Profiling: Insights Into Latent Diffusion Model Training. In *Proceedings of the 4th Workshop on Machine Learning and Systems* (Athens, Greece) *(EuroMLSys '24)*. Association for Computing Machinery, New York, NY, USA, 176–183. https://doi.org/10.1145/3642970.3655847

[2] Arden Dertat. 2017. *Applied Deep Learning - Part 3: Autoencoders.* https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

[3] Aparna Dhinakaran. 2023. *Understanding KL Divergence.* https://towardsdatascience.com/understandingkl-divergence-f3ddc8dff254

[4] Geeks for Geeks. 2023. *Training data vs Testing data.* https://www.geeksforgeeks.org/training-data-vs-testing-data/?ref=header_outind

[5] Geeks for Geeks. 2024. *What is Ground Truth in Machine Learning.* https://www.geeksforgeeks.org/what-is-ground-truth-in-machine-learning/

[6] Dominik Roblek Matthew Sharifi Kevin Kilgour, Mauricio Zuluaga. 2019. *Frechet Audio Distance: A Reference-free Metric for Evaluating Music Enhancement Algorithms.* https://www.isca-archive.org/interspeech_2019/kilgour19_interspeech.pdf

[7] Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D. Plumbley. 2024. AudioLDM 2: Learning Holistic Audio Generation With Self-Supervised Pretraining. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 32 (May 2024), 2871–2883. https://doi.org/10.1109/TASLP.2024.3399607