Language Model Based Text-to-Audio Generation: Anti-Causally Aligned Collaborative Residual Transformers

Juncheng Wang[12] **Chao Xu**[2] **Cheng Yu**[2] **Zhe Hu**[1]
Haoyu Xie[2] **Guoqi Yu**[1] **Lei Shang**[2*] **Shujun Wang**[1*]

[1] The Hong Kong Polytechnic University [2] Alibaba Group

Abstract

While language models (LMs) paired with residual vector quantization (RVQ) tokenizers have shown promise in text-to-audio (T2A) generation, they still lag behind diffusion-based models by a non-trivial margin. We identify a critical dilemma underpinning this gap: incorporating more RVQ layers improves audio reconstruction fidelity but exceeds the generation capacity of conventional LMs. To address this, we first analyze RVQ dynamics and uncover two key limitations: 1) orthogonality of features across RVQ layers hinders effective LMs training, and 2) descending semantic richness in tokens from deeper RVQ layers exacerbates exposure bias during autoregressive decoding. Based on these insights, we propose `Siren`, a novel LM-based framework that employs multiple isolated transformers with causal conditioning and anti-causal alignment via reinforcement learning. Extensive experiments demonstrate that `Siren` outperforms both existing LM-based and diffusion-based T2A systems, achieving state-of-the-art results. By bridging the representational strengths of LMs with the fidelity demands of audio synthesis, our approach repositions LMs as competitive contenders against diffusion models in T2A tasks. Moreover, by aligning audio representations with linguistic structures, `Siren` facilitates a promising pathway toward unified multimodal generation frameworks. The code is released at https://github.com/wjc2830/Siren.git.

1 Introduction

Autoregressive language models (LMs) (Vaswani et al., 2017; Brown et al., 2020; Touvron et al., 2023; Chowdhery et al., 2023; Bai et al., 2023) have emerged as the *de facto* paradigm for natural language generation (NLG) (Ouyang et al., 2022; OpenAI, 2023, 2022; Google, 2023; Anthropic, 2023), excelling in modeling discrete, categorical token sequences via next-token prediction.

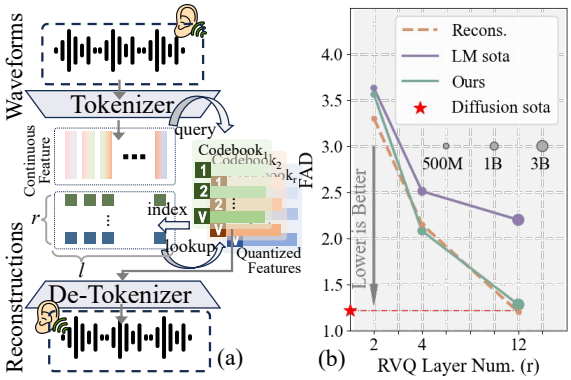

Figure 1: (a) **Residual Vector Quantization Process**, where a waveform is quantized into $r \times l$ discrete tokens. With $r = 1$, it degenerates to naive VQ. (b) **Performance changing curves as r going larger.**

Inspired by the advancements of LMs in NLG, recent works have adopted the LMs for text-to-audio (T2A) generation (Liu et al., 2024a; Ziv et al., 2024; Copet et al., 2024). However, this application presents challenges due to the representational gap between discrete LM tokens and continuous audio waveforms. To bridge this gap, vector quantization (VQ) (Van Den Oord et al., 2017; Razavi et al., 2019) becomes essential to map continuous audio into discrete token sequences, enabling audio generation through *next-token prediction*.

Nevertheless, considering VQ suffers from lossy compression (Esser et al., 2021; Yu et al., 2021, 2023b), *residual vector quantization* (RVQ) has been widely adopted (Wu and Yu, 2019; Défossez et al., 2022; Kumar et al., 2023) (Figure 1-a), where each temporal audio slice is recursively quantized into $r > 1$ residual tokens. Unlike standard VQ, RVQ necessitates *next-r-token prediction* (Lee et al., 2022; Copet et al., 2024; Kreuk et al., 2023).

Despite these advances, LM-based T2A systems (Kreuk et al., 2023; Copet et al., 2024; Ziv et al., 2024) still lag significantly behind diffusion models (Xue et al., 2024; Cheng et al., 2025; Tian et al., 2025; Haji-Ali et al., 2024). For instance, on the AudioCaps benchmark (Kim et al., 2019), state-

*Correspondence

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 26025–26043
November 4-9, 2025

of-the-art LMs (Copet et al., 2024) trail diffusion baselines (Haji-Ali et al., 2024) by 45% in Fréchet Audio Distance (FAD: 2.20 vs. 1.22; Figure 1-b). This gap persists even as RVQ tokenizers achieve high reconstruction fidelity with deeper layers ($r^{\uparrow}$), revealing a critical **dilemma**: while increasing r improves tokenizer reconstruction quality, it overtaxes the generative capacity of LMs. In Figure 1-b, LM-based model struggles to effectively use tokenizer with deeper RVQ layers. This leads to a performance ceiling, where increasing r does not lead to improvements in generation quality, despite enhanced reconstruction capability.

This raises a critical question for high-fidelity audio generation: *how can LMs more effectively predict next-r-codes as r increases?* Our pilot studies of RVQ properties and their impact on LMs reveals two key challenges. (1) Feature Orthogonality: Quantized features from distinct RVQ layers exhibit near-orthogonality in latent space. Standard approaches (Kreuk et al., 2023; Copet et al., 2024) typically employ a shared transformer to predict all RVQ tokens, which forces the model to aggregate diverse gradients from orthogonal feature targets. Nevertheless, this gradient conflict impedes model convergence and limits expressiveness; (2) Semantic Degradation in Deeper RVQ Layers: As RVQ layer goes deeper, the semantic richness within the corresponding quantized features decreases, resulting in a learning difficulty imbalance across different RVQ codebook classifiers. Consequently, such imbalance leads to different fitting degree to each RVQ codes. During autoregressive decoding, this imbalance exacerbates the issue of *exposure bias* (Schmidt, 2019), where errors accumulate as the model transitions from the ground-truth-conditioned training to self-conditioned inference.

Inspired by these findings, we propose `Siren`, anti-cau**S**ally al**I**gned collaborative **RE**sidual tra**N**sformers. To address the first challenge of gradient conflicts caused by orthogonal RVQ features, `Siren` distributes the prediction of r RVQ codes across $r/2$ collaborative transformers, which are trained independently, thus reducing learning diversity within each transformer. To preserve the causal dependencies among RVQ codes, accumulated conditions across models are further introduced to establish collaboration between transformers.

However, this partitioning alone does not address the inherent imbalance in learning difficulty across RVQ codes. The first transformer—assigned to semantic-rich shallow codes—faces high stochasticity in its predictions (Guo et al., 2025), propagating unstable conditions to downstream models and thereby amplifies exposure bias, as described in the aforementioned second challenge of semantic degradation. To mitigate this, we further fine-tune the first transformer using reinforcement learning (RL), to align the outputs from the first transformer towards the conditional *preferences* of subsequent transformers, enhancing decoding stability and fidelity of generated audios.

We conduct extensive experiments and empirically demonstrate that `Siren` achieves the state-of-the-art (*SOTA*) performance comparing with both LM-based and diffusion-based T2A methods. By reconciling the strengths of LMs with RVQ dynamics, our approach bridges the gap between discrete and continuous generation paradigms, paving the way for unified multi-modal generation where discrete tokens serve as a universal interface. In summary, our contributions are three-fold:

- We identify two critical limitations of RVQ in LM-based generation: RVQ layer-wise orthogonality of quantized features and exposure bias due to semantic degradation in deeper layers.
- We propose `Siren`, a novel LM architecture that employ collaborative transformers with RL based anti-causal alignment, to resolve gradient conflicts and distribution drift.
- We demonstrate `Siren`'s *SOTA* performance through extensive experiments, showing that it outperforms both LMs and diffusion based models.

2 Preliminaries

Residual Vector Quantization for High Quality Reconstruction Consider an audio waveform $x \in \mathbb{R}^{l_{wav} \times C_{wav}}$, with C_{wav} as channel number, l_{wav} as duration. To apply Language Models (LMs) to audios generation via next-token prediction, residual vector quantization (RVQ) (Kumar et al., 2023; Ji et al., 2025) tokenizes x into $f = \mathcal{E}(x) \in \mathbb{R}^{l \times C}$ feature through a tokenizer $\mathcal{E}$, with l as the compressed temporal length, C the feature channel. With the continuous feature, it is further quantized into discrete codes $q \in [V]^{r \times l}$, where V is the length of codebook, namely *vocabulary*, r is the number of RVQ layer. Specifically, each temporal step continual feature $f_t, t \leq l$ is decomposed of r layer features in a residual way:

$$f_t^0 = f_t,\ f_t^j = f_t^{j-1} - \hat{f}_t^{j-1},\ q_t^j = \mathcal{Q}^j(f_t^{j-1}),$$
$$\hat{f}_t^j = \texttt{lookup}(Z^j, q_t^j),\ \hat{x}_t = \mathcal{D}(\sum_{j=1}^{j\leq r} \hat{f}_t^j), \tag{1}$$

where $\mathcal{Q}^j$ is an independent quantizer[1] for j^{th} layer that maintains an independent codebook $Z^j \in \mathbb{R}^{V\times C}$ containing V vectors, $\texttt{lookup}(Z, q)$ means finding q^{th} vector in codebook Z. With original audio feature decomposed and quantized individually into $r \times l$ discrete tokens, the reconstructed audio can be derived from the last term in Eq. 1 through a de-tokenizer $\mathcal{D}$. During tokenizer training phase, it is optimized through a general audio reconstruction goal with different loss designs. Empirically, one can derive better reconstruction with larger r.

Enhanced Generation Complexity of RVQ for Transformer Generation With an audio tokenized into above tokens, the mainstream LM-based generators, like AudioGen (Kreuk et al., 2023), have a unified transformer model $\mathcal{F}$ to conduct the *next-time hidden state prediction*, which will be fed into r independent classifier heads $\{\mathcal{C}_j\}_{j=1}^r$ to conduct r times V-way classifications to finish *next-r-code prediction*. Then, the predicted r codes are transferred into corresponding r embeddings, and summed over r to derive input embedding to condition the further next time prediction. According to the above pipeline of generating audios from discrete tokens, decomposing one temporal step feature into r discrete tokens introduces better reconstruction fidelity, but the complexity of transformer generation that conducts r times V-way classifications at each time step is also enhanced. Despite previous works (Kreuk et al., 2023; Copet et al., 2024; Ziv et al., 2024) choose a relatively small r, like $r = 4$, its overall performance lags behind mainstreams, due to being unable to generate audios with r scaled up. Thus, it is critical to answer **how can we better predict multi-RVQ codes when r is large** to derive high fidelity audio generation.

3 Dilemma of Reconstruction Quality and Generation Hardness: A Challenge

To better understand RVQ, we conduct several pilot studies to reveal two key properties, and their corresponding influences to LMs training.

[1]See appendix for detailed process of quantization.

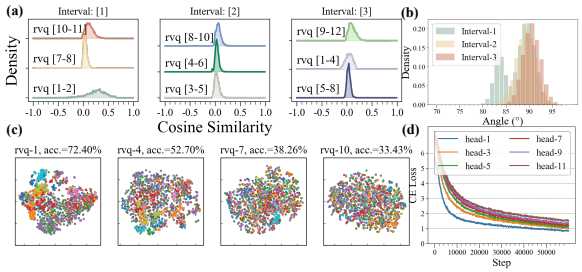

Figure 2: (a) Cosine distributions between quantized features from different RVQ layer. The *Interval* here means the layer interval between two paired features. (b) The angle distribution between gradient vectors that different classifier heads backpropagated to the shared transformer. (c) T-SNE processed quantized features from different RVQ layer, where each audio class has been colored, and corresponding accuracy is on the top. (d) Convergence curves of learning different RVQ codes. *Due to page limitation, we have placed the whole layer results in the Appendix.* Please zoom in for more details.

Property 1: *Orthogonality between two quantized features from different residual layers.* Formally, given j^{th} RVQ layer quantized feature $\hat{f}^j$, it attempts to model the residual difference between continuous feature with quantized one. Thus, the knowledge represented by each residual layer usually is distinct from others. To verify this, we utilize a trained RVQ tokenizer (Kumar et al., 2023) with $r = 12$, and visualize the distribution of cosine similarity between any two quantized features from different residual layers (j_1 and j_2) of the same time-step t on AudioCaps (Kim et al., 2019), like $\texttt{cosine}(\hat{f}_t^{j_1}, \hat{f}_t^{j_2})$. As shown by Figure 2-(a), the cosine similarities are distributed around 0, which represents (approximately) orthogonality.

Influence 1: *Orthogonality introduces diverse learning direction of transformers.* With RVQ decomposing a continuous feature into r (approximately) orthogonal discrete code features, the LM is required to fit r different distributions of $\{p(q_t^j|q_{<t}, c)\}_{j=1}^r$ simultaneously, where c is the textual prompts. Considering the diverse distributions, it incurs diverse learning directions to transformer, which significantly enhances the learning hardness. To demonstrate it, we utilize the LM based AudioGen to predict aforementioned RVQ tokenizer with $r = 12$. Specifically, we visualize the gradients that each independent classifier backpropagates to the last layer of $\mathcal{F}$. As shown by Figure 2-(b), the gradient directions of each classification heads are different. When aggregating the diverse gradients to the shared transformer, it increases the difficulty of convergence.

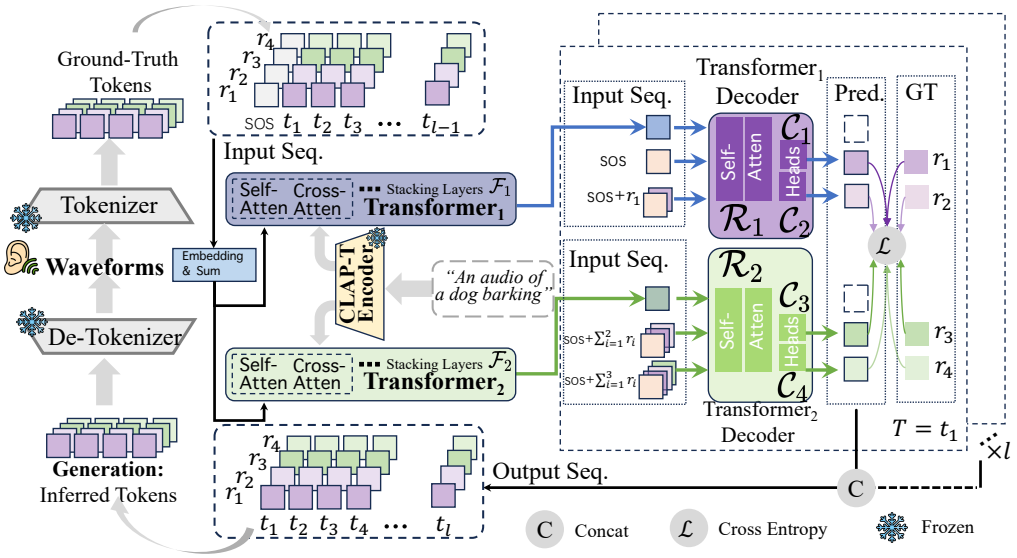

Figure 3: The pipeline of the proposed Siren. Due to spatial restriction, we simplify our real setting ($r = 12$) into a case that $r = 4$ where $r/2 = 2$ parallel transformers are fed with the same input to conduct *next-time feature prediction*, which are then factorized into corresponding two adjacent RVQ codes through independent decoders and classifiers. Then, the predicted tokens can be concatenated together and fed into de-tokenizer to recover into audio waveform. In training phase, teacher forcing is adopted where the contextual tokens are all the ground-truth.

Property 2: *Descending semantic richness when RVQ layer goes deeper.* We further investigate how each residual feature contributes to the final reconstruction. To this end, we first extract temporally complete discrete codes of an audio from different residual layers, *i.e.*, $\{\hat{f}^j \in \mathbb{R}^{l\times C}\}_{j=1}^{r}$. For j^{th} layer feature, on AVSync-15 dataset (Zhang et al., 2024), which has audios from 15 categories, we utilize these embeddings to train r audio classifiers. Following above setting, we utilize the same RVQ tokenizer with $r = 12$. Then, we exhibit classification results in Figure 2-(c). It is shown that as RVQ layer goes deeper, the classification performance goes worse, meaning less semantic richness.

Influence 2: *Imbalanced learning hardness among different residual layer code prediction heads.* As aforementioned, as RVQ layer j being larger, the quantized feature $\hat{f}^j$ contains less semantics. This makes $p(q_t^j|q_{<t})$ harder to fit. To support this, for above trained AudioGen, we visualize their loss curves. It is evident in Figure 2-(d), the convergence becomes slower and worse as layer goes deeper. In autoregressive training, the model is conditioned by ground-truth tokens from previous temporal steps or RVQ layers. While in inference, it is conditioned by self-inferred tokens. Thus, such an imbalanced convergence degree results in worse inconsistency over conditions, which reveals exposure bias.

In sum, the key insights from our pilot studies can be concluded as follows: 1) the orthogonality between RVQ layer features makes one model hard to fit them simultaneously; 2) the imbalanced semantic richness among RVQ layers results in exposure bias among RVQ layers. These hinder the high quality audio generation when r is large.

4 Siren: Anti-Causally Aligned Collaborative Residual Transformers

Based on the above insights, we propose Siren, anti-cau**S**ally al**I**gned collaborative **RE**sidual tra**N**sformers, with pipelines presented in Figure 3& 4, algorithms in Appendix C.3. Firstly, to alleviate the diverse gradient directions among learning different RVQ layers aforementioned in *Influence 1*, Siren distributes r RVQ codes prediction into $r/2$ collaborative transformers, which are trained independently, thus reducing diversity within each transformer, and making optimization easier. To further maintain the causal relationship between codes, accumulated conditions across models are used to establish collaboration between transformers. Secondly, individually training transformers cannot mitigate exposure bias incurred by different fitting hardness, but makes it worse. To address it, we deem the first model that is responsible for 1^{st} and 2^{nd} RVQ codes with most rich semantics, faces high stochasticity during sampling, propagating unstable conditions to downstream models. Thus, we introduce reinforcement learning (RL) to

align the sampled output from the first model to the condition preferences of other models with an anti-causal direction, aiming to enhance the overall performance of collaborative transformers.

4.1 Collaborative Residual Transformers

Isolation for Better Optimization To remedy the above discussed diverse gradient directions to the shared transformer, we distribute r codes prediction tasks to K models, each of which is only responsible for its assigned codes. Concretely, k^{th} model $\mathcal{H}_k$ is individually trained to predict $2k^{th}$ and $(2k+1)^{th}$ layer codes to isolate diverse optimization directions. Given RVQ sets $\{(q_t^1, ..., q_t^r)\}_{t=1}^{l}$, k^{th} model is to fit the distribution of:

$$p_{\theta_k}(q_1^{k_x}, q_2^{k_x}, ..., q_l^{k_x}) = \prod_{t=1}^{l} p_{\theta_k}(q_t^{k_x} | \hat{q}_{<t}^1, ... \hat{q}_{<t}^r), \tag{2}$$

where $k_x = 2k$ or $2k+1$, θ_k denotes the independently maintained parameters of transformers and classifier heads. Hence, compared to having a single model predict the diverse distributions, using K models makes the task more manageable.

Collaboration for Causal Relationship Although isolating the training of each model facilitates better learning target concentration, it incurs sampling cooperation issue during generation. Specifically, let $h_{k,t} = \mathcal{F}_k(q_{<t}^1, ..., q_{<t}^r, c)$ be the predicted next-time step feature from backbone transformer, the generation of code is to sample from the probability over codebook predicted by corresponding transformer, like:

$$p_{\theta_k}(q_t^{k_x}) = \text{softmax}(\mathcal{C}_{k_x}(h_{k,t})). \tag{3}$$

When transformers are independent, the generation among codes from the same temporal step but different residual layers are totally isolated. For example, in Eq. 3, given $k_x > 1$, the information of $q_t^{<k_x}$ is agnostic when predicting $q_t^{k_x}$. But the tokenizer training phase leaves a causal relationship among codes, *i.e.*, it depends on $q_t^{<k_x}$ to derive $q_t^{k_x}$. Thus, the independent prediction in Eq. 3 damages the causality and hinders generation.

To remedy this, additional transformer decoder layers $\mathcal{R}_k$ are introduced to process the causal relationship among residual codes. Concretely, $\mathcal{R}_k$ predicts $(k_x)^{th}$ code conditioned by previous residual codes from the same temporal step, like:

$$p_{\theta_k}(q_t^{k_x}) = \text{softmax}(\mathcal{C}_{k_x}(\mathcal{R}_k(h_{k,t}, q_t^{<k_x}))). \tag{4}$$

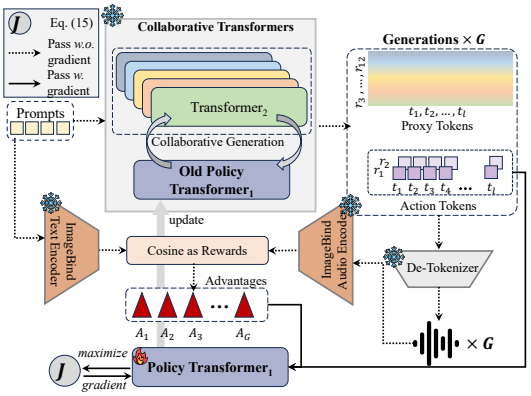

Figure 4: Overall pipeline for reinforcement learning. Given a prompt, first stage trained models generate whole audio tokens, where the output of the first transformer is used as action to align with the conditional preference of other models by an audio quality reward.

In k^{th} model, two adjacent $2k^{th}$ and $(2k+1)^{th}$ codes facilitates modeling the residual context. To shorten the context length of $\mathcal{R}_k$ while keeping the information flow from 1^{st} to $(2k-1)^{th}$ residual layers, we fold codes predicted from other transformers by accumulation, like:

$$\begin{aligned} &\mathcal{R}_k([\cdot, q_t^{2k}, q_t^{2k+1}]) = \\ &\mathcal{R}_k([h_{k,t}, \texttt{sos} + \sum_{j=1}^{2k-1} \hat{f}_t^j, \texttt{sos} + \sum_{j=1}^{2k} \hat{f}_t^j]), \\ &\text{where } \hat{f}_t^j = \texttt{lookup}(Z^j, q_t^j), \end{aligned} \tag{5}$$

This equation represents the input and output sequence of transformer decoder, with a causal inverted triangular attention mask. `sos` is a learnable start token. The training target for k^{th} model is:

$$\begin{aligned} &\mathcal{L}_k = \frac{1}{l} \sum_{t=1}^{l} (\mathcal{L}_{ce}(z_t^{2k}, q_t^{2k}) + \mathcal{L}_{ce}(z_t^{2k+1}, q_t^{2k+1})), \\ &\text{where } z_t^{k_x} = \mathcal{C}_{k_x}(\mathcal{R}_k(\mathcal{F}_k(q_{<t}^1, ..., q_{<t}^r, c), q_t^{<k_x})). \end{aligned} \tag{6}$$

4.2 Anti-Causally Alignment in RL

With individually trained transformers, the exposure bias can be worse, due to imbalance described in *Property 2* and *Influence 2*.

To alleviate this, `Siren` presents a second stage training by aligning output from the first model, *i.e.,* the one for 1^{st} and 2^{nd} codes, to others model's preferences, using reinforcement learning (Figure 4). Considering its difference with causal order of residual codes, we name it as anti-causal alignment, *i.e.,* the conditional preference of later models are known, and set as anchors to be aligned.

We adopt such a direction because: 1) As presented in *Property 2*, shallow RVQ layers contain more semantic information, leading to more stochastic sampling results (Guo et al., 2025); 2) According to *Influence 2*, learning deeper codes is harder, so fine-tuning the first transformer would be more accessible; 3) In autoregressive generation, initial tokens play a key role in determining the overall semantics of the sequence (Barbero et al., 2025).

Concretely, we define a Markov Decision Process (MDP) (Ouyang et al., 2022) of tokens output from first transformer, where 1^{st} and 2^{nd} RVQ codes as the action sequence and tokens from other transformers as proxy tokens for reward. For each temporal step, given a state $\mathbf{s}_t = (\hat{q}^1_{<t}, ..., \hat{q}^r_{<t}, c)$, the first transformer generates action $\mathbf{q}_t = (\hat{q}^1_t, \hat{q}^2_t)$ with a policy $\pi(\cdot|\mathbf{s}_t)$, defined as:

$$\pi(\cdot|\mathbf{s}_t) = \texttt{softmax}(\mathcal{H}_1(\hat{q}^1_{<t}, ..., \hat{q}^r_{<t}, c)), \quad (7)$$

where $\mathcal{H}_1 = \mathcal{C}_1 \times \mathcal{R}_1 \times \mathcal{F}_1$.

Then, we need to compute the reward for this generation. Recap our target is to align this action $\mathbf{q} \in [V]^{2\times l}$ to the preference of other transformers, yet we cannot directly derive quantitative metric to this *preference*, thus we use the finally generated audio quality as a proxy reward. Technically, we introduce the de-tokenizer $\mathcal{D}$ and ImageBind model (Girdhar et al., 2023) ϕ_{text} and ϕ_{audio} to compute the reward as:

$$R = \text{cosine}(\phi_{audio}(\mathcal{D}(\text{cat}(\mathbf{q}, \mathbf{q}_{prox}))), \phi_{text}(c)), \quad (8)$$

where $\mathbf{q}_{prox} \in [V]^{(r-2)\times l}$ are the proxy tokens.

Later, we use the proximal policy optimization (PPO) (Schulman et al., 2017) with a value-model free modification (Shao et al., 2024; Yu et al., 2025; Yuan et al., 2025; Lin et al., 2025) to seek efficiency. For a specific prompt-generated audio pair, the policy $\pi_{\theta_{old}}$ samples a group of individual action sequences $\{\mathbf{q}_i\}_{i=1}^G$. Then, the advantage of the i^{th} action sequence is calculated by normalizing the group-wise rewards $\{R_i\}_{i=1}^G$:

$$A_i = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^G)}{\text{std}(\{R_i\}_{i=1}^G)}. \quad (9)$$

The training objective $\mathcal{J}(\theta_1)$ is maximized as:

$$\mathbb{E}_{\{(q^1_t)_i\}_{i=1}^G \sim \pi_{\theta_{old}}(\cdot|\mathbf{q}_{<t}, c), \{(q^2_t)_i\}_{i=1}^G \sim \pi_{\theta_{old}}(\cdot|\mathbf{q}_{<t}, c, q^1_t)}$$
$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{l}\sum_{t=1}^{l}\frac{1}{2}\sum_{j=1}^{2}\min(r_{ij}A_i, v_{ij}A_i), \text{s.t. } |A_i| \geq \gamma, \quad (10)$$

where $v_{ij} = \text{clip}(r_{ij}, 1-\epsilon_d, 1+\epsilon_u)$,

$$r_{ij} = \frac{\pi_{\theta_1}((q^j_t)_i|(\mathbf{q}_{<t})_i, c, (q^{j-1}_t)_i)}{\pi_{\theta_{old}}((q^j_t)_i|(\mathbf{q}_{<t})_i, c, (q^{j-1}_t)_i)}, \quad (11)$$

where ϵ_d and ϵ_u are two coefficients to restrict the update the model in each step, and γ is a threshold to filter out those action sequences with lower absolute rewards to make model concentrate on sequences with more extinguish rewards.

After fine-tuning the first transformer with above reinforcement objective to align the condition preference of other transformers, it alleviates the exposure bias issue to enhance generation quality.

5 Experiment

5.1 Settings

Dataset and Training Details. Following previous works (Xue et al., 2024; Kreuk et al., 2023), we collect audios from AudioSet (Gemmeke et al., 2017), AudioCaps (Kim et al., 2019), Clotho (Drossos et al., 2020), ESC50 (Piczak, 2015), FreeSound (Font et al., 2013), VGGSound (Chen et al., 2020a), AVSync-15 (Zhang et al., 2024), BBC Sound Effects [2], and SoundBible [3]. We utilize QWen-2 Audio (Chu et al., 2024) to annotate those audios without textual prompts, which are polished through QWen-2.5 (Bai et al., 2023). Then, we filter the text-audio pairs by CLAP score (Wu et al., 2023b) with a threshold. In this paper, we utilize three levels of thresholds to obtain data collects for $1.6B$, $3.1B$, and reinforcement training. See appendix for detailed data engine process. As for training, we utilize a RVQ tokenizer (Kumar et al., 2023) with $r = 12$. Our first stage training is conducted through $150K$, and $500K$ steps for two model size variants. Then, the first transformer is further fine-tuned by additional $2K$ steps. The trained models are then tested on AudioCaps dataset.

Metrics Following previous works (Liu et al., 2023), we employ Fréchet Distance (FD), Fréchet Audio Distance (FAD), Kullback-Leibler (KL) divergence, Inception Score (IS), and CLAP score. Please see appendix for details in computation.

5.2 Main Results

We compare `Siren` with both diffusion- and LMs-based T2A generators in Table 1. For fair com-

[2] https://sound-effects.bbcrewind.co.uk/

[3] https://soundbible.com/

Method	#Parameter	#Train Sample	# Gen.Time	FAD↓	FD↓	ISC↑	KL↓	CLAP↑
continuous tokens (diffusion based)								
AudioLDM2 (Liu et al., 2024a)	712M	760K	53s	1.82	31.02	8.46	1.69	15.73
Auffusion (Xue et al., 2024)	1.1B	470K	58s	2.22	18.73	12.73	1.39	21.35
TANGO-Flux (Hung et al., 2024)	515M	445K	16s	2.25	17.99	12.81	1.41	24.65
ETTA (Lee et al., 2024)	-	2.77M	-	1.89	11.13	**15.05**	**1.26**	-
GenAU (Haji-Ali et al., 2024)	1.25B	811K	52s	**1.22**	15.86	11.90	<u>1.28</u>	24.07
MMAudio (Cheng et al., 2025)	1.03B	951K	7s	4.21	13.63	12.45	1.40	**30.63**
AudioX (Tian et al., 2025)	1.1B	330K (+5.9M)	58s	1.63	11.67	12.44	1.36	<u>28.14</u>
discrete tokens (autoregressive / masked transformer based)								
AudioGen^{+} (Kreuk et al., 2023)	1.6B	101K	11s	4.09	29.65	7.86	1.95	13.24
MagNet* (Ziv et al., 2024)	1.5B	-	4s	3.64	26.11	8.58	1.88	10.54
DelayPattern* (Copet et al., 2024)	1.5B	-	11s	2.51	12.47	10.62	1.96	14.23
Siren (ours)	1.6B	100K+1K	13s	1.35	<u>10.65</u>	12.85	1.33	24.18
DelayPattern^{+} (Copet et al., 2024)	3.3B	437K	25s	2.20	12.50	11.66	1.70	16.58
Siren (ours)	3.1B	436K+1K	25s	<u>1.28</u>	**10.35**	<u>13.93</u>	1.36	25.64

Table 1: Main results on AudioCaps test-set. *#Gen. Time* means the cost time to conduct inference with a batch size of 16 (effective sample count of 8 when using CFG) using NVIDIA-L20. The best performed metric is in **bold**, and the second best is <u>underlined</u>. * means we use officially available models, while $^{+}$ means we train them from scratch due to unavailable models.

parison, we report model size, training data scale, latency per batch, and employ CLAP-score-based reject sampling following Haji-Ali et al., 2024.

As shown in Table 1, conventional LM-based models underperform diffusion counterparts by wide margins (e.g., 2.20 vs. 1.22 FAD) despite comparable parameter counts, underscoring their struggle to leverage RVQ's potential. In contrast, `Siren` achieves state-of-the-art fidelity by deploying a deep RVQ tokenizer ($r = 12$) while maintaining efficient autoregressive decoding. This narrows the gap with diffusion models, demonstrating that discrete token modeling can rival continuous-domain approaches. Notably, `Siren` trails MMAudio (Cheng et al., 2025) and AudioX (Tian et al., 2025) in CLAP score by 2–4 points. We attribute this to their multi-modal training (text, video), which enriches semantic grounding even when inferring with text-only prompts.

# para.	Embed.	Dim	Main Layer	Main Para.	Decoder Layer	Decoder Para.	Head Count	Head Para.
1,632M	12.5 M	1024	14	231M	2	25M	12	12.6
3,084M	12.5M	1024	24	396M	8	103M	12	12.6

Table 2: Details of transformer' s architecture.

Architecture Details In this section, we present a comprehensive overview of Siren's architectural blueprint, detailing each core component that constitutes the full generative pipeline. Our design comprises four principal modules: (1) Multimodal Embedding Layers, responsible for encoding heterogeneous input modalities (e.g., text, audio tokens, or conditioning signals) into a unified latent space; (2) Main Transformer Blocks, which form the backbone of the model and perform deep contextual reasoning over the embedded sequences; (3) Decoder Transformers, specialized modules that progressively refine latent representations into structured output tokens; and (4) Prediction Heads, lightweight output layers that map the decoder's final representations to target audio token distributions or waveform parameters.

5.3 Ablation Study

Setting	#Parameter	FAD	FD	CLAP
Small Single	530M	8.22	40.44	4.81
Base Single	1.6B	3.88	25.18	7.79
Large Single	3.3B	1.92	11.12	15.64
Isolated-Small	1.6B	1.44	12.09	17.07
Isolated-Large	3.1B	1.21	9.90	20.24

Table 3: Ablating different architecture options.

Ablating among architectures In this subsection, we compare different architecture options, including small single LM, larger single LM, small isolated transformers, and larger isolated transformers. To conduct a fair comparison, we benchmarked our parallel architecture against a series of single Transformer models whose total parameter counts closely match those of our multi-LM setup. Cru-

Collab.	Residual Con.	Accumulated Cond.	RL Align	FD	ISC	CLAP
✗	✗	✗	✗	29.65	7.86	13.24
✓	✗	✗	✗	*itg*		
✓	✓	✗	✗	14.66	8.90	15.09
✓	✗	✓	✗	12.09	10.70	17.07
✓	✗	✓	✓	10.65	12.85	24.18

Table 4: Ablating among major modules.

RVQ Prediction	Loss Mean	Loss Ratio	FAD
Indpdt.	1.19	1.95	4.09
AR	0.88	2.17	4.19
Accumu. AR	0.80	2.15	3.88
Collaborative (ours)	0.29	1.55	1.44

Table 5: Ablating different conditional strategies to model the causal relationship of inter-RVQ layer codes.

cially, the only architectural difference in this comparison is the isolation: ours uses separate models, while the baseline uses a single transformer.

The results in Table 3 demonstrate that our isolated LLM architecture outperforms the single-model baseline across key evaluation metrics. This suggests that the parallel, modular design itself—not just increased scale—provides a fundamental advantage, likely due to enhanced specialization or reduced interference between tasks.

Ablating Method Module.

To validate `Siren`'s effectiveness, we present results in Table 4. Row 1 shows a baseline without specific design, yielding suboptimal performance. Introducing collaboration in row 2 allows models to fit their respective codes during training, but disrupts RVQ's causal relationship, thus inability to generation (*itg*). Conditions are added to resolve this, using either the previous layer's code (residual approach) or the sum of all previous codes (accumulated approach). The latter in row 4 proves superior, so it's adopted. To reduce exposure bias, RL Alignment is added in row 5, greatly enhancing overall performance.

Ablating the Condition of RVQ Prediction

In Table 5, we focus on two metrics to further verify inter-model conditioning: Loss Mean, indicating training fit by the smallest average of RVQ layer code losses; and Loss Ratio, indicating training balance by the ratio of the 12^{th} to the

Model Distribution	#Single Para.	FD	ISC	CLAP	Loss Ratio
1-12	1.6B	25.18	7.79	11.43	2.15
2-6	800M	23.89	6.62	10.84	2.09
4-3	400M	15.90	11.03	13.24	1.74
6-2	270M	12.09	10.70	17.07	1.55
12-1	140M	*itg*			

Table 6: Ablating among different model distributions.

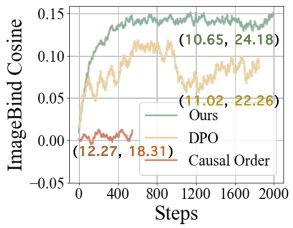

Figure 5: Rewards (ImageBind Cosine) changing curves among different reinforcement learning strategy. The figure pair aside each curve means (FD, CLAP Score).

1^{st} code loss. Independent heads (row 1) struggle to converge, with deeper layers facing more challenges. Incorporating conventional AR mode (row 2) enhances fit but makes deeper code learning harder. Accumulation mode (row 3) conditions on rich semantic cues, greatly improving fit and balance. Combined with collaborative method (row 4) achieves excellent code balance and overall fit.

Align Direction	Num. Model	# Train Parameter	CLAP	ISC	FD
Anti-Causal	1	270M	24.18	12.85	10.65
Anti-Causal	2	540M	25.12	12.77	10.80
Causal	1	270M	18.31	9.55	12.27

Table 7: Ablating reinforcement learning components.

Ablating the Number of RVQ Codes Per Model To determine the number of codes each model handles, we present results for five variants in the Table 6. For efficiency, more models mean fewer parameters per model. For performance, the 1-12 shows significant degradation, as a single model cannot handle the diverse code distribution, and deeper layers increase prediction difficulty, seen in the larger Loss Ratio. Reducing codes per model improves focus and performance, with the 6-2 configuration being optimal. Notably, 12-1 fails to generate reasonable audio because its single model capacity is too limited to handle all temporal cues.

Ablating the Anti-Causal Alignment We perform ablations on the optimization direction and the number of models optimized in anti-causal alignment. Firstly, a causal direction, *i.e.*, training the last model to align with earlier models' preferences, results in reward around zero in Figure 5 and Table 7 row 3 confirms ineffectiveness. This is because the last two codes contain the least semantics, thus having minimal impact on the performance. Secondly, training the first two models under the anti-causal strategy achieves comparable results as shown in row 2, suggesting one model is sufficient. Besides, combining this strategy with DPO (Rafailov et al.,

2023) also yields improvements, but not as effectively as our RL method, as Figure 5 depicts.

Method	General Fidelity	ME. Fidelity	OOD Fidelity	General Ins.	ME. Ins.	OOD Ins.
DelayPattern	27.3	25.4	29.3	15.2	17.5	28.3
AudioX	28.1	35.9	33.5	26.5	39.2	33.8
Ours	44.6	38.7	37.2	58.3	43.3	37.9

Table 8: Results from user study.

5.4 User Study

To better validation, we conducted a comprehensive user study involving 20 human experts, who evaluated audio samples generated by three distinct systems: AudioX (representing the state-of-the-art in diffusion models), DelayPattern (the leading language model-based approach), and our proposed method, Siren. The results have been exhibited in Table 8. In each evaluation round, raters were presented with a set of three audio clips—one from each system—and were asked to rank them according to two key dimensions: Fidelity Quality, which assesses overall audio realism and sound clarity, and Instruction Following, which measures how accurately the audio reflects the given prompt.

To further probe the models' capabilities under more challenging conditions, we curated three specialized subsets of prompts. The first, General Scenes, consists of 30 randomly selected prompts from the AudioCaps dataset to establish a baseline. The second, Multi-Event, includes 30 prompts from AudioSet that describe scenes with more than two concurrent sound events—selected with the assistance of the Qwen3-235B-A22B (Yang et al., 2025) LLM to ensure complexity. The third, OOD Prompts, comprises 30 synthetically generated prompts, also crafted using Qwen3-235B-A22B, that depict rare or unusual acoustic scenarios designed to test the models' generalization and compositional reasoning by combining unlikely or previously unseen events.

The results, measured by the percentage of times each method was ranked best across all evaluations, show that Siren outperforms AudioX and DelayPattern in fidelity and instruction following.

6 Related Work

Diffusion-Based Audio Generation Denoising diffusion models (Ho et al., 2020; Song and Ermon, 2019; Song et al., 2020; Dhariwal and Nichol, 2021; Lu et al., 2022; Rombach et al., 2022; Peebles and Xie, 2023) excel in continuous-domain generation by iteratively refining noisy signals into structured outputs, making them a natural fit for audio synthesis. Prior work applies these models to audio via latent-space diffusion: 1D tokenizer operate on waveform embeddings (Evans et al., 2025; Lee et al., 2024), while 2D architectures process mel-spectrograms using U-Nets (Liu et al., 2022, 2023, 2024a; Xue et al., 2024; Evans et al., 2024; Xing et al., 2024; Du et al., 2023; Liu et al., 2024b; Agostinelli et al., 2023; Majumder et al., 2024; Wang et al., 2025) or diffusion transformers (DiT) (Lee et al., 2024; Valle et al., 2025). Recent advance (Cheng et al., 2025; Valle et al., 2025) integrate flow matching (Lipman et al., 2023; Liu et al., 2025) to accelerate sampling, achieving state-of-the-art fidelity. However, reliance on continuous latent representations creates a fundamental divergence from discrete token-based paradigms like LMs, complicating efforts toward unified multimodal frameworks (Wu et al., 2023a; Fei et al., 2024; Xu et al., 2025; Du et al., 2025).

Language Models for Audio Synthesis Inspired by successes in NLP, LM-based audio generators map waveforms to discrete tokens via vector quantization (VQ) (Chen et al., 2020b; Chang et al., 2022; Li et al., 2023; Yu et al., 2023a; Zheng et al., 2022; Chang et al., 2023; Tian et al., 2024). Early approaches paired VAEs with LMs, but VQ's lossy compression limited audio fidelity (Yu et al., 2023b; Mentzer et al., 2023). Residual VQ (RVQ) (Wu and Yu, 2019; Défossez et al., 2022; Kumar et al., 2023) emerged as a dominant alternative. While RVQ-enhanced LMs improve fidelity, their performance remains sensitive to the choice of RVQ layers (r): deeper hierarchies strain autoregressive modeling due to feature orthogonality and exposure bias. Our work directly addresses these limitations through architectural innovations that disentangle RVQ layer-specific learning while preserving cross-layer coherence.

7 Conclusion

To enable LMs to predict multi-RVQ codes with a deep RVQ layer in text-to-audio generation, we propose `Siren`, a novel framework that employs collaborative transformers with anti-causal alignment. By disentangling RVQ code-specific conditional learning objectives and harmonizing cross-model conditional reference alignment via reinforcement learning, `Siren` achieves SOTA performance.

Limitations

While Siren advances autoregressive text-to-audio generation, three limitations merit discussion: (1) Training Efficiency: Partitioning r RVQ codes across $r/2$ isolated transformers mitigates gradient conflicts and enables training billion-parameter models (e.g., 1.6B–3.1B) on consumer-grade GPUs (e.g., lower to 24GB VRAM). However, this design increases wall-clock training time: deploying Siren requires sequential training of up to six transformer modules without sufficient parallelization when no available devices. Future work will explore hybrid RVQ tokenizers that achieve comparable reconstruction fidelity with fewer layers (r), reducing both model number and training overhead. (2) Model Size vs. Semantic Richness: Although Siren matches diffusion models in inference speed and surpasses them in fidelity metrics (e.g., FD), its parameter count exceeds diffusion counterparts. We attribute this to the inherent trade-off between RVQ's discrete tokens (low semantic density) and LM scalability: richer semantics per token could enable smaller models. Improving tokenizers to encode higher-level acoustic semantics—akin to linguistic units in text—remains a critical direction. (3) Data Scaling: Our experiments use a curated dataset smaller than those in prior work (e.g., GenAU, MMAudio). While rigorous filtering ensures quality, expanding data diversity—particularly with multi-modal (text, video) or long-form audio—could enhance semantic grounding and temporal coherence. Future efforts will prioritize scalable data collection pipelines to balance quality and quantity.

Acknowledgement

This work was partially supported by RGC Collaborative Research Fund (No. C5055-24G), the Start-up Fund of The Hong Kong Polytechnic University (No. P0045999), the Seed Fund of the Research Institute for Smart Ageing (No. P0050946), and Tsinghua-PolyU Joint Research Initiative Fund (No. P0056509), and PolyU UGC funding (No. P0053716).

References

Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, and 1 others. 2023. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*.

Anthropic. 2023. Claude. https://www.anthropic.com/index/introducing-claude.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Federico Barbero, Alvaro Arroyo, Xiangming Gu, Christos Perivolaropoulos, Michael Bronstein, Razvan Pascanu, and 1 others. 2025. Why do llms attend to the first token? *arXiv preprint arXiv:2504.02732*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, and 1 others. 2023. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. 2022. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325.

Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. 2020a. Vggsound: A large-scale audio-visual dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 721–725. IEEE.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. 2020b. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR.

Ho Kei Cheng, Masato Ishii, Akio Hayakawa, Takashi Shibuya, Alexander Schwing, and Yuki Mitsufuji. 2025. Taming multimodal joint training for high-quality video-to-audio synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, and 1 others. 2024. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2024. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2022. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. 2020. Clotho: an audio captioning dataset. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 736–740. IEEE.

Henghui Du, Guangyao Li, Chang Zhou, Chunjie Zhang, Alan Zhao, and Di Hu. 2025. Crab: A unified audio-visual scene understanding model with explicit cooperation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Yuexi Du, Ziyang Chen, Justin Salamon, Bryan Russell, and Andrew Owens. 2023. Conditional generation of audio from video via foley analogies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2436.

Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883.

Zach Evans, Julian D Parker, CJ Carr, Zack Zukowski, Josiah Taylor, and Jordi Pons. 2024. Long-form music generation with latent diffusion. *arXiv preprint arXiv:2404.10301*.

Zach Evans, Julian D Parker, CJ Carr, Zack Zukowski, Josiah Taylor, and Jordi Pons. 2025. Stable audio open. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Hao Fei, Han Zhang, Bin Wang, Lizi Liao, Qian Liu, and Erik Cambria. 2024. EmpathyEar: An open-source avatar multimodal empathetic chatbot. In *Annual Meeting of the Association for Computational Linguistics*.

Frederic Font, Gerard Roma, and Xavier Serra. 2013. Freesound technical demo. In *ACM Multimedia Conference, MM '13, Barcelona, Spain, October 21-25, 2013*, pages 411–412. ACM.

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*, pages 776–780. IEEE.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. 2023. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190.

Google. 2023. Bard. https://bard.google.com/.

Ziyao Guo, Kaipeng Zhang, and Michael Qizhe Shieh. 2025. Improving autoregressive image generation through coarse-to-fine token prediction. *arXiv preprint arXiv: 2503.16194*.

Moayed Haji-Ali, Willi Menapace, Aliaksandr Siarohin, Guha Balakrishnan, Sergey Tulyakov, and Vicente Ordonez. 2024. Taming data and transformers for audio generation. *arXiv preprint arXiv:2406.19388*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, and 1 others. 2017. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Chia-Yu Hung, Navonil Majumder, Zhifeng Kong, Ambuj Mehrish, Amir Ali Bagherzadeh, Chuan Li, Rafael Valle, Bryan Catanzaro, and Soujanya Poria. 2024. Tangoflux: Super fast and faithful text to audio generation with flow matching and clap-ranked preference optimization. *arXiv preprint arXiv:2412.21037*.

Shengpeng Ji, Ziyue Jiang, Xize Cheng, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Ruiqi Li, Ziang Zhang, Xiaoda Yang, Rongjie Huang, Yidi Jiang, Qian Chen, Siqi Zheng, Wen Wang, and Zhou Zhao. 2025. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. In *International Conference on Learning Representations*.

Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. 2018. Fr\'echet audio distance: A metric for evaluating music enhancement algorithms. *arXiv preprint arXiv:1812.08466*.

Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. 2019. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132.

Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. 2020. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894.

Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. 2023. Audiogen: Textually guided audio generation. In *International Conference on Learning Representation*.

Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. 2023. High-fidelity audio compression with improved rvqgan. *Advances in Neural Information Processing Systems*, 36:27980–27993.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532.

Sang-gil Lee, Zhifeng Kong, Arushi Goel, Sungwon Kim, Rafael Valle, and Bryan Catanzaro. 2024. Etta: Elucidating the design space of text-to-audio models. *arXiv preprint arXiv:2412.19351*.

Tianhong Li, Huiwen Chang, Shlok Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. 2023. Mage: Masked generative encoder to unify representation learning and image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2142–2152.

Zhihang Lin, Mingbao Lin, Yuan Xie, and Rongrong Ji. 2025. Cppo: Accelerating the training of group relative policy optimization-based reasoning models. *arXiv preprint arXiv:2503.22342*.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. 2023. Flow matching for generative modeling. In *International Conference on Learning Representations*.

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 preprint arXiv:2301.12503*.

Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. 2024a. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen, and Zhou Zhao. 2022. Diffsinger: Singing voice synthesis via shallow diffusion mechanism. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 11020–11028.

Peng Liu, Dongyang Dai, and Zhiyong Wu. 2025. Rfwave: Multi-band rectified flow for audio waveform reconstruction. In *International Conference on Learning Representations*.

Xiulong Liu, Kun Su, and Eli Shlizerman. 2024b. Tell what you hear from what you see–video to audio generation through text. *arXiv preprint arXiv:2411.05679*.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787.

Navonil Majumder, Chia-Yu Hung, Deepanway Ghosal, Wei-Ning Hsu, Rada Mihalcea, and Soujanya Poria. 2024. Tango 2: Aligning diffusion-based text-to-audio generations through direct preference optimization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 564–572.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. 2023. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*.

OpenAI. 2022. Chatgpt. https://openai.com/blog/chatgpt.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205.

Karol J. Piczak. 2015. ESC: dataset for environmental sound classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM '15, Brisbane, Australia, October 26 - 30, 2015*, pages 1015–1018. ACM.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. 2019. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. *arXiv preprint arXiv:1910.00292*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. 2024. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*.

Zeyue Tian, Yizhu Jin, Zhaoyang Liu, Ruibin Yuan, Xu Tan, Qifeng Chen, Wei Xue, and Yike Guo. 2025. Audiox: Diffusion transformer for anything-to-audio generation. *arXiv preprint arXiv:2503.10522*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Rafael Valle, Rohan Badlani, Zhifeng Kong, Sang-gil Lee, Arushi Goel, Sungwon Kim, Joao Felipe Santos, Shuqi Dai, Siddharth Gururani, Aya Aljafari, and 1 others. 2025. Fugatto 1: Foundational generative audio transformer opus 1. In *International Conference on Learning Representations*.

Aaron Van Den Oord, Oriol Vinyals, and 1 others. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Juncheng Wang, Chao Xu, Cheng Yu, Lei Shang, Zhe Hu, Shujun Wang, and Liefeng Bo. 2025. Synchronized video-to-audio generation via mel quantization-continuum decomposition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. 2023a. Next-gpt: Any-to-any multimodal llm. *arXiv preprint arXiv:2309.05519*.

Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2023b. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Ze-bin Wu and Jun-qing Yu. 2019. Vector quantization: a review. *Frontiers of Information Technology & Electronic Engineering*, 20(4):507–524.

Yazhou Xing, Yingqing He, Zeyue Tian, Xintao Wang, and Qifeng Chen. 2024. Seeing and hearing: Open-domain visual-audio generation with diffusion latent aligners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7151–7161.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. 2025. Qwen2.5-omni technical report. *arXiv, preprint arXiv: 2503.20215*.

Jinlong Xue, Yayue Deng, Yingming Gao, and Ya Li. 2024. Auffusion: Leveraging the power of diffusion and large language models for text-to-audio generation. *arXiv preprint arXiv:2401.01044*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. 2021. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and 1 others. 2023a. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10459–10469.

Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, and 1 others. 2023b. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, and 1 others. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Lin Zhang, Shentong Mo, Yijing Zhang, and Pedro Morgado. 2024. Audio-synchronized visual animation. In *European Conference on Computer Vision*, pages 1–18. Springer.

Chuanxia Zheng, Tung-Long Vuong, Jianfei Cai, and Dinh Phung. 2022. Movq: Modulating quantized vectors for high-fidelity image generation. *Advances in Neural Information Processing Systems*, 35:23412–23425.

Alon Ziv, Itai Gat, Gael Le Lan, Tal Remez, Felix Kreuk, Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2024. Masked audio generation using a single non-autoregressive transformer. *arXiv preprint arXiv:2401.04577*.

A Data Engine

We detail our data pipeline for curating high-quality audio-text pairs, comprising four stages: collection, preprocessing, captioning, and filtering.

Data Collection Following established practices in T2A research (Xue et al., 2024; Haji-Ali et al., 2024), we aggregate audio from public repositories. For video-derived audio, we extract and segment raw audio tracks, retaining only segments with detectable acoustic activity (SNR > 6dB). This yields **2.1 million unlabeled audio clips** spanning 2–100 seconds.

Preprocessing To align inputs with our RVQ tokenizer (Kumar et al., 2023), we standardize waveforms as follows: Channel Conversion: Convert stereo/multi-channel audio to mono. Resampling: Downsample to 16 kHz to match the tokenizer's Nyquist frequency. Segmentation: Split clips exceeding 10 seconds into fixed-length 10s chunks using a sliding window (stride=10s, no overlap), discarding residual segments (<10s). Shorter clips are zero-padded to 10s. This produces **4.2 million standardized 10s audio segments**, doubling the initial dataset through segmentation.

Captioning Only data from AudioCaps, Clotho include human-annotated captions. For the remaining, we generate synthetic captions using QWen-2 Audio, a multimodal LLM fine-tuned for acoustic understanding. Each audio segment receives five candidate captions via the prompt:

```
Please describe this audio
in detail, including events,
timbres, temporal structure, and
emotional tone.
```

To refine captions, we apply Qwen-2.5 LLM with rule-based filtering: Removal: Eliminate non-descriptive text (e.g., transcribed speech, non-English content). Normalization: Standardize acoustic terms (e.g., "barking" $\rightarrow$ "dog bark," "low-pitched rumble" $\rightarrow$ "engine noise"). This ensures captions are concise, objective, and acoustically grounded.

Data Filtering We compute audio-text alignment scores using CLAP , retaining pairs where:

$$\text{CLAP-score} = \frac{\text{Audio} \cdot \text{Text}}{|\text{Audio}||\text{Text}|} \geq \tau$$

For the 1.6B `Siren` model, we set $\tau = 0.4$ yielding 117k high-confidence pairs. From these, we:

Reserve the top 1k (highest CLAP-score) for reinforcement learning (RL) fine-tuning. Randomly select 100k for base model pretraining. To scale to the 3.1B model, we lower τ to 0.35, adding 320k moderately aligned pairs. Combined with the initial 116k, this forms a 436k training corpus, balancing quality and diversity.

B Experimental Details

Embedding Layer Setup The RVQ tokenizer from Kumar et al., 2023 employs low-dimensional codebooks (dim=8) followed by a post-projector to restore waveform fidelity. To adapt these tokens for transformer-based generation, we retain all quantization layers and introduce 12 parallel MLP projectors to map discrete token indices into high-dimensional embeddings compatible with the transformer's hidden dimension. This preserves quantization fidelity while ensuring seamless integration with the autoregressive architecture.

Textual Condition Encoder Text prompts are encoded into embeddings via the frozen CLAP-Text encoder (Wu et al., 2023b). These embeddings are injected into the transformer through cross-attention layers in all decoder blocks, where CLAP embeddings serve as keys/values and audio tokens as queries.

Training Protocol This training employs the AdamW optimizer with a 3e-4 learning rate and 24 batch size on each NVIDIA L20 GPU, which costs around 600 GPU hours for $1.6B$ variant, and $2.5K$ for $3.1B$ variant. To conduct data augmentation, we randomly crop 6s segments from 10s audio, retaining only crops with CLAP similarity higher than 0.20 to their original caption.

Inference & Sampling Autoregressive decoding is accelerated using KV-caching. While classifier-free guidance (CFG) is common in generative models, its integration with reinforcement learning-based anti-causal alignment remains unstable; thus, we disable CFG. Bridging this gap—enabling controllable generation via guidance in LM-based systems—is a key direction for future work.

C Additional Technical Details

C.1 Metrics

The main objective evaluation metrics we use are Frechet Distance (FD), Inception Score (IS), and Kullback–Leibler (KL) divergence. These metrics

are based on the state-of-the-art audio classifier PANNs (Kong et al., 2020). Specifically, FD, analogous to the Frechet Inception Distance in image generation, measures the similarity between generated and target audio samples. IS evaluates both sample quality and diversity. KL divergence is computed at the paired-sample level and averaged for the final score. We also report Frechet Audio Distance (FAD) (Kilgour et al., 2018), which follows a similar principle but uses VGGish (Hershey et al., 2017)—a potentially less effective classifier than PANNs. In addition, the CLAP score (Wu et al., 2023b) is adopted to measure the semantic alignment of audio-text.

C.2 Tokenizer

Consider an audio waveform $x \in \mathbb{R}^{l_{wav} \times C_{wav}}$, where C_{wav} is the number of channel, l_{wav} is the duration length. The goal of tokenization is to compress x into the latent space $f \in \mathbb{R}^{l \times C}$, where empirically we have $C > C_{wav}$, and $l < l_{wav}$. To apply autoregressive transformer modeling to audios via next-token prediction, we must tokenize an audio into l *discrete* tokens.

Vector Quantization (VQ) To this end, VQ firstly converts an audio into continuous tokens (feature) $f \in \mathbb{R}^{l \times C}$, which is then quantized into discrete tokens $q \in [V]^l$:

$$f = \mathcal{E}(x), \qquad q = \mathcal{Q}(f), \tag{12}$$

where $\mathcal{E}$ denotes a tokenizer, $\mathcal{Q}$ a quantizer. The quantizer typically includes a learnable codebook $Z \in \mathbb{R}^{V \times C}$ containing V vectors. The quantization process can be described as mapping each entry t ($t \leq l$) of feature f_t to the code index q_t of its nearest code in the Euclidean sense:

$$q_t = (\arg\min_{v \in [V]} \ \|\texttt{lookup}(Z, v) - f_t\|_2) \in [V], \tag{13}$$

where $\texttt{lookup}(Z, v)$ denotes taking the v^{th} vector in codebook Z. Then, Z is looked up by every entry q_t independently to retrieve corresponding code $\hat{f}_t \in \mathbb{R}^{1 \times C}$, which is an approximate of original feature f_t. To recover the $\hat{f}$ into soundable waveform, a de-tokenizer $\mathcal{D}$ is applied, where the whole modules are trained with an audio reconstruction goal:

$$\hat{x} = \mathcal{D}(\hat{f}), \qquad \arg\min_{\mathcal{E},\mathcal{Q},\mathcal{D}} \mathcal{L}_{\texttt{Recon}}(\hat{x}, x), \tag{14}$$

where $\hat{x}$ is the reconstructed version of audio x, and $\mathcal{L}_{\texttt{Recon}}$ is a general representation, with different designs in existing works.

Audio Generation by Autoregressive Transformer With an audio x represented as discrete, we can derive a sequence of codes $\{q_t | t = 1, 2, ..., l; 0 \leq q_t < V\}$. Then, the autoregressive transformer produces a distribution over codebook by a V-way classifier to fit a conditional distribution of $p(q_t | q_{<t}, c)$, where c can be any conditions to control the generated audio, like textual prompts in this paper. During inference, commencing from a user given c, the transformer $\mathcal{F}$ predicts next time-step discrete token autoregressively. After obtained all of tokens, they are concatenated along the temporal dimension, then utilized to lookup the codebook to derive the codes, which are then feed into de-tokenizer to derive generated waveform $\tilde{x}$.

C.3 Algorithms of Siren

Algorithm 1 First Stage Training Pipeline of `Siren`

Require: Ground-Truth RVQ tokens $\mathbf{q} \in [V]^{r \times l}$;
Require: Hyperparameters: the number of RVQ layer r, the number of transformer model $K = r/2$;
Require: Each transformer $\mathcal{H}_k$ is composed of backbone model $\mathcal{F}_k$, residual layer decoding model $\mathcal{R}_k$, and two classifiers $(\mathcal{C}_{2k}, \mathcal{C}_{2k+1})$.

for $k = 1, \cdots, K$ **do**
  for train loops **do**
    Predict next-time feature: $(h_{1 \cdots l})_k = \mathcal{F}(\sum_{j=1}^{r} \texttt{lookup}(Z^j, \mathbf{q}^j_{sos, 1 \cdots l-1}), c)$;
    Predict next-r-codes: $\hat{\mathbf{q}}^{2k}_{1 \cdots l} = \mathcal{C}_{2k}\mathcal{R}_k((h_{1 \cdots l})_k, \mathbf{q}^{<2k}_{1, \cdots, l})$;
    Predict next-r-codes: $\hat{\mathbf{q}}^{2k+1}_{1 \cdots l} = \mathcal{C}_{2k+1}\mathcal{R}_k((h_{1 \cdots l})_k, \mathbf{q}^{<2k+1}_{1, \cdots, l})$;
    Compute loss: $\mathcal{L} = \mathcal{L}_{ce}(\hat{\mathbf{q}}^{2k}_{1 \cdots l}, \mathbf{q}^{2k}_{1 \cdots l}) + \mathcal{L}_{ce}(\hat{\mathbf{q}}^{2k+1}_{1 \cdots l}, \mathbf{q}^{2k+1}_{1 \cdots l})$;
  end for
end for

Ensure: $\{\mathcal{H}_k | \mathcal{H}_k = \mathcal{F}_k \times \mathcal{R}_k \times \mathcal{C}_{2k} \times \mathcal{C}_{2k+1}\}_{k=1}^{r/2}$

Algorithm 2 Sampling Pipeline of `Siren`

Require: Trained $r/2$ transformers in the first/second stage;
Require: Detokenizer $\mathcal{D}$;
for $t = 1, \cdots, l$ **do**
  for $k = 1, \cdots, r/2$ **do**
    Predict next-time feature: $(h_t)_k = \mathcal{F}(\sum_{j=1}^{r} \texttt{lookup}(Z^j, \mathbf{q}^j_{sos,<t}), c)$;
    Predict next-first-code: $\hat{\mathbf{q}}_t^{2k} = \mathcal{C}_{2k}(\mathcal{R}_k((h_t)_k, \hat{\mathbf{q}}_t^{<2k}))$;
    Predict next-second-code: $\hat{\mathbf{q}}_t^{2k+1} = \mathcal{C}_{2k+1}(\mathcal{R}_k((h_t)_k, \hat{\mathbf{q}}_t^{<2k+1}))$;
    Update: $\hat{\mathbf{q}} \leftarrow \hat{\mathbf{q}}_t^{2k}$, $\hat{\mathbf{q}} \leftarrow \hat{\mathbf{q}}_t^{2k+1}$.
  end for
end for
$\hat{x} = \mathcal{D}(\hat{\mathbf{q}})$
Ensure: $\hat{x}$

Algorithm 3 Second Stage Training Pipeline of `Siren`

Require: Trained $r/2$ transformers in the first stage;
Require: Detokenizer $\mathcal{D}$ and ImageBind models $\phi_{audio}, \phi_{text}$;
Require: Hyperparameters: Advantage threshold γ, clip thresholds ϵ_d, ϵ_u;
for train loops **do**
  Sample a batch of prompts $\{c_1, ..., c_B\}$;
  Update the old policy model $(\pi_{\theta_1})_{old} \leftarrow \pi_{\theta_1}$;
  Sample G roll-out $\{(\hat{\mathbf{q}}_{1,...,l})_i\}_{i=1}^{G} \sim (\pi_{\theta_1})_{old}(\cdot|c_b)$ for each prompt $c_b, b \leq B$;
  Compute rewards $\{R_i\}_{i=1}^{G}$ for each roll-out sequence by Eq. 8;
  For each $(\hat{\mathbf{q}}_{1,...,l})_i$ in the buffer, compute advantage A_i via Eq. 9, and filter long G using γ;
  for iteration=1,...,μ **do**
    Update the policy model π_{θ_1} by maximizing the objective in Eq. 10.
  end for
end for
Ensure: θ_1

D Additional Experimental Results

D.1 Property 1& Influence 1

We extract each layer' s quantized features and average pool them along temporal dimension. Hence, we derive a feature $\hat{f}^j = \texttt{pool}(\hat{f}^j_{1,...,l})$. Then, we compute the cosine similarity between quantized features from any two different layers. As shown in Figure 8, we illustrate the distribution of cosine similarity between any-two layers. It is evident that the overall cosines are distributed around 0, which represents the near orthogonality. It is noted that the cosines between the adjacent layers from begining or ending are distributed relatively larger than 0. We deem that it is because of local homogeneity among those adjacent layer when represented knowledge is extreme, *extremely semantic rich or poor*. As for gradient distributions in influence 1, we gather the gradients tensor, usually in two-dimension, and average along output dimension to derive the gradient influence to the input neurons. Then, we compute any two of layers gradient vector angles and derive a 90 degree-around distribution, which also supports our claim that orthogonality incurs diverse optimization direction aggregated to the input neurons.

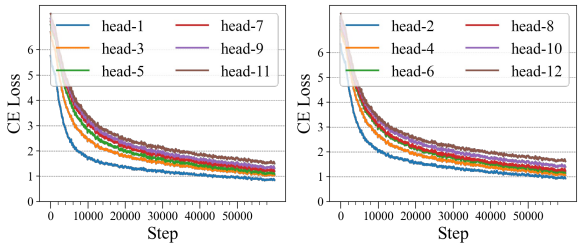

Figure 6: Training convergence curves of learning different RVQ layers.

D.2 Property 2& Influence 2

In this section, we firstly complement the details of training the classifiers. Given an audio sample, that has been tokenized into $\mathbf{q} \in [V]^{r \times l}$, we extract each row vector $\mathbf{q}^j \in [V]^l$ as input tokens, and use it to loop up corresponding codebooks to derive classifiers' input features $\hat{f}^j \in \mathbb{R}^{l \times C}$, where C is the number of feature channel. Then, we further introduce two MLP layers to conduct 15-way classification on AVSync-15 dataset. Then, we report the top-1 accuracy over test set. As shown by Figure 9, the classification accuracy is descending as RVQ layer goes deeper. According the related research in representation learning (He et al., 2020), one can infer the semantic richness within a representation from its corresponding classification accuracy performance. For example, we can distinguish objects according to its semantic information. Thus, we can derive a conclusion in Property 2. Then, we further illustrate the training curves of

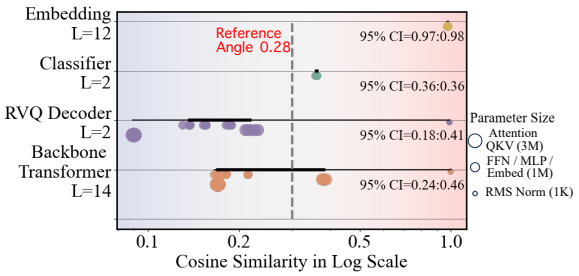

Figure 7: Cosine similarity between the same layer' s *task vector* from different learned transformer. Each scatter denotes an average cosine value over all transformer' s two-combinations. The boxplots exhibit the quartiles and extremes over each module' s *task vector*. As a reference, we put the average cosine values between the models in 10^{th} epoch with the last epoch in red fonts in this figure.

AudioGen on our 100K version dataset in Figure 6. The convergence processes of different heads that is responsible for RVQ layers show that descending semantics incurs imbalanced learning difficulties.

D.3 Analytic Results

The Necessarty of Introducing Collaboration. We define the task vector as the difference between the weights of the best six models and the same initialization weights, reflecting the impact of different tasks on model weights. As shown in Figure 7, we categorize the model into four structures: Embedding, Classifier, RVQ Decoder, and Backbone Transformer, as indicated on the vertical axis. The RVQ Decoder and Backbone Transformer have three layers with distinct parameters, represented by circles of varying diameters. The horizontal axis shows the weight similarity among the same layers within each structure, totaling 15 combinations.

It can be observed that the RVQ Decoder establishes causal relationships between codes, striving for orthogonality between each code, thus exhibiting lower similarity. The Backbone Transformer and Classifier follow, influenced by the task and sharing some common characteristics. Other structures and layers have little relationship with the task, displaying higher similarity among them. This indicates that the core structure and the codes each model handles are closely related, suggesting the necessity for separation.

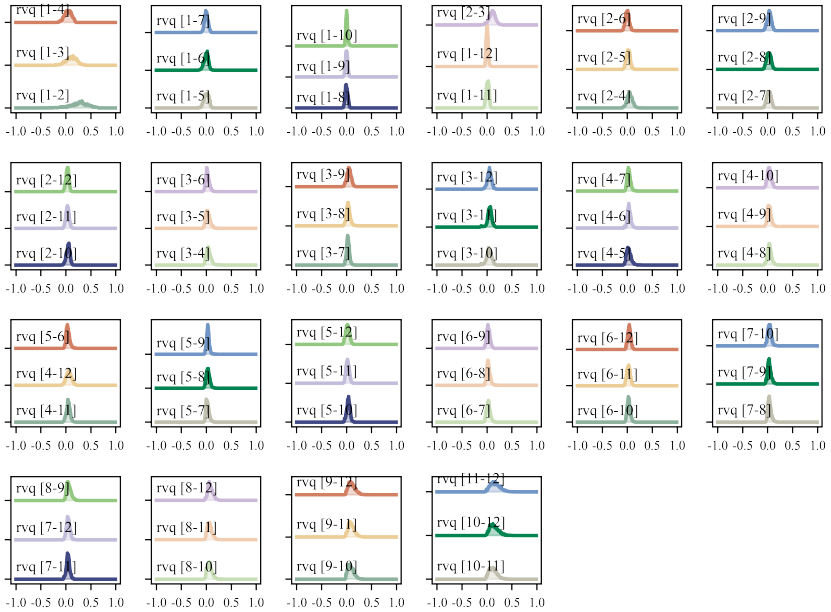

Figure 8: Cosine distributions between quantized features from any two RVQ layers.

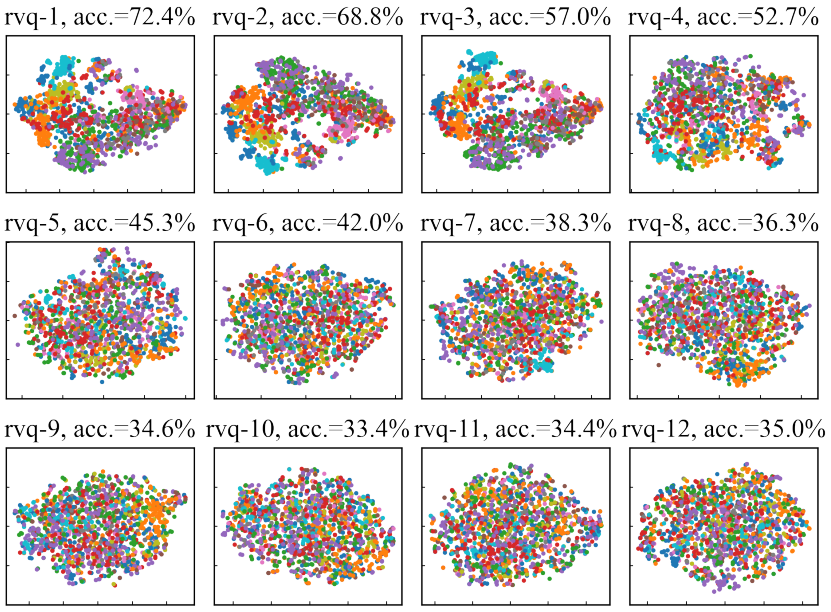

Figure 9: T-SNE processed quantized features' s distribution and corresponding top-1 classification accuracy.