Early Speech-Like Audio Emergence in a Small OLMo-Hybrid Speech Codec Language Model

Completed 12k-Step A100 Report on LJ Speech

Micheal Ohagwu
obioh@icloud.com

March 2026

Abstract

We study whether a small OLMo-Hybrid-style recurrent-attention language model can learn enough speech codec structure to generate clearly speech-like audio, without text conditioning, on a clean single-speaker corpus. The model operates on 8-codebook EnCodec 24 kHz tokens extracted from LJ Speech and uses an 8-layer decoder with a 3:1 recurrent-to-attention schedule, yielding 6 Gated DeltaNet blocks and 2 attention blocks for a total of 21.9M parameters. We first observed speech-like emergence in an interrupted A100 pilot through step 1800, then resumed the same stable configuration and completed training to step 12000 on an NVIDIA A100-SXM4-80GB. Across the full completed run, EMA validation loss improved monotonically from 6.7878 at step 200 to 3.8207 at step 12000, with perplexity improving from 886.99 to 45.63. Direct listening to locally decoded sample bundles confirmed a clear speaking timbre and speech-like local structure; failure modes were dominated by babble, weak articulation, and missing semantics rather than static, drone artifacts, or codec collapse. The stable run did not use the intended fused Flash Linear Attention Gated DeltaNet kernel because of upstream Triton kernel failures, so the reported systems performance should be interpreted as conservative relative to the architecture's intended fast path. We present this as a technical report rather than a benchmark claim: the main result is that an OLMo-Hybrid / Gated-DeltaNet-style backbone is viable for speech codec language modeling at small scale, and remains stable enough to improve throughout a full 12k-step A100 run.

1 Introduction

Recent open work has renewed interest in hybrid backbones that mix attention with modern recurrent or state-space layers, rather than treating transformers and recurrence as mutually exclusive design choices [6, 8]. In parallel, work on neural audio codecs and token language models has shown that discrete audio tokens are a practical interface for speech and audio generation [1, 3, 7, 9]. A natural next question is whether these ideas combine well: can a small hybrid recurrent-attention backbone model speech codec tokens effectively enough to generate clearly speech-like samples?

This report documents a deliberately narrow pilot designed to answer that question. We did not attempt to build a full text-to-speech system, a streaming agent, or a production voice model. Instead, we focused on a threshold question:

> *Can a compact OLMo-Hybrid-style decoder trained unconditionally on speech codec tokens cross the boundary from static or degenerate decoder output into clearly speech-like audio?*

That threshold matters because it separates architecture and pipeline viability from final product quality. If the model cannot produce anything beyond hiss, drone artifacts, or obvious tokenization failure, then text conditioning, speaker conditioning, and larger-scale training are premature. If it can produce speech-like babble at small scale, then the architecture is at least plausible for more serious follow-up work.

An earlier draft of this report documented only the interrupted pilot through step 1800. The present version incorporates the completed continuation of that same A100 run out to step 12000. The completed run matters because it answers two additional questions that the interrupted pilot left open:

1. whether the clean A100 validation trend observed through step 1800 would persist much deeper into training; and
2. whether the architecture would plateau or overfit quickly on a narrow single-speaker corpus once the learning rate decayed into its low-LR tail.

Our main result is still modest but now much better supported: on LJ Speech [4], a 21.9M-parameter OLMo-Hybrid-style speech codec language model trained on EnCodec 24 kHz tokens [3] produced clearly speech-like audio by step 1800 on an A100, then continued to improve monotonically all the way to step 12000, reaching EMA validation loss 3.8207 and perplexity 45.63. To our knowledge, public speech token language modeling work has focused primarily on transformer-heavy or pure state-space backbones [1, 5, 7, 9]; we are not aware of a prior public technical report centered on an OLMo-Hybrid / Gated-DeltaNet-style backbone for unconditional speech codec language modeling at this scale.

We therefore position this document as a technical report rather than a benchmark paper. Its contributions are:

1. a concrete OLMo-Hybrid-style adaptation for speech codec language modeling, described at the level of block schedule, head geometry, token interface, and loss;
2. a completed 12k-step A100 run showing monotonic EMA validation improvement from 6.7878 to 3.8207;
3. qualitative evidence that the model learned speech-like local structure well before full convergence and retained that behavior through the completed run; and
4. an honest systems account of what did and did not work, including the stable no-FLA A100 path, resumed training after pod failures, and the operational consequences of volatile rented infrastructure.

Representative audio samples, checkpoints, and evaluation history are released alongside this report on the accompanying project page and code repository.[1]

2 Related Work

Hybrid recurrent-attention language models. OLMo Hybrid argues that models mixing attention and linear RNN layers are not merely inference-efficient alternatives to transformers, but a broader family that can remain expressive while scaling effectively in pretraining [6]. Their released

[1] https://obiohagwu.github.io/obiohagwu/2026/03/10/A-21.9M-hybrid-recurrent-attention-to-learn-speech-on-1.8k-steps.html and https://github.com/Obiohagwu/olmo-hybrid-speech

7B model uses a 3:1 schedule of Gated DeltaNet and attention layers. The underlying recurrent operator, Gated DeltaNet, extends DeltaNet-style recurrence with gated parameterization, short causal convolutions, and a training-oriented formulation intended to pair with fused Flash Linear Attention kernels [8].

Speech token language modeling. AudioLM established that language models over discrete audio tokens can generate coherent speech and other audio without relying on text supervision [1]. VALL-E reframed zero-shot TTS as neural codec language modeling and showed that codec-token autoregression can support high-quality conditional speech synthesis at scale [7]. SpeechTokenizer argued that tokenizers designed specifically for speech language modeling can outperform more generic alternatives and paired them with a unified speech language model [9]. On the state-space side, DuplexMamba explored Mamba-based speech interaction, but in a duplex streaming speech-to-text conversational setting rather than an unconditional speech codec language model [5].

Token interleaving for multi-codebook generation. We train with a delay-pattern objective closely aligned with the multi-codebook interleaving strategy popularized by MusicGen [2]: codebook k is shifted by k positions so that a single autoregressive step predicts one token per codebook on a staircase-shaped frontier.

3 Model

3.1 Overview

The model is a decoder-only language model over residual vector quantization (RVQ) codes. Audio is tokenized into $K = 8$ EnCodec codebooks; the model consumes delay-patterned tokens and predicts the next delayed token autoregressively. Table 1 summarizes the configuration of the completed stable A100 run.

3.2 Token Interface and Delay Pattern

Let raw codec tokens be $z \in \{0, \ldots, V-1\}^{K \times T}$ with $K = 8$ codebooks and vocabulary size $V = 1027$, including special tokens. We apply a delay pattern Δ,

$$\tilde{z}_{k,t} = \begin{cases} z_{k,t-k}, & \text{if } 0 \le t-k < T, \\ \text{PAD}, & \text{otherwise,} \end{cases}$$

so the model predicts a staircase of codebook targets rather than all raw codebooks aligned at the same step. The copied run configuration retained a stale fallback `frame_rate` field unrelated to the actual run, so for this report we treat the dataset metadata and sample manifests as authoritative. Those artifacts imply an effective frame rate of 75 Hz for the EnCodec token stream: a 6-second continuation contains 450 raw frames, and an 8-second training chunk contains 600 raw frames. Under the 8-codebook delay pattern, an 8-second chunk becomes 607 delayed steps. The 1024-step context therefore covers the full training chunk without truncation.

3.3 Backbone

The backbone follows the public OLMo Hybrid pattern [6]: three recurrent blocks followed by one attention block, repeated across the network, with the final layer forced to attention. In this 8-layer

Table 1: Architecture and training configuration for the stable completed A100 run.

Item	Value
Total parameters	21,904,648
Embedding parameters	3,154,944
Backbone parameters	15,594,376
Output-head parameters	3,155,328
Layers	8
Width	$d_{\text{model}} = 384$
FFN width	$d_{\text{ff}} = 1024$
Attention heads	6
KV heads	2
Attention schedule	every 4th block; final block forced to attention
Block mix	6 Gated DeltaNet + 2 attention
Dropout	0.1
Max sequence length	1024 delayed steps
Codebooks / vocab	8 / 1027
Codec	EnCodec 24 kHz, 6.0 kbps
Chunk length	8.0 s
Batch size	24
Optimizer	fused AdamW, $\beta = (0.9, 0.95)$, weight decay 0.01
Schedule	warmup 500, cosine decay from 3×10^{-4} to 10^{-5}
Precision	bfloat16

instance, the schedule is

$$[\text{GDN}, \text{GDN}, \text{GDN}, \text{Attn}, \text{GDN}, \text{GDN}, \text{GDN}, \text{Attn}].$$

Each block is pre-norm with RMSNorm, mixer, residual add, RMSNorm, SwiGLU MLP, and a second residual add. RVQ embeddings are summed across codebooks at each delayed step to form a single 384-dimensional token representation; there is no learned absolute positional embedding.

3.4 Attention Blocks

Attention blocks use grouped-query attention with 6 query heads and 2 KV heads, so the attention head dimension is $384/6 = 64$. Queries and keys are normalized with per-head RMSNorm, rotary position embeddings use $\theta = 500{,}000$, and PyTorch scaled dot-product attention is used on the fast CUDA path.

3.5 Gated DeltaNet Blocks

The recurrent blocks implement a paper-aligned Gated DeltaNet-style mixer. Relative to the attention geometry, the recurrent blocks shrink the q/k head width and expand the value width:

$$d_{qk}^{\text{GDN}} = 0.75 \times 64 = 48, \qquad d_{v}^{\text{GDN}} = 2 \times 48 = 96.$$

For each timestep, the block forms q, k, v, and gating signals a and b, together with learnable $A_{\log}$ and dt_{bias} parameters and short depthwise causal convolutions over the projected sequences. In the plain recurrent scan actually used for stable training, the state update is

$$r_t = S_{t-1} k_t, \tag{1}$$

$$S_t = \alpha_t \odot \left(S_{t-1} - \beta_t \odot (r_t k_t^\top) \right) + \beta_t \odot (v_t k_t^\top), \tag{2}$$

$$y_t = S_t q_t, \tag{3}$$

with q_t and k_t L2-normalized and with a gated RMSNorm output projection. This mirrors the intended Gated DeltaNet structure from the released OLMo-core implementation, but in the stable A100 run it executed through the plain PyTorch fallback rather than the fused FLA kernel.

3.6 Objective

The training objective is next-token cross-entropy on delayed tokens:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{k,t} \text{CE}\left(p_\theta(\tilde{z}_{k,t+1} \mid \tilde{z}_{\leq t}), \tilde{z}_{k,t+1} \right),$$

ignoring pad positions and using label smoothing of 0.1. This is an unconditional codec language model: there is no text prompt, no speaker embedding, and no semantic token stream.

4 Data and Training Setup

4.1 Dataset

We used LJ Speech [4], a clean single-speaker English corpus frequently used in TTS research. The appeal of LJ Speech for this pilot was not scale but cleanliness: if the goal is to determine whether the backbone can learn local speech structure, a narrow acoustic domain is preferable to a noisy heterogeneous corpus.

We tokenized the corpus into non-overlapping 8-second chunks and split at the utterance level to avoid train/validation leakage across chunks from the same source waveform. The resulting tokenized dataset contained 12,624 training files and 666 validation files.

4.2 Codec and Metadata

The authoritative dataset metadata came from the generated `codec_meta.json`: EnCodec 24 kHz, 8 codebooks, codebook size 1024, 6.0 kbps bandwidth, and 8.0-second chunking. One copied run configuration retained stale fallback codec defaults unrelated to the actual run, so for this report we reconstruct the codec description from the dataset metadata and saved sample manifests rather than relying on the copied run configuration alone.

4.3 Stable A100 Run

The stable run used:

- one NVIDIA A100-SXM4-80GB GPU,
- bfloat16 mixed precision,

Table 2: A100 operating-point search for the stable no-FLA training path.

Configuration	Effective batch	Outcome	Role in final result
8×4	32	stable but underutilized	initial safe A100 launch
16×2	32	stable, better utilization	intermediate tuning point
24×1	24	stable, best practical tradeoff	final reported run
32×1	32	out-of-memory / unstable	rejected

- true batch size 24,
- 8 dataloader workers with prefetch factor 4, pinned memory, and non-blocking transfers,
- fused AdamW,
- CUDA scaled dot-product attention flash path enabled,
- `torch.compile` disabled for the stable run, and
- the fused recurrent FLA path disabled because of backward-kernel instability on the available Triton stack.

An earlier batch-32 attempt ran out of memory. Smaller batch/accumulation combinations were also tried, but batch 24 with no accumulation was the best stable operating point found under the available budget.

4.4 Run Chronology

The completed result reported here spans two phases of the same A100 configuration. The initial pilot reached step 1800, where the best preserved checkpoint had EMA validation loss 4.2847 and perplexity 72.58. Because of pod volatility, that pilot was resumed from the preserved step-1800 checkpoint on a fresh A100 pod and continued to the planned step-12000 budget with the same stable no-FLA configuration.

This chronology matters for interpretation. The final result is not a different training recipe; it is the completed continuation of the same stable A100 run lineage. That makes the evaluation history from step 200 through step 12000 directly interpretable as one coherent training trajectory, even though operationally it crossed multiple rented pod sessions.

4.5 Systems Caveat

The intended fast path for the recurrent mixer was the fused Flash Linear Attention Gated DeltaNet kernel. In practice, the available pod/software stack failed in the fused backward kernel with Triton code-generation errors. As a result, the stable run reported here combines strong CUDA kernels for attention and optimization with an unfused recurrent fallback. This matters for systems interpretation: the run demonstrates architectural viability, but not the architecture's best possible throughput on its intended kernel stack.

Table 3: Selected evaluation checkpoints from the completed A100 run.

Step	Train loss	LR	Grad norm	EMA val loss	PPL
200	5.0049	1.19×10^{-4}	0.28	6.7878	886.99
1000	4.1367	2.99×10^{-4}	0.21	4.8510	127.87
1800	4.2224	2.91×10^{-4}	0.18	4.2847	72.58
3200	4.2821	2.62×10^{-4}	0.20	4.0500	57.40
5200	3.7050	1.96×10^{-4}	0.19	3.9194	50.37
7600	3.7936	1.03×10^{-4}	0.21	3.8514	47.06
10000	3.8445	3.11×10^{-5}	0.22	3.8267	45.91
12000	3.7043	1.00×10^{-5}	0.22	3.8207	45.63

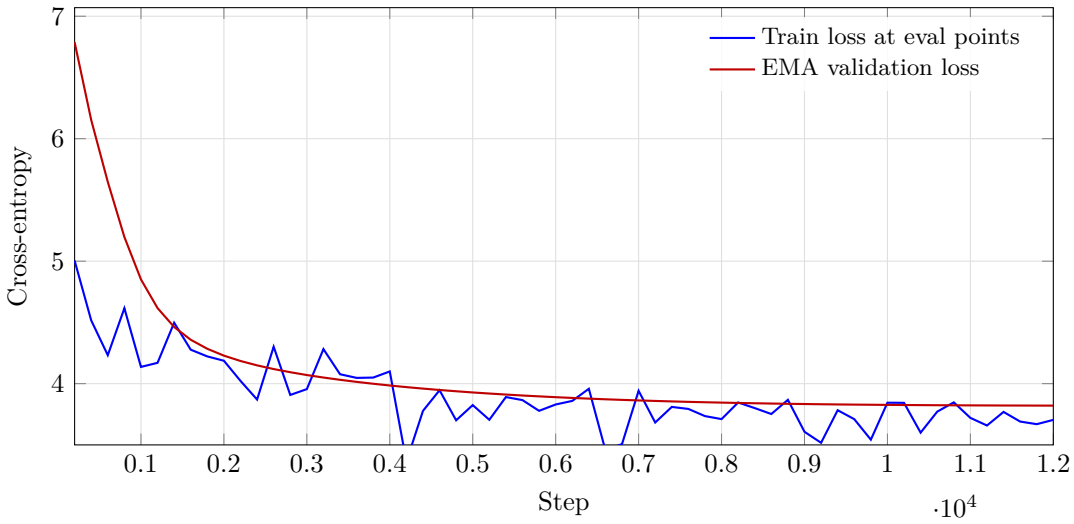

Figure 1: Training and validation loss over the completed A100 run. Validation improved monotonically from step 200 through step 12000, with no reversal observed within the allotted budget.

5 Results

5.1 Validation Trajectory

The completed run answered the main unresolved question from the earlier pilot: the clean validation trend did not reverse after step 1800. Instead, it continued to improve at every recorded evaluation through the end of training. The absolute gain from the preserved pilot checkpoint to the final best checkpoint was:

$$4.2847 \rightarrow 3.8207 \quad (\Delta = -0.4640),$$

with perplexity improving from 72.58 to 45.63. In relative terms, perplexity fell by approximately 37% across the continuation from step 1800 to step 12000.

This is a stronger result than merely saying the model "still trained" after the pilot. The completed run shows that the pilot was not sitting on the edge of a transient local optimum. The same configuration remained stable and productive across the full planned budget.

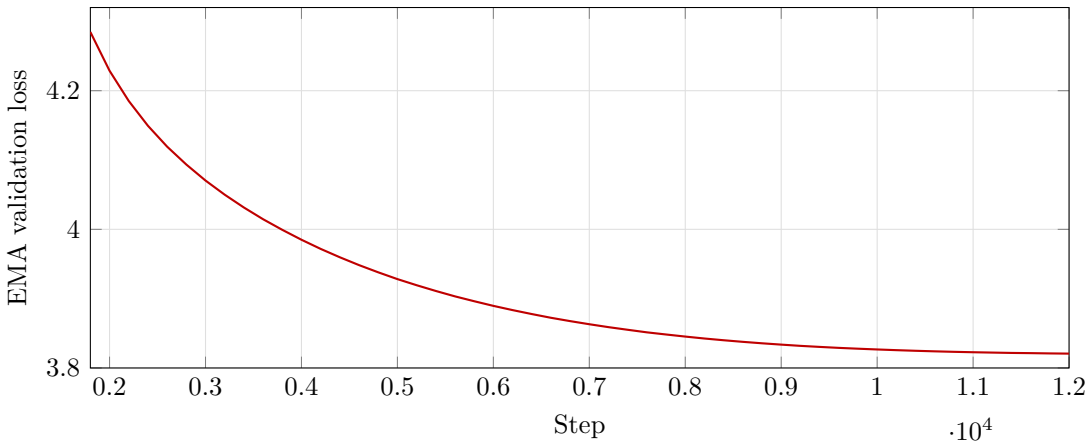

Figure 2: Late-stage validation trajectory from the step-1800 pilot checkpoint through the completed step-12000 run. Improvement remained monotonic but entered a clear diminishing-returns regime after roughly step 8000.

5.2 Late-Stage Behavior

The completed run also clarifies the character of the late training regime. By step 7800, the model had already entered a gradual diminishing-returns phase: every evaluation still improved, but only by small increments. For example:

- step 7800: val loss 3.8483,
- step 9000: val loss 3.8337,
- step 10400: val loss 3.8248,
- step 12000: val loss 3.8207.

This behavior is exactly what one would expect from a stable run that has entered the low-learning-rate tail of cosine decay rather than a run that has begun to overfit or oscillate destructively.

5.3 Checkpoint Chronology

The final checkpoint cadence was every 200 steps. The locally preserved A100 checkpoint chronology now covers the full run budget:

$$200 \rightarrow 400 \rightarrow \cdots \rightarrow 11800 \rightarrow 12000.$$

In addition to the earlier pilot checkpoints, we preserved late-run checkpoints at steps 8000, 9000, 10000, 11000, and 12000 locally for qualitative progression analysis. The final best checkpoint coincides with the final training step: step 12000.

5.4 Qualitative Sampling Protocol

Qualitative evaluation was performed by direct listening to locally decoded 6-second continuations under fixed prompt and sampling settings. All qualitative judgments in this report are therefore author judgments rather than blinded external ratings.

The most informative final probe used the step-12000 best checkpoint with a corrected delay-space sampler and a conservative speech-oriented decode configuration:

- one raw seed frame from a fixed training chunk as prompt,
- duration 6.0 seconds,
- `best_of = 3`,
- top-$p = 0.9$,
- codebook-aware temperature schedule $0.72 \rightarrow 0.55$,
- codebook-aware top-k schedule $48 \rightarrow 24$, and
- repetition penalty 1.08 over a 48-token window.

The final-best local five-sample bundle used those exact settings and decoded the best checkpoint selected by EMA. Earlier pilot sampling used the same corrected delay-space logic but without the final best-of sweep.

5.5 Qualitative Outcome

The central qualitative result remains the same as in the interrupted pilot, but is now better supported by a much longer training run: the model lives in a speech-like output regime rather than a codec-collapse regime. Outputs are not dominated by static, hiss, or drone artifacts. Instead, they exhibit:

- a clearly human speaking timbre,
- local phonetic and prosodic structure,
- partial articulation and cadence, and
- failure modes dominated by babble, weak semantics, and variable pacing.

That distinction matters. The language model itself learned enough token dynamics to stay within a recognizably speech-like manifold. What is missing is not the existence of voice, but higher-order control: semantics, stable articulation, and robust sentence-level structure.

5.6 Systems Performance

Throughput on the final A100 operating point was typically in the 17.8k–18.3k token-equivalents/s range on ordinary training windows. With batch size 24, 8 codebooks, and 1024 delayed steps, that corresponds to roughly 196,608 token-equivalents per optimizer step, or approximately 10.7 seconds per optimizer step excluding evaluation/save overhead. GPU snapshots during active training commonly showed about 59 GB of 80 GB allocated, with instantaneous utilization around 69% and power draw around 170 W. Lower instantaneous utilization samples were observed around evaluation/save boundaries and should not be read as the representative steady-state regime.

The key systems point is unchanged: the bottleneck was not attention, dataloading, or basic CUDA support. It was the unfused recurrent Gated DeltaNet path. This explains why the run was materially faster than the M4 path while still leaving headroom on an A100 that would normally look almost idle on a tiny dense transformer.

6 Discussion

The completed run strengthens the central claim of the earlier pilot. What was previously a promising interrupted result is now a completed, coherent trajectory with three strong properties:

1. **architectural viability:** the hybrid backbone does learn a stable speech-token distribution;
2. **optimization stability:** the model remained numerically and statistically well behaved through the entire 12k-step budget; and
3. **qualitative threshold crossing:** the model entered and remained in a clearly speech-like output regime.

The most important update relative to the interrupted report is that no validation reversal was observed within the full run budget. On the narrow single-speaker LJ Speech corpus, one plausible concern was that the model might improve quickly and then plateau or overfit once the learning rate decayed. That did not happen. Instead, the run exhibited a long smooth tail of slow but monotonic improvement.

At the same time, this result should still be interpreted narrowly. It does not show that this backbone is superior to a matched transformer. It does not show that the model is close to production-quality TTS. It does not show strong semantic coherence. What it does show is that the architecture deserves to be taken seriously as a speech codec language-modeling backbone.

7 Limitations and Threats to Validity

This report still has several important limitations.

No matched baseline. We did not run a same-parameter transformer baseline under the same data, tokenizer, and training budget. This prevents any claim of architectural superiority.

No human study. Qualitative evaluation was direct listening by the experimenter, not a blinded human evaluation with MOS, CMOS, or transcription-based intelligibility metrics.

Single-speaker data. LJ Speech is intentionally narrow. That is useful for isolating whether the architecture can learn speech structure, but it limits claims about general multi-speaker speech modeling.

No text conditioning. This was an unconditional codec language model. The report does not address pronunciation control, text alignment, or TTS quality.

Kernel mismatch. The stable run did not use the intended fused FLA recurrent path. The result therefore validates the model family more than it validates the final optimized training stack.

Infrastructure volatility. The training environment remained operationally fragile. Pod restarts changed SSH endpoints and made it necessary to secure artifacts incrementally off-pod. The final result was completed successfully, but this required active artifact preservation rather than trusting pod-local storage.

8 Conclusion

We presented a completed technical report on an OLMo-Hybrid-style speech codec language model trained unconditionally on LJ Speech tokenized with EnCodec 24 kHz. A 21.9M-parameter model with 6 Gated DeltaNet blocks and 2 attention blocks reached EMA validation loss 3.8207 and perplexity 45.63 by step 12000 on an A100, and direct listening confirmed clearly speech-like outputs rather than static or decoder collapse. The model remained far from semantic or production-quality speech synthesis, but it crossed the threshold that matters for a pilot and then continued improving across the full training budget.

The next steps are now straightforward and better motivated than they were after the interrupted pilot:

1. a matched transformer baseline,
2. text conditioning,
3. more formal qualitative and quantitative evaluation,
4. a progression study across late checkpoints, and
5. eventually a fully fused recurrent kernel stack.

If those follow-ups hold, this report will have served its purpose not as a finished speech system, but as a clean demonstration that this architecture family belongs in the speech-token modeling conversation.

Artifact note. The locally preserved artifact set now includes the full completed training log for the resumed A100 continuation, the final run configuration, the final best checkpoint at step 12000, the final numbered checkpoint, intermediate late-run checkpoints at 8000/9000/10000/11000, and local qualitative sample bundles from both the earlier pilot and the final best checkpoint.

References

[1] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matthew Sharifi, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. AudioLM: A language modeling approach to audio generation, 2022. https://arxiv.org/abs/2209.03143.

[2] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation, 2023. https://arxiv.org/abs/2306.05284.

[3] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression, 2022. https://arxiv.org/abs/2210.13438.

[4] Keith Ito and Linda Johnson. The LJ Speech dataset, 2017. https://arxiv.org/abs/1707.03982.

[5] Xiangyu Lu, Xu Wang, Haoyu Wang, Hongyun Zhou, Haiyan Zhao, Conghui Zhu, Tiejun Zhao, and Muyun Yang. DuplexMamba: Enhancing real-time speech conversations with duplex and streaming capabilities, 2025. https://arxiv.org/abs/2502.11123.

[6] William Merrill, Yanhong Li, Tyler Romero, Anej Svete, Caia Costello, Pradeep Dasigi, Dirk Groeneveld, David Heineman, Bailey Kuehl, Nathan Lambert, Chuan Li, Kyle Lo, Saumya Malik, DJ Matusz, Benjamin Minixhofer, Jacob Morrison, Luca Soldaini, Finbarr Timbers, Pete Walsh, Noah A. Smith, Hannaneh Hajishirzi, and Ashish Sabharwal. OLMo Hybrid: From theory to practice, 2026. Technical report.

[7] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. Neural codec language models are zero-shot text to speech synthesizers, 2023. https://arxiv.org/abs/2301.02111.

[8] Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule, 2024. https://arxiv.org/abs/2412.06464.

[9] Xin Zhang, Dong Zhang, Shimin Li, Yaqian Zhou, and Xipeng Qiu. SpeechTokenizer: Unified speech tokenizer for speech large language models, 2023. https://arxiv.org/abs/2308.16692.