GSLM2: Multi-Stream Speech Representation Learning and Generative Spoken Language Modeling

Anonymous authors
Paper under double-blind review

Abstract

Generative spoken language models (SLMs) offer a route to learning linguistic structure purely from speech, yet they lag far behind text-based models in both capability and data scaling efficiency. We identify two bottlenecks underlying this gap: the limited linguistic quality of existing speech units and the reliance on a single stream of discrete units to represent speech. Multiple codebooks are widely used to encode acoustic features but rarely to represent speech semantics. We present GSLM2, which learns and models complementary properties of speech at multiple levels of abstraction. Its encoder, **SpidR 2.0**, uses self-supervised learning with differentiable prediction heads to induce high-quality discrete units at multiple levels. A **multi-stream SLM** then jointly models these units using a shared temporal backbone, stream-specific predictors, and an autoregressive head that captures cross-stream dependencies. Across training data sizes and model scales, up to 6M hours and 6.5B parameters, GSLM2 consistently improves linguistic competence while scaling more efficiently with data. As a result, it substantially narrows the gap between spoken and text-based LMs; across read and conversational speech domains, we shrink the gap by about two orders of magnitude on sWuggy, sBLIMP, and tSC, while also outperforming competitive speech-text systems on all three tasks.

1 Introduction

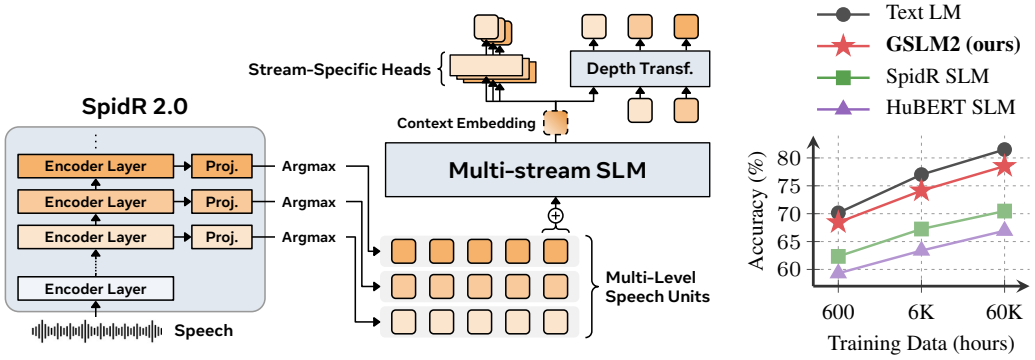


Figure 1: **Overview of GSLM2.** *Left*: SpidR 2.0 extracts discrete speech units from multiple encoder layers, which are jointly modeled by the multi-stream SLM. *Right*: Scaling performance of SLMs and a text LM trained on varying amounts of Libri-Light data, measured by the average accuracy across the sWuggy, sBLIMP, and tSC benchmarks.

Generative spoken language modeling (GSLM) (Lakhotia et al., 2021) seeks to learn the structure of language directly from speech, typically by modeling sequences of discrete speech units. This textless approach aligns with embodied and cognitive developmental perspectives of intelligence, which view language as building on representations of the world learned through perception and action (LeCun, 2022; Dupoux et al., 2026). A central challenge is to discover this structure in continuous, high-dimensional signals without symbolic supervision—a problem human learners demonstrably solve. Models that learn from speech alone are not restricted to languages with a written form and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

they acquire speech-native representations that retain both linguistic content and speech-specific information such as phonetic variation. These representations are particularly valuable for speech-first applications such as spoken dialogue modeling (Nguyen et al., 2023). However, a major obstacle to textless modeling remains: spoken language models (SLMs) can require up to four orders of magnitude more training data than text LMs to achieve comparable linguistic performance (Cuervo & Marxer, 2024; Poli et al., 2025b; Maimon et al., 2025b), as illustrated by HuBERT- and SpidR-based SLMs in Figure 1. Bridging this data-efficiency gap is the goal of this paper.

The gap may partly stem from the high sampling rate and low information density of speech representations: the same content requires many more speech units than text tokens (Cuervo & Marxer, 2024). Recent work has explored representations with lower sampling rates, e.g., word-level (Algayres et al., 2023), syllable-like (Cho et al., 2025; Baade et al., 2025; Visser et al., 2026), and latent patch-based units (Lu et al., 2026). By shortening sequences, these approaches expand the effective context for next-unit prediction and improve linguistic abilities, yet they still lag behind text LMs. Despite their differing granularities, they retain the text-inspired assumption that speech should be modeled as a single stream of discrete units, seeking more efficient substitutes for text tokens. Compression alone, however, does not resolve which distinctions these units should preserve. Coarser units can improve efficiency and performance on sentence-level semantic tasks, but may lose finer phonetic and lexical distinctions (Kando et al., 2025; Visser & Kamper, 2025). These trade-offs suggest that the most useful level of detail may vary with the task and context, motivating flexible access to different levels of abstraction. Choosing a single unit type fixes the balance between compactness and detail, potentially restricting available information as these demands change.

We argue that this reflects a broader representational bottleneck. SLMs typically rely on speech representations or discrete units derived from a single layer of a self-supervised speech encoder, despite evidence that different layers capture complementary phonetic and linguistic information (Pasad et al., 2021; Shen et al., 2023; Pasad et al., 2024). Consequently, a single representation is expected both to preserve fine-grained phonetic distinctions and to expose the high-level semantic regularities required for efficient language modeling. Satisfying these competing objectives within one sequence may unnecessarily burden the model and limit its ability to learn linguistic structure.

To this end, we introduce **GSLM2**, a framework for representing and modeling speech at multiple levels of abstraction. Its encoder, **SpidR 2.0**, uses self-supervised student-teacher pretraining with layer-wise differentiable tokenization. Rather than relying on a post-hoc k-means quantizer, SpidR 2.0 learns multiple discrete unit spaces jointly during pretraining. Shared prediction heads map student and teacher representations into common, stable unit spaces, yielding high-quality discrete units at multiple encoder depths. A **multi-stream SLM** then jointly models the unit streams extracted from these layers. Its shared LM backbone captures temporal dependencies across sequence positions, stream-specific heads model transitions within each stream, and a generative head captures cross-stream dependencies. By distributing speech properties across complementary streams, GSLM2 preserves fine-grained distinctions while exposing higher-level regularities for efficient language modeling, thereby narrowing the performance gap with text LMs (e.g., Figure 1). Empirically, GSLM2 shows improved scaling at model sizes up to 6.5B parameters with 6M hours of speech data and outperforms competing systems, including speech-text models, across several key metrics. These advances provide a foundation for speech-first, textless applications that model both linguistic and phonetic content directly from speech. We will release the models and code upon acceptance to support further research and application development.

2 GSLM2

GSLM (Lakhotia et al., 2021) aims to model the distribution of speech using discrete speech units. A standard GSLM pipeline consists of three separately trained components: a speech tokenizer, an SLM, and a unit vocoder. The speech tokenizer maps a continuous speech signal into a sequence of discrete units $Q = (q_1, \ldots, q_U)$, typically by applying k-means clustering to hidden representations from a self-supervised speech encoder (Hsu et al., 2021). As in conventional language modeling, the SLM autoregressively predicts the next speech unit conditioned on the preceding units. Finally, the unit vocoder (Polyak et al., 2021) converts speech units back into a waveform. We focus here on the standard GSLM formulation; Appendix A discusses related work on speech tokenization, multi-level speech representations, and multi-stream sequence modeling.

The tokenizer is central to GSLM because it determines which speech properties are exposed to the SLM. Standard tokenizers generally quantize representations from a single encoder layer, restricting the resulting units to one level of abstraction. However, self-supervised speech encoders distribute complementary phonetic and semantic information across layers (Pasad et al., 2021; Shen et al., 2023). Standard GSLMs therefore leave much of this multi-level information unexploited.

We propose GSLM2, an extension of GSLM comprising two components, as illustrated in Figure 1. First, we introduce **SpidR 2.0**, an encoder pretraining method that learns structured and stable code distributions throughout the encoder. Second, we propose a **multi-stream SLM** that jointly models unit streams extracted from different encoder layers. Together, these components enable GSLM2 to combine complementary speech information at multiple levels of representation.

2.1 Encoder Pretraining: SpidR 2.0

SpidR 2.0 builds on the DinoSR and SpidR family (Liu et al., 2023; Poli et al., 2025b) within a student-teacher framework (Grill et al., 2020). We extend SpidR with differentiable tokenization and entropy regularization to learn stable, informative code distributions across multiple encoder layers. The student and teacher are L-layer Transformer encoders. The student receives a masked input sequence (Hsu et al., 2021), while the teacher processes its unmasked counterpart. At each selected layer, the student predicts targets derived from the corresponding teacher layer. The teacher encoder parameters are updated as an exponential moving average (EMA) of the student parameters.

Layer-wise differentiable tokenization. DinoSR and SpidR construct discrete targets by assigning teacher representations to the nearest entries in separate codebooks updated by EMA. SpidR 2.0 instead uses *shared prediction heads* to map teacher and student representations into a common code space at each selected layer, inspired by Zhou et al. (2022). These heads produce soft code assignments, making tokenization fully differentiable during pretraining and eliminating the need to maintain separate online codebooks (Liu et al., 2023).

We attach a layer-specific prediction head to each encoder layer in a selected set $\mathcal{K} \subseteq \{1, \dots, L\}$ (as "Proj." in Figure 1). For every $l \in \mathcal{K}$, the same head is applied to the corresponding teacher and student representations and emits a distribution over a vocabulary of C codes. During pretraining, these distributions serve as soft code assignments; at inference, they are converted by argmax into discrete units for SLMs. For each selected layer l, the prediction head $\phi_l : \mathbb{R}^{d_{\text{enc}}} \to \mathbb{R}^C$ consists of a layer normalization across channels, without scale or bias parameters, followed by a linear projection to C-dimensional logits. The projection matrix at layer l has C rows, with row i given by $\mathbf{c}_{l,i}^\top$, where $\mathbf{c}_{l,i} \in \mathbb{R}^{d_{\text{enc}}}$ is the embedding of the i-th codeword. We parametrize the rows of each projection layer $\mathbf{c}_{l,i}$ to have unit L_2 norm, hence pushing the codewords onto the unit sphere $\mathcal{U}(\mathbb{R}^{d_{\text{enc}}})$. The backward pass differentiates through this parametrization. Let $\mathbf{h}_{l,t}, \tilde{\mathbf{h}}_{l,t} \in \mathbb{R}^{d_{\text{enc}}}$ denote the teacher and student representations at layer l and frame t, respectively. Using the shared head, the teacher target and student prediction are computed as

$$\mathbf{p}_{l,t} = \text{Softmax}\Big(\phi_l\big(\text{IN}(\mathbf{h}_{l,t})\big)\Big), \qquad \tilde{\mathbf{p}}_{l,t} = \text{Softmax}\big(\phi_l(\tilde{\mathbf{h}}_{l,t})\big), \tag{1}$$

where $\text{IN}(\cdot)$ indicates instance normalization on the time dimension, and $\mathbf{p}_{l,t}, \tilde{\mathbf{p}}_{l,t} \in \Delta^{C-1}$ are distributions over the shared code vocabulary. Each selected layer therefore is associated with a differentiable tokenizer whose code space is optimized directly by distillation rather than through independently maintained online codebooks. The same code space can be used directly for speech unit extraction, eliminating the need for post-hoc, non-differentiable k-means on continuous representations (Lakhotia et al., 2021), while also improving performance (see Appendix C.2.8).

The self-supervised objective is the cross-entropy between the detached teacher targets and student predictions, averaged over the selected layers $\mathcal{K}$ and masked frames $\mathcal{M}$:

$$\mathcal{L}_{\text{ssl}} = -\frac{1}{|\mathcal{K}|} \sum_{l \in \mathcal{K}} \frac{1}{|\mathcal{M}|} \sum_{t \in \mathcal{M}} \sum_{c=1}^{C} \text{sg}(\mathbf{p}_{l,t}[c]) \, \log \tilde{\mathbf{p}}_{l,t}[c], \tag{2}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator. Gradients are propagated only through the student branch, while the teacher encoder parameters are updated using an EMA of the student parameters.

Entropy regularization. To stabilize the shared prediction heads, we regularize the teacher distributions to avoid both codebook collapse and overly diffuse assignments. Following the ME-MAX

principle for equipartition-constrained clustering (Assran et al., 2022), we combine a *conditional-entropy* term $\mathbb{E}_t[H_c(\mathbf{p}_{l,t})]$ with a *marginal-entropy* term $H_c(\mathbb{E}_t[\mathbf{p}_{l,t}])$ as

$$\mathcal{L}_{\mathsf{reg}} = \frac{1}{|\mathcal{K}|} \sum_{l \in \mathcal{K}} \Big[\mu \, \mathbb{E}_t \big[H_c(\mathbf{p}_{l,t}) \big] - \lambda \, H_c\big(\mathbb{E}_t[\mathbf{p}_{l,t}] \big) \Big], \tag{3}$$

where μ and λ control the strengths of the two terms. Minimizing the conditional entropy encourages confident code assignments for individual frames, while maximizing the marginal entropy promotes broad code usage across frames. Together, these objectives push for stable and informative discrete representations across multiple encoder layers, whereas SpidR produces its highest-quality units primarily at a single layer (see Section 3.3). Applying this loss on the teacher distribution only ensures that the gradients from this regularization loss do not backpropagate to the encoder, but only constrain the unit space. A study on these regularization choices is provided in Appendix C.2.4.

Training objective. The overall objective for encoder pretraining combines the self-supervised distillation loss from Eq. (2) with the entropy regularization term in Eq. (3) as $\mathcal{L}_{\mathsf{enc}} = \mathcal{L}_{\mathsf{ssl}} + \mathcal{L}_{\mathsf{reg}}$.

2.2 Multi-Stream Spoken Language Modeling

The multi-stream SLM jointly models speech units extracted from multiple encoder layers. Let $\mathcal{J} \subseteq \mathcal{K}$ be the set of layers used as unit streams. For each $l \in \mathcal{J}$, we run the student encoder without input masking and extract the most likely code at each frame: $z_{l,t} = \arg\max_c \tilde{\mathbf{p}}_{l,t}[c]$. This gives a raw unit sequence $Z_l = (z_{l,1}, \ldots, z_{l,T})$ for each stream. To shorten the sequence while preserving changes in any stream, we apply *cross-stream deduplication*: for $t > 1$, frame t is removed only if $z_{l,t} = z_{l,t-1}$ for all $l \in \mathcal{J}$, and is retained otherwise. The retained frames thus form the union of the change points across all streams. As a result, a stream contains repeated units at consecutive positions when another stream changes, unlike independent single-stream deduplication. The resulting multi-stream sequence is $\mathcal{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_U)$, where $\mathbf{q}_u = (q_{l,u} \mid l \in \mathcal{J})$ is the tuple of stream units at the u-th retained frame, and U is the deduplicated sequence length shared by all streams. The multi-stream SLM models the joint distribution over $\mathcal{Q}$ as $p(\mathcal{Q}) = \prod_{u=1}^{U} p(\mathbf{q}_u \mid \mathbf{q}_1, \ldots, \mathbf{q}_{u-1})$.

Architecture. Each stream has its own embedding layer, $\mathrm{Emb}_l : \{1, \ldots, C\} \to \mathbb{R}^{d_{\mathsf{slm}}}$, which maps a discrete unit to the d_{slm}-dimensional SLM embedding space. At position u, the stream units are embedded independently and averaged: $\mathbf{x}_u = \frac{1}{|\mathcal{J}|} \sum_{l \in \mathcal{J}} \mathrm{Emb}_l(q_{l,u}) \in \mathbb{R}^{d_{\mathsf{slm}}}$. A shared decoder-only Transformer processes the preceding embeddings $(\mathbf{x}_1, \ldots, \mathbf{x}_{u-1})$ and produces a context embedding $\mathbf{y}_u \in \mathbb{R}^{d_{\mathsf{slm}}}$. This shared temporal backbone captures dependencies across sequence positions.

The context embedding is then fed to two types of output heads: *stream-specific heads* and a *generative head*, as illustrated in Figure 1. For each stream $l \in \mathcal{J}$, the *stream-specific head* predicts the next unit from the shared context $\mathbf{y}_u$ (or equivalently from the preceding tuple sequence) as

$$p_l^{\mathsf{ms}}(\cdot \mid \mathbf{q}_1, \ldots, \mathbf{q}_{u-1}) = \mathrm{Softmax}\big(\mathrm{Linear}_l(\mathbf{y}_u)\big), \tag{4}$$

where $\mathrm{Linear}_l : \mathbb{R}^{d_{\mathsf{slm}}} \to \mathbb{R}^C$. These heads provide explicit per-stream supervision for the unit transition dynamics, encouraging the shared backbone to preserve structure useful for joint prediction.

The stream-specific heads predict current units conditioned only on the shared multi-stream history. Sampling independently from these predicted distributions therefore imposes conditional independence among current units. To model dependencies within each tuple, we introduce a *generative head* based on the depth Transformer architecture (Lee et al., 2022). Let $\pi : \{1, \ldots, |\mathcal{J}|\} \to \mathcal{J}$ be a fixed bijection specifying an ordering of the streams in $\mathcal{J}$. Starting from the context embedding $\mathbf{y}_u$, the generative head predicts the units in $\mathbf{q}_u$ autoregressively according to the ordering π as

$$p(\mathbf{q}_u \mid \mathbf{q}_1, \ldots, \mathbf{q}_{u-1}) = \prod_{j=1}^{|\mathcal{J}|} p\big(q_{\pi(j),u} \mid \mathbf{y}_u, q_{\pi(1),u}, \ldots, q_{\pi(j-1),u}\big). \tag{5}$$

Thus, the shared Transformer models temporal dependencies across sequence positions, while the generative head captures dependencies among streams within each position. During generation, the units in each tuple are sampled sequentially according to the ordering π.

Training. The multi-stream SLM is trained with a combined objective consisting of a single-stream loss, a transition-level self-distillation loss, and a multi-stream generative loss.

Single-stream loss. For each training example, we sample a stream $l \sim \mathrm{Unif}(\mathcal{J})$ and independently deduplicate its frame-level sequence Z_l. This yields $\bar{Q}_l = (\bar{q}_{l,1}, \ldots, \bar{q}_{l,\bar{U}_l})$, where adjacent units are always distinct, and $\bar{U}_l$ is the deduplicated sequence length. At position u, the model takes the stream-specific embedding $\mathrm{Emb}_l(\bar{q}_{l,u})$ as input. The resulting sequence is processed by the shared temporal Transformer, and next-unit predictions are produced using the same stream-specific head Linear_l as in multi-stream mode (cf. Eq. (4)). The single-stream loss is defined as

$$\mathcal{L}_{\mathsf{single}} = -\mathbb{E}_{l\sim\mathrm{Unif}(\mathcal{J})}\left[\frac{1}{\bar{U}_l}\sum_{u=1}^{\bar{U}_l}\log p_l^{\mathsf{ss}}(\bar{q}_{l,u} \mid \bar{q}_{l,1}, \ldots, \bar{q}_{l,u-1})\right]. \tag{6}$$

Because consecutive repetitions have been removed, this objective emphasizes transitions between distinct units within each stream.

Transition-level self-distillation. The cross-stream-deduplicated sequence $\mathcal{Q}$ contains additional positions introduced by changes in other streams. To transfer the transition behavior learned in single-stream mode to these multi-stream contexts, we align the multi-stream predictive distribution with a detached single-stream reference distribution. Let $\mathcal{A}_l = (a_{l,1}, \ldots, a_{l,\bar{U}_l})$ denote the positions in $\mathcal{Q}$ corresponding to the successive units in $\bar{Q}_l$. Thus, $q_{l,a_{l,u}} = \bar{q}_{l,u}$ and, for $u > 1$, $a_{l,u}$ is a retained position at which stream l changes. At each aligned position, we define the self-distillation loss as

$$\mathcal{L}_{\mathsf{distill}} = \mathbb{E}_{l\sim\mathrm{Unif}(\mathcal{J})}\left[\frac{1}{\bar{U}_l}\sum_{u=1}^{\bar{U}_l}\mathrm{KL}\Big(\mathrm{sg}\big(p_l^{\mathsf{ss}}(\cdot \mid \bar{q}_{l,1}, \ldots, \bar{q}_{l,u-1})\big) \,\|\, p_l^{\mathsf{ms}}\big(\cdot \mid \mathbf{q}_1, \ldots, \mathbf{q}_{a_{l,u}-1}\big)\Big)\right]. \tag{7}$$

The single-stream distribution provides rich supervision focused on transitions between distinct units. Matching this distribution at aligned positions encourages the shared temporal backbone to capture per-stream transition structure efficiently from the complex multi-stream context.

Multi-stream generative loss. The generative head is trained on the cross-stream-deduplicated sequence $\mathcal{Q}$. Using the factorization in Eq. (5), the multi-stream generative loss is

$$\mathcal{L}_{\mathsf{gen}} = -\frac{1}{U|\mathcal{J}|}\sum_{u=1}^{U}\sum_{j=1}^{|\mathcal{J}|}\log p(q_{\pi(j),u} \mid \mathbf{y}_u, q_{\pi(1),u}, \ldots, q_{\pi(j-1),u}). \tag{8}$$

This objective trains the model to predict a coherent tuple at each position by accounting for both the preceding sequence and units generated earlier within the current tuple.

The final multi-stream SLM objective is $\mathcal{L}_{\mathsf{slm}} = (1-\alpha)\left[(1-\beta)\mathcal{L}_{\mathsf{single}} + \beta\mathcal{L}_{\mathsf{distill}}\right] + \alpha\mathcal{L}_{\mathsf{gen}}$, where α controls the contribution of multi-stream generation and β balances single-stream modeling against transition-level self-distillation. The two types of heads play complementary roles in training the shared temporal backbone. The stream-specific heads provide supervision on transitions between distinct units, while the generative head models the joint distribution over complete tuples, including per-stream unit persistence and within-tuple dependencies (see Appendix C.1.5 for ablation studies).

3 Experiments

3.1 Experimental Setup

Models. We compare four models: **SpidR SLM**, trained on discrete units extracted by the original SpidR encoder (Poli et al., 2025b); **SpidR 2.0 SLM**, trained on a single unit stream extracted by our SpidR 2.0 encoder; **GSLM2**, our multi-stream model trained on unit streams extracted from multiple SpidR 2.0 layers; and **Text LM**, trained on text transcriptions as a topline. Where applicable, we match model size and training data across systems to enable controlled comparisons.

Data. We mainly use the English Libri-Light corpus (Kahn et al., 2020). The speech encoders are pretrained on the 960-hour LibriSpeech corpus (Panayotov et al., 2015), while the SLMs are trained on the 600-, 6K-, and 60K-hour Libri-Light subsets. The corresponding Text LMs are trained on Libriheavy transcriptions (Kang et al., 2024). To study model scaling beyond Libri-Light, we additionally use a 6M-hour corpus of English conversational speech sourced from the internet. We train SLMs on progressively larger subsets, ranging from 600 hours to the full 6M hours, and train

Table 1: Spoken-language evaluation suite. Metrics described with "vs." use likelihood-based paired comparisons, while the others assess prompted generation.

Benchmark	Metric	Capability	Evaluation
ZeroSpeech'21	**sWuggy(-all)**	Lexical	Real word vs. phonotactically matched nonword.
	sBLIMP	Syntax	Grammatical vs. minimally different ungrammatical sentence.
StoryCloze (SC)	**Topic SC (tSC)**	Semantics	Correct ending vs. random ending from another story.
	Spoken SC (sSC)	Semantics	Correct ending vs. adversarial, superficially plausible ending.
LongTail-Swap	**WordSwap (WS)**	Semantics	Context-appropriate word vs. frequency- and POS-matched substitution.
	InflectionSwap (IS)	Morphosyntax	Context-appropriate inflection vs. another form of the same lemma.
	AgreementSwap (AS)	Syntax	Matching vs. mismatching grammatical agreement.
Continuation	**ASR-PPL**	Fluency	LM perplexity of an ASR-transcribed continuation at a calibrated temperature.
	LLM Judge	Quality	Rubric-based relevance, coherence, and fluency of the transcribed continuation.

the corresponding Text LMs on transcripts generated by automatic speech recognition (ASR). For each setting, we train a byte-pair encoding (BPE) (Sennrich et al., 2016) tokenizer with a vocabulary of 4,096 tokens, using the corresponding text data. Full dataset details are provided in Appendix B.1.

Architecture. For both SpidR and SpidR 2.0, the student and teacher encoders each comprise $L = 12$ Transformer encoder layers and contain 90M parameters. For SpidR 2.0, the shared prediction heads are applied to layers $\mathcal{K} = \{5, 6, \ldots, 12\}$, each projecting to a vocabulary of $C = 256$ codes. The SLMs and Text LMs use decoder-only Transformers instantiated at five parameter scales: 90M, 300M, 1.2B, 2.5B, and 6.5B parameters. Full architecture details are provided in Appendix B.2.

Training. Encoder pretraining largely follows the setup used for SpidR. For SpidR 2.0, the conditional-entropy weight is fixed at $\mu = 0.1$, while the marginal-entropy weight is initialized at $\lambda = 0.1$ and reduced to 0.001 after 20k steps. SpidR SLM uses units from layer 6, while SpidR 2.0 SLM uses layer 7. GSLM2 uses six streams extracted from layers $\mathcal{J} = \{5, 6, \ldots, 10\}$ and generates them in ascending order of layer-wise bitrate, following $(10, 5, 6, 9, 8, 7)$. Layer selections are based on development-set performance. We set both loss coefficients, α and β, in $\mathcal{L}_{\mathsf{slm}}$ to 0.1. Appendix B.3 provides full training details and motivates our hyperparameter choices.

Evaluation. We evaluate complementary linguistic capabilities using the metrics summarized in Table 1. For the paired-stimulus tasks from ZeroSpeech'21 (ZS21) (Nguyen et al., 2020), StoryCloze (SC) (Mostafazadeh et al., 2016; Hassid et al., 2023), and LongTail-Swap (LT-Swap) (Algayres et al., 2025), SLMs score spoken stimuli and Text LMs score their textual counterparts; accuracy is the proportion of pairs for which the correct item receives the higher score. The single-stream models use log-likelihoods averaged over units or tokens; for sWuggy, Text LMs instead use unnormalized log-likelihoods (Poli et al., 2025b). GSLM2 uses the stream-specific heads, averaging log probabilities over unit-change positions within each stream and then across streams. This evaluates distinct-unit transitions from temporal context, following the scoring convention of the independently deduplicated single-stream SLMs. The resulting transition score differs from the joint sequence likelihood computed with the generative head, which also accounts for per-stream repetitions and within-tuple dependencies. Further analysis of scoring choices is provided in Appendix C.1.6. Our primary metrics are **sWuggy**, **sBLIMP**, and **tSC**, widely used measures of lexical, syntactic, and semantic competence, respectively. We also report sSC, a more challenging variant of tSC that requires greater story-level contextual understanding. We additionally develop a spoken adaptation of LT-Swap that evaluates performance across word-frequency bins using WordSwap (WS), InflectionSwap (IS), and AgreementSwap (AS), which probe semantic, morphosyntactic, and syntactic abilities, respectively. We report the macro-average of the per-bin accuracies. Alongside the paired-stimulus tasks, we assess generation quality by producing prompt-conditioned continuations with top-p sampling, synthesize them using a duration predictor and unit vocoder, and transcribe the resulting speech via ASR. For GSLM2, the generative head samples each tuple autoregressively. For each model, we calibrate the decoding temperature on a held-out set using ground-truth repetition statistics measured by uni/bigram auto-BLEU (Lakhotia et al., 2021). We then compute perplexity (PPL) over the ASR transcriptions (ASR-PPL) using Llama3 (Grattafiori et al., 2024) and assess the transcribed continuations using a large language model (LLM) judgment score (LLMJ) assigned by Prometheus 2 (Kim et al., 2024). For speech encoders, we evaluate their representations using zero-shot metrics, including triphone-level ABX discrimination (Schatz, 2016) and word-level mean average precision (MAP) (Carlin et al., 2011). Full evaluation details are provided in Appendix B.4.

Table 2: Main results on ZeroSpeech'21 (ZS21), StoryCloze (SC), and LongTail-Swap (LT-Swap) for SLMs trained on Libri-Light at three scales. Scale labels indicate "train data hours / parameter count". At each scale, the Text LM serves as a topline. Scores are accuracies (%; chance level: 50%).

		ZS21 (↑)		SC (↑)		LT-Swap (↑)		
	Model	sWug.	sBLI.	tSC	sSC	WS	IS	AS
600 h / 90M	Text LM	73.4	61.7	75.4	53.4	73.5	80.6	62.2
	SpidR SLM	67.4	54.2	65.4	52.4	54.3	57.9	54.2
	SpidR 2.0 SLM	69.2	57.7	68.1	**52.7**	58.0	63.5	55.0
	GSLM2	**71.5**	**59.8**	**74.1**	52.0	**62.4**	**68.2**	**59.1**
6K h / 300M	Text LM	79.5	66.8	84.8	52.2	80.1	90.9	79.1
	SpidR SLM	71.1	59.3	71.4	50.7	62.0	63.0	58.5
	SpidR 2.0 SLM	73.9	65.1	74.7	52.2	67.5	69.0	64.9
	GSLM2	**76.3**	**67.2**	**78.9**	**52.5**	**75.6**	**75.5**	**69.8**
60K h / 1.2B	Text LM	83.8	71.5	89.2	54.6	87.3	94.2	85.4
	SpidR SLM	73.9	62.4	75.1	52.8	66.4	65.5	60.7
	SpidR 2.0 SLM	76.7	69.6	81.0	56.2	74.8	72.5	66.6
	GSLM2	**79.3**	**71.8**	**84.5**	**57.6**	**79.5**	**78.5**	**69.9**

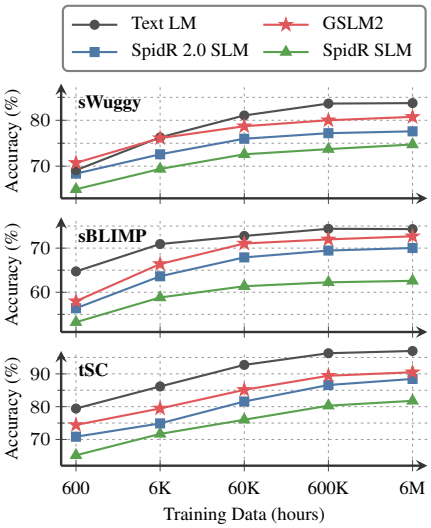


Figure 2: Scaling trends for SLMs and Text LM trained on increasing amounts of internet-sourced conversational speech, up to 6M h.

3.2 SLM Evaluation

Main results. Table 2 compares the SLM variants across the three Libri-Light scales, with Text LMs serving as reference toplines. On sWuggy, sBLIMP, and tSC, SpidR 2.0 SLM consistently outperforms SpidR SLM at every scale, suggesting that its units better preserve linguistically relevant information, which is examined in Section 3.3. Multi-stream modeling provides substantial further gains: GSLM2 achieves the best overall SLM performance at every scale, considerably narrowing the gap with Text LM. Although sSC scores generally remain close to chance, both SpidR 2.0 SLM and GSLM2 improve at 60K hours, notably outperforming Text LM. On LT-Swap, GSLM2 consistently outperforms both single-stream SLMs, but large gaps to Text LM remain. The gaps on IS and AS, which probe syntactic abilities, widen with scale and are particularly pronounced at 60K hours. Because LT-Swap is constructed from the training data at each scale, however, these comparisons do not provide a controlled assessment of scalability. Yet, the remaining gaps suggest that LT-Swap reveals limitations in spoken language modeling that are less apparent on ZS21 and SC, highlighting the need for broader evaluation coverage. Appendix C.1.9 further analyzes the LT-Swap results.

Scalability study. Results so far cover scaling on Libri-Light up to 60K hours (Table 2, Figure 1). To probe scaling further, we turn to conversational speech from the internet, reaching 6M hours. Unlike the read audiobook speech of Libri-Light, this is spontaneous conversational speech, letting us test whether our conclusions hold across domains (see Appendix B.1 for details). Figure 2 reports sWuggy, sBLIMP, and tSC as we scale models to 2.5B and 6.5B parameters, trained on 600K and 6M hours respectively. Relative to SpidR SLM, GSLM2 substantially narrows the gap to the Text LM toplines at every scale and on every metric. At 6M hours, the additional data an SLM would need to match the Text LM falls from roughly four orders of magnitude for SpidR SLM, to two orders for GSLM2 on the average metric. Our two modeling improvements contribute differently: the gain from SpidR 2.0 units compounds with scale, whereas multi-stream modeling contributes a roughly constant error reduction at every scale (see Appendix C.1.2 for detailed scaling analyses).

Comparison against previous SLMs. Table 3 compares GSLM2 with previous SLMs trained under different regimes. Among models trained solely on speech, GSLM2 achieves the best overall performance across model and data scales. Despite using a similar architecture, our 1.2B model outperforms the 7B Moshi on all four metrics, highlighting strong parameter and data efficiency. Scaling GSLM2 to 6.5B parameters yields further gains. GSLM2 also remains competitive with models that use pretrained LMs or text data. Such models perform well on semantic tasks (i.e., tSC and sSC), but tend to lag on sWuggy and sBLIMP, which can require finer lexical and syntactic distinctions. GSLM2 leads on the latter benchmarks while remaining competitive on semantic tasks, suggesting that its multi-level units support broad and balanced linguistic competence. Appendix C.1.8 provides additional results for speech-capable LLMs trained at much larger scales.

Table 3: Comparison of GSLM2 with prior SLMs under different training regimes. Prior results are from cited papers unless marked ‡ (our evaluation). Bold indicates the best overall result; underlining indicates the best prior baseline in each regime. †The amount of speech data used for training.

Training Regime	Model	#Params	#Data† [h]	sWuggy	sBLIMP	tSC	sSC
Speech-only training	GSLM2 (ours; Libri-Light)	300M	6K	76.3	67.2	78.9	52.5
		1.2B	60K	79.3	71.8	84.5	57.6
	GSLM2 (ours; Conversational Speech)	6.5B	6M	**80.7**	**72.9**	**90.5**	**63.2**
	GSLM (Lakhotia et al., 2021)	100M	6K	64.8	54.2	66.6	53.3
	AudioLM (Borsos et al., 2023)	300M	60K	71.5	64.7	–	–
	Cuervo & Marxer (2024)	823M	160K	–	61.3	75.4	–
	Moshi (Défossez et al., 2024)	7B	7M	74.8	59.9	80.9	56.9
	SyllableLM (Baade et al., 2025)	300M	60K	72.2	63.7	75.4	–
Speech-only training + text-LM init.	TWIST (Hassid et al., 2023)	13B	150K	74.5	59.2	76.4	55.4
	Moshi (Défossez et al., 2024)	7B	7M	74.3	58.9	81.8	58.7
	Flow-SLM (Chou et al., 2025)	7B	45K	69.8	60.0	–	–
	Llama-Mimi (Sugiura et al., 2025)	8B	240K	68.8	55.1	67.6	–
Speech-text training + text-LM init.	VoxtLM (Maiti et al., 2024)	1.3B	60K	66.1	57.1	–	–
	Moshi (Défossez et al., 2024)	7B	7M	72.6	58.8	83.0	60.8
	SpiRit-LM (Nguyen et al., 2025)	7B	458K	69.0	58.3	82.9	61.0
	Align-SLM (Lin et al., 2025)	7B	4K	77.9	62.3	86.8	61.1
	BSLM (Luo et al., 2026)	4B	220K	74.1	60.2	84.1	61.3
	LST (Lu et al., 2026)	1.8B	164K	72.8	59.0	79.5	60.8
	PersonaPlex‡ (Roy et al., 2026)	7B	2K	65.4	54.9	80.1	58.9

Generative evaluation. Table 4 reports prompt-conditioned continuation results on ASR-PPL and LLMJ. We also report the decoding temperature $\hat{\tau}$, calibrated per model against ground-truth repetition statistics (see Appendix B.4.4 for details). Among the SLMs, SpidR 2.0 consistently improves over SpidR, while GSLM2 provides further gains, suggesting that the advantages in paired-stimulus evaluation transfer to generation. Text LM shows a different trend: with more training data, its PPL improves, but its LLMJ score worsens relative to SLMs. This may reflect a granularity mismatch between fine-grained speech units and coarser BPE tokens, with SLMs better preserving the local lexical coherence favored by LLMJ. In the 60K-hour setting, for example, GSLM2 produces 33% more continuations rated 4 or 5 than Text LM. The LLM judge explicitly describes nearly all of these high-scoring continuations as coherent in its rationales. GSLM2 also outperforms Text LM on ASR-PPL in the 600- and 6K-hour settings, though these comparisons are not strictly fair because SLM outputs are evaluated through ASR. The calibrated temperatures are also noticeably higher than those typically used for LM decoding. Appendix C.1.3 details the temperature calibration process, while Appendix C.1.10 examines the effects of evaluation protocol differences and presents fixed-temperature comparisons.

Table 4: Prompt-conditioned continuation results. $\hat{\tau}$ is the decoding temperature calibrated with ground-truth repetition statistics.

	Model	$\hat{\tau}$	ASR-PPL (↓)	LLMJ (↑)
600h / 90M	Text LM	1.22	690.0 ± 12.9	1.88 ± 0.02
	SpidR SLM	1.31	670.5 ± 9.4	1.73 ± 0.02
	SpidR 2.0 SLM	1.34	626.5 ± 14.3	1.78 ± 0.02
	GSLM2	1.49	**583.5** ± 12.1	**1.80** ± 0.02
6K h / 300M	Text LM	1.21	331.0 ± 6.5	2.30 ± 0.03
	SpidR SLM	1.39	407.2 ± 9.7	2.20 ± 0.02
	SpidR 2.0 SLM	1.31	367.5 ± 8.0	2.27 ± 0.03
	GSLM2	1.58	**326.9** ± 11.3	**2.43** ± 0.03
60K h / 1.2B	Text LM	1.20	172.7 ± 6.8	2.87 ± 0.04
	SpidR SLM	1.50	259.4 ± 7.3	2.78 ± 0.03
	SpidR 2.0 SLM	1.45	247.4 ± 3.5	2.89 ± 0.05
	GSLM2	1.59	**189.1** ± 5.5	**3.05** ± 0.04

3.3 Evaluation of Encoder Representations

Layer-wise evaluations. To examine how complementary information is distributed across the SpidR 2.0 encoder layers, we evaluate representations from each layer using ABX and MAP. Table 5 compares SpidR 2.0 with official checkpoints of SpidR and DinoSR. To our knowledge, SpidR achieves the best previously reported ABX performance among self-supervised speech encoders. SpidR 2.0 outperforms both encoders on ABX and matches DinoSR on MAP, with phoneme-/word-level specialization at layers 10 and 7, respectively. Appendix C.2.10 extends this comparison to a broader set of speech encoders, further supporting the effectiveness of SpidR 2.0 on both metrics.

Table 5: ABX and MAP results for DinoSR, SpidR, and SpidR 2.0, reported at the best layer l^* for each metric.

	ABX (↓)				MAP (↑)	
Model	l^*	within	across	Avg.	l^*	Score
DinoSR	5	4.58	6.02	5.30	11	69.49
SpidR	6	3.53	4.31	3.92	5	63.42
SpidR 2.0	10	**2.54**	**3.13**	**2.83**	7	**69.58**

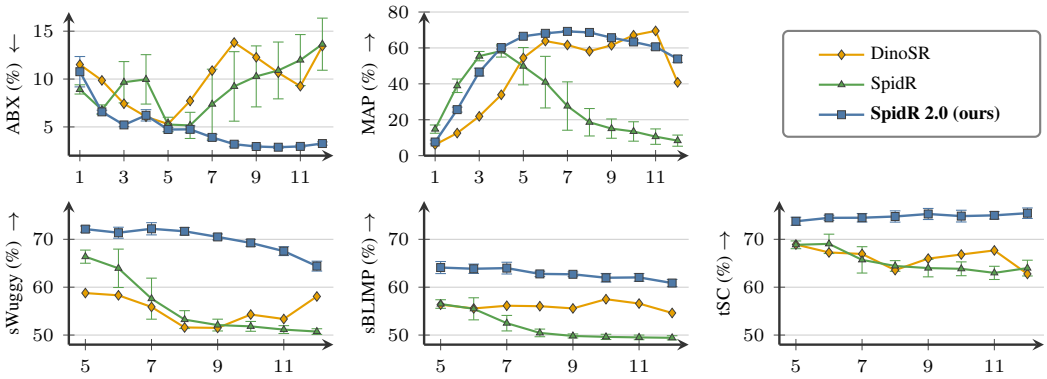


Figure 3: Layer-wise ABX, MAP, and single-stream SLM results. SLMs have 90M parameters and are trained on 6K hours of Libri-Light. The horizontal axis denotes the layer index l. Error bars for SpidR and SpidR 2.0 indicate the standard deviation across eight encoder-training seeds.

Figure 3 compares the layer-wise ABX and MAP performance across the three speech encoders. For SpidR and SpidR 2.0, we report results from eight encoders initialized and trained with different random seeds. SpidR 2.0 maintains strong ABX and MAP scores from layer 5 onward, whereas SpidR and DinoSR peak sharply at a single layer. To further probe layer-wise specialization, we train single-stream SLMs on units associated with each layer and evaluate them on sWuggy, sBLIMP, and tSC. We use the best-performing discretization method for each model at layer l: SpidR applies k-means to representations from student layer l; DinoSR predicts codebook assignments from teacher layer l using the corresponding prediction head at the final student layer; and SpidR 2.0 uses codebook prediction from student layer l. For SpidR 2.0, sWuggy and sBLIMP scores are highest with units from shallower layers, whereas tSC benefits more from units extracted from deeper layers. Every evaluated SpidR 2.0 layer outperforms SpidR and DinoSR. These findings suggest that SpidR 2.0 learns consistently high-quality representations that capture complementary linguistic information across layers, providing a strong foundation for multi-stream modeling. Appendices C.2.1, C.2.7, and C.2.9 further study these differences through analyzing codebook properties, probing representations on supervised downstream tasks, and assessing stream similarities, respectively.

Layer-wise stability across training runs. Figure 3 shows that SpidR 2.0 achieves more stable unit quality across training runs than SpidR. At the best-performing layer, SpidR 2.0 reduces the standard deviation of ABX across training seeds by a factor of ten relative to SpidR. The difference is most pronounced at layers 7-12, where SpidR exhibits substantial variation on ABX and MAP scores across seeds, while SpidR 2.0 consistently preserves high-quality representations. This stability enables more reliable unit extraction for subsequent SLM training. Ablations in Appendix C.2.3 show how each design change from SpidR to SpidR 2.0 affects representation quality and stability.

4 Conclusion

We presented GSLM2, a framework for efficient spoken language modeling at multiple levels of abstraction. SpidR 2.0 learns stable, complementary discrete unit spaces across multiple encoder layers, while the multi-stream SLM captures dependencies within and across the resulting streams. Across data scales, model sizes, and domains, GSLM2 improved lexical, syntactic, semantic, and generation performance, narrowing the data-efficiency gap with text LMs. These findings suggest multi-level speech representations as a promising foundation for efficient speech-native modeling.

Limitations. Here, § points to related appendix sections. This work primarily focuses on modeling linguistic and phonetic content from speech without explicitly preserving acoustic properties such as prosody, which are important for tasks including speech resynthesis (§C.3). Multi-stream modeling also increases computational cost, particularly during training (§B.3). Our analyses are limited to empirical comparisons and probing, providing only a partial understanding of how individual streams contribute to GSLM2 and relate to one another (§C.1.7 and C.2.9). Finally, our evaluation relies on synthetic benchmarks and indirect generation metrics that, despite their widespread use, may conflate language modeling errors with those introduced by TTS, vocoding, or ASR (§C.1.9, C.1.10, and C.1.12).

AI USE STATEMENT

In this work, we used generative AI systems to construct portions of our spoken-language benchmarks, including translating source material, filtering samples for suitability for speech synthesis, and synthesizing speech using text-to-speech models (see Appendices B.4.1-B.4.3). Generative AI also provided feedback on experimental design and assisted in implementing some portions of the method and its associated training and evaluation procedures. We further used LLMs as automated judges during generative evaluation (see Appendix B.4.5). We did not use generative AI to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide arguments for or assist in writing mathematical proofs, propose or refine research hypotheses, or interpret experimental results.

Additionally, we used generative AI to suggest experimental parameters; generate, edit, and debug code; produce ASR-generated transcriptions for model training and generative evaluation (see Appendices B.1 and B.4); draft and improve the readability of manuscript text; search for, identify, and summarize relevant literature; and support brainstorming about potential experimental directions. Apart from outputs produced as part of the explicitly described benchmark-construction and evaluation pipelines, AI outputs were treated as suggestions and incorporated only after author review.

AI-assisted code was manually inspected by at least one author, compared against reference implementations where applicable, and validated by reproducing the reported results. The entire manuscript was reviewed by the authors. Literature summaries, citations, and factual claims were checked against the original publications. Potential overlap with prior work was assessed through a manual literature survey. Experimental results were verified through repeated runs, inspection of raw outputs, statistical analyses, and independent review by other authors. The authors were responsible for interpreting the results and drawing scientific conclusions. We take responsibility for the final content of this work, including all text, claims, code, data, and other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We document our methods and experimental procedures throughout the main paper, appendices, and supplementary materials. Section 2 presents the proposed method and training procedure. Section 3.1 provides a concise overview of the experimental settings, while Appendices B.2 and B.3 provide further implementation details, including model configurations, optimization settings, computational resources, and experimental hyperparameters. Appendix B.1 identifies the source datasets and explains the preprocessing procedures. Appendix B.4 describes the public benchmarks used in our experiments, details our benchmark-construction process (including translation, filtering, speech synthesis, and quality control), and specifies the evaluation protocols, metrics, prompts, and model versions. Appendices C.1 and C.2 provide additional analyses supporting our hyperparameter choices across multiple components of the method. To assess the statistical reliability of our findings, Appendix C.4 quantifies statistical uncertainty in model comparisons using confidence intervals and assesses variability across encoder training runs with different random seeds. The supplementary materials include anonymized source code, configuration files, prompts, and data-processing and evaluation scripts used in the main experiments. Upon acceptance, we will release code, model checkpoints, and derived benchmark artifacts as permitted by the applicable licenses. For artifacts that cannot be redistributed, we will provide scripts and instructions to reconstruct them.

REFERENCES

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, et al. Phi-4-Mini technical report: Compact yet powerful multimodal language models via mixture-of-LoRAs. *arXiv preprint arXiv:2503.01743*, 2025.

Robin Algayres, Mohamed Salah Zaiem, Benoît Sagot, and Emmanuel Dupoux. Evaluating the reliability of acoustic speech embeddings. In *Proceedings of Interspeech*, pp. 4621–4625, 2020.

Robin Algayres, Yossi Adi, Tu Anh Nguyen, Jade Copet, Gabriel Synnaeve, Benoit Sagot, and Emmanuel Dupoux. Generative spoken language model based on continuous word-sized audio tokens. In *Proceedings of EMNLP*, pp. 3008–3028, 2023.

Robin Algayres, Charles-Éric Saint-James, Mahi Luthra, Jiayi Shen, Youssef Benchekroun, Dongyan Lin, Rashel Moritz, Juan Pino, and Emmanuel Dupoux. LongTail-Swap: benchmarking language models' abilities on rare words. In *Findings of EMNLP*, pp. 11231–11251, 2025.

Keyu An, Qian Chen, Chong Deng, Zhihao Du, Changfeng Gao, Zhifu Gao, Yue Gu, Ting He, Hangrui Hu, Kai Hu, et al. FunAudioLLM: Voice understanding and generation foundation models for natural interaction between humans and LLMs. *arXiv preprint arXiv:2407.04051*, 2024.

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *Proceedings of ECCV*, pp. 456–473, 2022.

Alan Baade, Puyuan Peng, and David Harwath. SyllableLM: Learning coarse semantic units for speech language models. In *Proceedings of ICLR*, volume 2025, pp. 15577–15598, 2025.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proceedings of NeurIPS*, pp. 12449–12460, 2020.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *Proceedings of ICML*, pp. 1298–1312, 2022.

Alexei Baevski, Arun Babu, Wei-Ning Hsu, and Michael Auli. Efficient self-supervised learning with contextualized target representations for vision, speech and language. In *Proceedings of ICML*, pp. 1416–1429, 2023.

Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al. Seamless: Multilingual expressive and streaming speech translation. *arXiv preprint arXiv:2312.05187*, 2023.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. AudioLM: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2523–2533, 2023.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, et al. Accounting for variance in machine learning benchmarks. *Proceedings of MLSys*, pp. 747–769, 2021.

Hervé Bredin. pyannote.audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. In *Proceedings of Interspeech*, pp. 1983–1987, 2023.

Michael A Carlin, Samuel Thomas, Aren Jansen, and Hynek Hermansky. Rapid evaluation of speech representations for spoken term discovery. In *Proceedings of Interspeech*, pp. 821–824, 2011.

Heng-Jui Chang, Alexander H. Liu, and James Glass. Self-supervised fine-tuning for improved content representations by speaker-invariant clustering. In *Proceedings of Interspeech*, pp. 2983–2987, 2023.

Heng-Jui Chang, Saurabhchand Bhati, James Glass, and Alexander H Liu. USAD: Universal speech and audio representation via distillation. In *Proceedings of ASRU*, pp. 1–8, 2025a.

Heng-Jui Chang, Hongyu Gong, Changhan Wang, James Glass, and Yu-An Chung. DC-Spin: A speaker-invariant speech tokenizer for spoken language models. In *Proceedings of Interspeech*, pp. 5723–5727, 2025b.

Li-Wei Chen, Takuya Higuchi, Zakaria Aldeneh, Ahmed Hussen Abdelaziz, and Alexander Rudnicky. A variational framework for improving naturalness in generative spoken language models. In *Proceedings of ICML*, pp. 8226–8244, 2025.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

William Chen, Wangyou Zhang, Yifan Peng, Xinjian Li, Jinchuan Tian, Jiatong Shi, Xuankai Chang, Soumi Maiti, Karen Livescu, and Shinji Watanabe. Towards robust speech representation learning for thousands of languages. In *Proceedings of EMNLP*, pp. 10205–10224, 2024.

Cheol Jun Cho, Nicholas Lee, Akshat Gupta, Dhruv Agarwal, Ethan Chen, Alan Black, and Gopala Anumanchipalli. Sylber: Syllabic embedding representation of speech from raw audio. In *Proceedings of ICLR*, pp. 22012–22038, 2025.

Ju-Chieh Chou, Jiawei Zhou, and Karen Livescu. Flow-SLM: Joint learning of linguistic and acoustic information for spoken language modeling. In *Proceedings of ASRU*, pp. 1–7, 2025.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. In *Proceedings of NeurIPS*, pp. 47704–47720, 2023.

Santiago Cuervo and Ricard Marxer. Scaling properties of speech language models. In *Proceedings of EMNLP*, pp. 351–361, 2024.

Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, et al. MiniCPM-o 4.5: Towards real-time full-duplex omni-modal interaction. *arXiv preprint arXiv:2604.27393*, 2026.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Transactions on Machine Learning Research*, 2023.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*, 2024.

Amala Sanjay Deshmukh, Kateryna Chumachenko, Tuomas Rintamaki, Matthieu Le, Tyler Poon, Danial Mohseni Taheri, Ilia Karmanov, Guilin Liu, Jarno Seppanen, Arushi Goel, et al. Nemotron 3 Nano Omni: efficient and open multimodal intelligence. *arXiv preprint arXiv:2604.24954*, 2026.

Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, Fan Yu, Huadai Liu, Zhengyan Sheng, Yue Gu, Chong Deng, Wen Wang, Shiliang Zhang, Zhijie Yan, and Jingren Zhou. CosyVoice 2: Scalable streaming speech synthesis with large language models. *arXiv preprint arXiv:2412.10117*, 2024.

Nicolas Dufour, Alexei A Efros, and Patrick Pérez. The FID lottery: Quantifying hidden randomness in generative-model evaluation. *arXiv preprint arXiv:2606.20536*, 2026.

Emmanuel Dupoux, Yann LeCun, and Jitendra Malik. Why AI systems don't learn and what to do about it: Lessons on autonomous learning from cognitive science. *arXiv preprint arXiv:2603.15381*, 2026.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic BERT sentence embedding. In *Proceedings of ACL*, pp. 878–891, 2022.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Sreyan Ghosh, Arushi Goel, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang-gil Lee, Chao-Han Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, et al. Audio Flamingo 3: Advancing audio intelligence with fully open large audio language models. In *Proceedings of NeurIPS*, pp. 41819–41886, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Proceedings of NeurIPS*, pp. 21271–21284, 2020.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. In *Proceedings of Interspeech*, pp. 5036–5040, 2020.

Haohan Guo, Fenglong Xie, Kun Xie, Dongchao Yang, Dake Guo, Xixin Wu, and Helen Meng. SoCodec: A semantic-ordered multi-stream speech codec for efficient language model based text-to-speech synthesis. In *Proceedings of SLT*, pp. 645–651, 2024.

Michael Hassid, Tal Remez, Tu Anh Nguyen, Itai Gat, Alexis Conneau, Felix Kreuk, Jade Copet, Alexandre Defossez, Gabriel Synnaeve, Emmanuel Dupoux, et al. Textually pretrained speech language models. In *Proceedings of NeurIPS*, pp. 63483–63501, 2023.

Haorui He, Zengqiang Shang, Chaoren Wang, Xuyuan Li, Yicheng Gu, Hua Hua, Liwei Liu, Chen Yang, Jiaqi Li, Peiyang Shi, et al. Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation. In *Proceedings of SLT*, pp. 885–890. IEEE, 2024.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of AAAI*, 2018.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

He Huang, Taejin Park, Kunal Dhawan, Ivan Medennikov, Krishna C Puvvada, Nithin Rao Koluguri, Weiqing Wang, Jagadeesh Balam, and Boris Ginsburg. NEST: Self-supervised fast Conformer as all-purpose seasoning to speech processing tasks. In *Proceedings of ICASSP*, pp. 1–5, 2025.

Jeff Hwang, Moto Hira, Caroline Chen, Xiaohui Zhang, Zhaoheng Ni, Guangzhi Sun, Pingchuan Ma, Ruizhe Huang, Vineel Pratap, Yuekai Zhang, et al. TorchAudio 2.1: Advancing speech recognition, self-supervised learning, and audio processing components for PyTorch. In *Proceedings of ASRU*, pp. 1–9, 2023.

Keith Ito and Linda Johnson. The LJ speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

Nat Jeffries, Evan King, Manjunath Kudlur, Guy Nicholson, James Wang, and Pete Warden. Moonshine: Speech recognition for live transcription and voice commands. *arXiv preprint arXiv:2410.15608*, 2024.

Jaap Jumelet, Leonie Weissweiler, Joakim Nivre, and Arianna Bisazza. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. *Transactions of the Association for Computational Linguistics*, 14:193–216, 2026.

Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. Libri-Light: A benchmark for ASR with limited or no supervision. In *Proceedings of ICASSP*, pp. 7669–7673, 2020.

Shunsuke Kando, Yusuke Miyao, and Shinnosuke Takamichi. Exploring the effect of segmentation and vocabulary size on speech tokenization for speech language models. In *Proceedings of Interspeech*, pp. 5728–5732, 2025.

Wei Kang, Xiaoyu Yang, Zengwei Yao, Fangjun Kuang, Yifan Yang, Liyong Guo, Long Lin, and Daniel Povey. Libriheavy: A 50,000 hours ASR corpus with punctuation casing and context. In *Proceedings of ICASSP*, pp. 10991–10995, 2024.

Emmanuel Keuleers and Marc Brysbaert. Wuggy: A multilingual pseudoword generator. *Behavior research methods*, 42(3):627–633, 2010.

Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu Anh Nguyen, Morgane Riviere, Abdelrahman Mohamed, Emmanuel Dupoux, et al. Text-free prosody-aware generative spoken language modeling. In *Proceedings of ACL*, pp. 8666–8681, 2022.

Sameer Khurana, Dominik Klement, Antoine Laurent, Dominik Boboš, Juraj Novosad, Peter Gazdik, Ellen Zhang, Zili Huang, Amir Hussein, Ricard Marxer, Yoshiki Masuyama, Ryo Aihara, Chiori Hori, François G. Germain, Gordon Wichern, and Jonathan Le Roux. Factorized RVQ-GAN for disentangled speech tokenization. In *Proceedings of Interspeech*, pp. 3514–3518, 2025.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of EMNLP*, pp. 4334–4353, 2024.

KimiTeam. Kimi-Audio technical report. *arXiv preprint arXiv:2504.18425*, 2025.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Proceedings of NeurIPS*, pp. 17022–17033, 2020.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of ACL*, pp. 66–75, 2018.

Tom Labiausse, Laurent Mazaré, Edouard Grave, Alexandre Défossez, and Neil Zeghidour. High-fidelity simultaneous speech-to-speech translation. In *Proceedings of ICML*, pp. 32116–32129, 2025.

Kushal Lakhotia, Eugene Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Abdelrahman Mohamed, et al. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354, 2021.

Marvin Lavechin, Marianne Métais, Hadrien Titeux, Alodie Boissonnet, Jade Copet, Morgane Rivière, Elika Bergelson, Alejandrina Cristia, Emmanuel Dupoux, and Hervé Bredin. Brouhaha: Multi-task training for voice activity detection, speech-to-noise ratio, and C50 room acoustics estimation. In *Proceedings of ASRU*, pp. 1–7, 2023.

Yann LeCun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of CVPR*, pp. 11513–11522, 2022.

Guan-Ting Lin, Prashanth Gurunath Shivakumar, Aditya Gourav, Yile Gu, Ankur Gandhe, Hung-yi Lee, and Ivan Bulyko. Align-SLM: Textless spoken language models with reinforcement learning from AI feedback. In *Proceedings of ACL*, pp. 20395–20411, 2025.

Alexander H Liu, Heng-Jui Chang, Michael Auli, Wei-Ning Hsu, and James Glass. DinoSR: Self-distillation and online clustering for self-supervised speech representation learning. In *Proceedings of NeurIPS*, pp. 58346–58362, 2023.

Alexander H Liu, Andy Ehrenberg, Andy Lo, Clément Denoix, Corentin Barreau, Guillaume Lample, Jean-Malo Delignon, Khyathi Raghavi Chandu, Patrick von Platen, Pavankumar Reddy Muddireddy, et al. Voxtral. *arXiv preprint arXiv:2507.13264*, 2025.

Haohe Liu, Xuenan Xu, Yi Yuan, Mengyue Wu, Wenwu Wang, and Mark D Plumbley. SemantiCodec: An ultra low bitrate semantic audio codec for general sound. *IEEE Journal of Selected Topics in Signal Processing*, 18(8):1448–1461, 2024.

Yen-Ju Lu, Yashesh Gaur, Wei Zhou, Benjamin Muller, Jesus Villalba, Najim Dehak, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Srini Iyer, and Duc Le. Latent speech-text Transformer. In *Proceedings of ICLR*, pp. 60387–60404, 2026.

Tianze Luo, Zixin Wang, Kaizhi Qian, Yang Zhang, and Chuang Gan. BSLM: A bi-level speech-language model for the joint modeling of discrete and continuous tokens. In *AAAI 2026 Workshop on Audio-Centric AI*, 2026.

Gallil Maimon, Avishai Elmakies, and Yossi Adi. Slamming: Training a speech language model on one GPU in a day. In *Findings of ACL*, pp. 12201–12216, 2025a.

Gallil Maimon, Michael Hassid, Amit Roth, and Yossi Adi. Scaling analysis of interleaved speech-text language models. In *Proceedings of COLM*, 2025b.

Soumi Maiti, Yifan Peng, Shukjae Choi, Jee-weon Jung, Xuankai Chang, and Shinji Watanabe. VoxtLM: Unified decoder-only models for consolidating speech recognition, synthesis and speech, text continuation tasks. In *Proceedings of ICASSP*, pp. 13326–13330, 2024.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of NAACL-HLT*, pp. 839–849, 2016.

Tu Anh Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Evgeny Kharitonov, Alexei Baevski, Ewan Dunbar, and Emmanuel Dupoux. The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling. *arXiv preprint arXiv:2011.11588*, 2020.

Tu Anh Nguyen, Benoit Sagot, and Emmanuel Dupoux. Are discrete units necessary for spoken language modeling? *IEEE Journal of Selected Topics in Signal Processing*, 16:1415–1423, 2022.

Tu Anh Nguyen, Eugene Kharitonov, Jade Copet, Yossi Adi, Wei-Ning Hsu, Ali Elkahky, Paden Tomasello, Robin Algayres, Benoit Sagot, Abdelrahman Mohamed, et al. Generative spoken dialogue language modeling. *Transactions of the Association for Computational Linguistics*, 11: 250–266, 2023.

Tu-Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R Costa-jussà, Maha Elbayad, Sravya Popuri, Christophe Ropers, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, et al. SpiRit-LM: Interleaved spoken and written language model. *Transactions of the Association for Computational Linguistics*, 13:30–52, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an ASR corpus based on public domain audio books. In *Proceedings of ICASSP*, pp. 5206–5210, 2015.

Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. Layer-wise analysis of a self-supervised speech representation model. In *Proceedings of ASRU*, pp. 914–921, 2021.

Ankita Pasad, Chung-Ming Chien, Shane Settle, and Karen Livescu. What do self-supervised speech models know about words? *Transactions of the Association for Computational Linguistics*, 12: 372–391, 2024.

Yifan Peng, Jinchuan Tian, William Chen, Siddhant Arora, Brian Yan, Yui Sudo, Muhammad Shakeel, Kwanghee Choi, Jiatong Shi, Xuankai Chang, Jee weon Jung, and Shinji Watanabe. OWSM v3.1: Better and faster open Whisper-style speech models based on E-Branchformer. In *Interspeech*, pp. 352–356, 2024.

Maxime Poli, Emmanuel Chemla, and Emmanuel Dupoux. fastabx: A library for efficient computation of ABX discriminability. *arXiv preprint arXiv:2505.02692*, 2025a.

Maxime Poli, Mahi Luthra, Youssef Benchekroun, Yosuke Higuchi, Martin Gleize, Jiayi Shen, Robin Algayres, Yu-An Chung, Mido Assran, Juan Pino, and Emmanuel Dupoux. SpidR: Learning fast and stable linguistic units for spoken language models without supervision. *Transactions on Machine Learning Research*, 2025b.

Maxime Poli, Manel Khentout, Angelo Ortiz Tandazo, Ewan Dunbar, Emmanuel Chemla, and Emmanuel Dupoux. DiscoPhon: Benchmarking the unsupervised discovery of phoneme inventories with discrete speech units. In *Proceedings of Interspeech*, 2026.

Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharitonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. Speech resynthesis from discrete disentangled self-supervised representations. In *Proceedings of Interspeech*, pp. 3615–3619, 2021.

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, et al. Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52, 2024.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In *Proceedings of ACL*, pp. 1882–1892, 2020.

Krishna C. Puvvada, Piotr Żelasko, He Huang, Oleksii Hrinchuk, Nithin Rao Koluguri, Kunal Dhawan, Somshubra Majumdar, Elena Rastorgueva, Zhehuai Chen, Vitaly Lavrukhin, Jagadeesh Balam, and Boris Ginsburg. Less is more: Accurate speech recognition & translation without web-scale data. In *Proceedings of Interspeech*, pp. 3964–3968, 2024.

Kaizhi Qian, Yang Zhang, Heting Gao, Junrui Ni, Cheng-I Lai, David Cox, Mark Hasegawa-Johnson, and Shiyu Chang. Contentvec: An improved self-supervised speech representation by disentangling speakers. In *Proceedings of ICML*, pp. 18003–18017, 2022.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *Proceedings of ICML*, pp. 28492–28518, 2023.

Jason Ramapuram, Eeshan Gunesh Dhekane, Amitis Shidani, Dan Busbridge, Bogdan Mazoure, Zijin Gu, Russ Webb, Tatiana Likhomanenko, and Navdeep Jaitly. Scaling properties of continuous diffusion spoken language models. *arXiv preprint arXiv:2604.24416*, 2026.

Dima Rekesh, Nithin Rao Koluguri, Samuel Kriman, Somshubra Majumdar, Vahid Noroozi, He Huang, Oleksii Hrinchuk, Krishna Puvvada, Ankur Kumar, Jagadeesh Balam, et al. Fast Conformer with linearly scalable attention for efficient speech recognition. In *Proceedings of ASRU*, pp. 1–8, 2023.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. FastSpeech 2: Fast and high-quality end-to-end text to speech. In *Proceedings of ICLR*, 2021.

Rajarshi Roy, Jonathan Raiman, Sang-gil Lee, Teodor-Dumitru Ene, Robert Kirby, Sungwon Kim, Jaehyeon Kim, and Bryan Catanzaro. PersonaPlex: Voice and role control for full duplex conversational speech models. In *Proceedings of ICASSP*, pp. 16137–16141, 2026.

Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. UTMOS: UTokyo-SaruLab system for VoiceMOS challenge 2022. In *Proceedings of Interspeech*, pp. 4521–4525, 2022.

Thomas Schatz. *ABX-discriminability measures and applications*. PhD thesis, Université Paris 6 (UPMC), 2016.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pp. 1715–1725, 2016.

Gaofei Shen, Afra Alishahi, Arianna Bisazza, and Grzegorz Chrupała. Wave to syntax: Probing spoken language models for syntax. In *Proceedings of Interspeech*, pp. 1259–1263, 2023.

Jiatong Shi, Hirofumi Inaguma, Xutai Ma, Ilia Kulikov, and Anna Sun. Multi-resolution HuBERT: Multi-resolution speech self-supervised learning with masked unit prediction. In *Proceedings of ICLR*, volume 2024, pp. 25421–25457, 2024a.

Jiatong Shi, Xutai Ma, Hirofumi Inaguma, Anna Sun, and Shinji Watanabe. MMM: Multi-layer multi-residual multi-stream discrete speech representation from self-supervised learning model. In *Proceedings of Interspeech*, pp. 2569–2573, 2024b.

Amitay Sicherman and Yossi Adi. Analysing discrete self supervised speech representation for spoken language modeling. In *Proceedings of ICASSP*, pp. 1–5, 2023.

Issa Sugiura, Shuhei Kurita, Yusuke Oda, and Ryuichiro Higashinaka. Llama-Mimi: Speech language models with interleaved semantic and acoustic tokens. *arXiv preprint arXiv:2509.14882*, 2025.

Liang-Hsuan Tseng, Yi-Chang Chen, Kuan Lee, Da-Shan Shiu, and Hung-yi Lee. TASTE: Text-aligned speech tokenization and embedding for spoken language modeling. In *Proceedings of ICLR*, volume 2026, pp. 125293–125314, 2026.

Arnon Turetzky and Yossi Adi. LAST: Language model aware speech tokenization. *arXiv preprint arXiv:2409.03701*, 2024.

Nicol Visser and Herman Kamper. Spoken language modeling with duration-penalized self-supervised units. In *Proceedings of Interspeech*, pp. 1968–1972, 2025.

Nicol Visser, Simon Malan, Danel Slabbert, and Herman Kamper. ZeroSyl: Simple zero-resource syllable tokenization for spoken language modeling. In *Proceedings of Interspeech*, 2026.

Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proceedings of ACL-IJCNLP*, pp. 993–1003, 2021a.

Xinyi Wang, Sebastian Ruder, and Graham Neubig. Multi-view subword regularization. In *Proceedings of NAACL-HLT*, pp. 473–482, 2021b.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

Dan Wells, Hao Tang, and Korin Richmond. Phonetic analysis of self-supervised representations of English speech. In *Proceedings of Interspeech*, pp. 3583–3587, 2022.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, et al. Qwen2.5-Omni technical report. *arXiv preprint arXiv:2503.20215*, 2025a.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, et al. Qwen3-Omni technical report. *arXiv preprint arXiv:2509.17765*, 2025b.

Dongchao Yang, Jinchuan Tian, Xu Tan, Rongjie Huang, Songxiang Liu, Xuankai Chang, Jiatong Shi, Sheng Zhao, Jiang Bian, Zhou Zhao, et al. UniAudio: An audio foundation model toward universal audio generation. *arXiv preprint arXiv:2310.00704*, 2023.

Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y Lin, Andy T Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, et al. SUPERB: Speech processing universal performance benchmark. In *Proceedings of Interspeech*, pp. 1194–1198, 2021.

Zhen Ye, Peiwen Sun, Jiahe Lei, Hongzhan Lin, Xu Tan, Zheqi Dai, Qiuqiang Kong, Jianyi Chen, Jiahao Pan, Qifeng Liu, et al. Codec does matter: Exploring the semantic shortcoming of codec for audio language model. In *Proceedings of AAAI*, pp. 25697–25705, 2025.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

Dong Zhang, Gang Wang, Jinlong Xue, Kai Fang, Liang Zhao, Rui Ma, Shuhuai Ren, Shuo Liu, Tao Guo, Weiji Zhuang, et al. MiMo-Audio: Audio language models are few-shot learners. *arXiv preprint arXiv:2512.23808*, 2025.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. OPT: Open pre-trained Transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Xin Zhang, Dong Zhang, Shimin Li, Yaqian Zhou, and Xipeng Qiu. SpeechTokenizer: Unified speech tokenizer for speech language models. In *Proceedings of ICLR*, volume 2024, pp. 31798–31818, 2024.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. Image BERT pre-training with online tokenizer. In *Proceedings of ICLR*, 2022.

Yongxin Zhu, Dan Su, Liqiang He, Linli Xu, and Dong Yu. Generative pre-trained speech language model with efficient hierarchical Transformer. In *Proceedings of ACL*, pp. 1764–1775, 2024.

A Additional Related Work

Speech tokenization for spoken language modeling. A central distinction among speech tokenizers is the type of information their units are optimized to preserve. Neural codecs, such as SoundStream (Zeghidour et al., 2021) and EnCodec (Défossez et al., 2023), learn quantized representations optimized for waveform reconstruction. Although this objective preserves the acoustic detail required for high-fidelity resynthesis, the resulting units may retain variation irrelevant to linguistic prediction. Thus, these units may not be linguistically invariant or readily predictable in language modeling (Chang et al., 2025b). SLMs therefore commonly rely on units derived from self-supervised speech encoders.[1] A canonical example is HuBERT (Hsu et al., 2021), which is trained with a self-supervised masked-prediction objective over discrete targets and learns hidden representations that capture phonetic and linguistic structure (Wells et al., 2022; Sicherman & Adi, 2023). These representations are most often discretized by applying post-hoc k-means clustering to a selected encoder layer (Lakhotia et al., 2021). DinoSR (Liu et al., 2023) and SpidR (Poli et al., 2025b) instead learn discrete targets online during masked-prediction-based pretraining. Nevertheless, online codes do not uniformly outperform post-hoc k-means units; for SpidR in particular, post-hoc units perform better on several downstream SLM metrics.

SpidR 2.0 advances this line of work by learning differentiable unit spaces at multiple encoder layers through shared student-teacher prediction heads (Zhou et al., 2022; Oquab et al., 2024) and entropy regularization (Assran et al., 2022). These techniques originate in self-supervised vision models and remain underexplored for discrete speech tokenization. Adapting them to speech enables SpidR 2.0 to learn stable, linguistically informative unit spaces at multiple levels of abstraction during pretraining, eliminating post-hoc k-means quantization and providing complementary unit streams for downstream spoken language modeling.

Complementary approaches use continuous speech representations directly for spoken language modeling (Algayres et al., 2023; Ramapuram et al., 2026), or jointly model discrete semantic units and continuous acoustic or paralinguistic representations (Chen et al., 2025; Chou et al., 2025). Given our emphasis on linguistic competence over reconstruction fidelity, we focus on discrete units as a representational bottleneck that can suppress linguistically irrelevant acoustic variation (Nguyen et al., 2022).

Multi-level speech representations. Prior work has represented speech using multiple discrete levels, most often to separate high-level content from the acoustic detail required for waveform reconstruction. AudioLM (Borsos et al., 2023) cascades semantic, coarse acoustic, and fine acoustic tokens. Related codec-based approaches realize this content-fidelity trade-off in different ways: SpeechTokenizer (Zhang et al., 2024) distributes semantic and acoustic information hierarchically

[1] Methods that use text supervision or pretrained LLMs to learn semantically aligned speech tokens (Turetzky & Adi, 2024; Zhang et al., 2025; Tseng et al., 2026) are complementary to our approach; here, we focus on textless GSLM.

across residual vector-quantization (RVQ) layers; SemantiCodec (Liu et al., 2024) uses separate semantic and acoustic encoders; and X-Codec (Ye et al., 2025) injects pretrained semantic features into an acoustic codec and introduces a semantic reconstruction objective. HAC (Khurana et al., 2025) further factorizes a codec bottleneck into acoustic, phonetic, and lexical token sets using distillation from HuBERT and LaBSE (Feng et al., 2022). Despite their architectural differences, these methods organize speech representations around the trade-off between content modeling and reconstruction fidelity.

A complementary line of work exploits representations from different encoder depths. MR-HuBERT (Shi et al., 2024a) adapts HuBERT's training procedure to learn units at multiple temporal resolutions from such representations. More closely related to our representation scheme, MMM (Shi et al., 2024b) constructs multiple discrete streams by applying iterative residual k-means quantization to several layers of a self-supervised speech encoder, and evaluates them on ASR, speech resynthesis, and text-to-speech (TTS). GSLM2 likewise exploits representations across encoder depth, but learns the corresponding unit spaces jointly during self-supervised pretraining through layer-wise student-teacher prediction. It then treats the resulting units as parallel, complementary levels of abstraction for spoken language modeling rather than as a hierarchy of residual refinements for waveform reconstruction.

Multi-stream sequence modeling. Multi-stream architectures differ not only in what their streams represent, but also in how they factorize the joint distribution over those streams. In textless GSLM, pGSLM (Kharitonov et al., 2022) jointly models phonetic units, duration, and pitch, whereas dGSLM (Nguyen et al., 2023) jointly models the two speaker channels of spoken dialogue. Codec-based LMs instead operate on multiple quantizer streams. MusicGen (Copet et al., 2023) and SoCodec (Guo et al., 2024) use delay patterns to express generation across time and codebook depth as a parallel autoregressive sequence. Hierarchical audio models such as UniAudio (Yang et al., 2023) and GPST (Zhu et al., 2024) adapt the RQ-Transformer principle (Lee et al., 2022), using a large temporal or global Transformer to model dependencies across frames and a smaller local Transformer to autoregressively model codec tokens within each frame. Moshi (Défossez et al., 2024) employs this factorization for spoken dialogue generation: its temporal Transformer conditions on all codec streams from preceding frames, while its depth Transformer autoregressively predicts the semantically distilled first codebook and the residual acoustic codebooks at the current frame. Hibiki (Labiausse et al., 2025) extends this approach to simultaneous speech-to-speech translation.

Our multi-stream SLM repurposes this temporal-depth factorization for time-aligned units extracted from different layers of a speech encoder, with each stream representing a distinct level of abstraction rather than a level of residual codec quantization. The shared temporal backbone integrates the histories of all streams, while the generative head autoregressively factorizes the joint next-step distribution over the layer-specific units, thereby capturing dependencies across levels of abstraction. In parallel, stream-specific heads directly predict the next unit in each stream, providing explicit objectives for modeling its temporal transition dynamics.

B Additional Experimental Details

This section details the experimental setup for both our main experiments (Section 3) and additional experiments (Appendix C). The material is organized as follows:

- Appendix B.1: Datasets and preprocessing
- Appendix B.2: Model architecture and design choices
- Appendix B.3: Training configurations
- Appendix B.4: Evaluation metrics, protocols, and configurations

B.1 Datasets and Preprocessing

Table 6 summarizes the training data for the four GSLM components—the speech encoder, SLM, duration predictor, and unit vocoder—as well as the corresponding text LMs. Unless otherwise noted, we use the released audio without applying corpus-specific filtering.

Table 6: Training corpora, languages, and data scales for GSLM components and text LMs.

Corpus	Language(s)	Component	Training data
LibriSpeech	English	Speech encoder	960 h
Libri-Light	English	SLM Duration predictor	600 h / 6K h / 60K h 600 h
Libriheavy	English	Text LM	Transcripts aligned with Libri-Light
LJ Speech	English	Unit vocoder	24 h
VoxPopuli	En, Fr, De	Speech encoder SLM Text LM	6K h per language 6K h per language ASR transcripts of the same 6K h
Conversational Speech	English	SLM Text LM	600 h / 6K h / 60K h / 600K h / 6M h ASR transcripts of the same 600 h to 6M h subsets

LibriSpeech. The speech encoders, including HuBERT, DinoSR, SpidR and SpidR 2.0, are trained on the full 960-hour LibriSpeech training set (Panayotov et al., 2015), comprising `train-clean-100`, `train-clean-360`, and `train-other-500` subsets of English audiobook speech. Pretraining is fully self-supervised and uses only the speech waveforms, without access to the corresponding transcriptions. We use `dev-clean` and `dev-other` for validation during encoder development. All encoders operate on raw waveforms.

Libri-Light. The SLMs are trained on the `unlab-600`, `unlab-6k`, and `unlab-60k` subsets of Libri-Light (Kahn et al., 2020), which contain approximately 600, 6K, and 60K hours of untranscribed English audiobook speech, respectively. The subsets are nested, such that each smaller training set is contained within the next larger one. We use the official Libri-Light segmentation pipeline[2] to divide the raw recordings into segments of approximately 60 seconds.

To construct the SLM training data, we pass the segmented Libri-Light waveforms through the corresponding pretrained speech encoder and extract sequences of discrete speech units. For tokenizers requiring post-hoc quantization, i.e., HuBERT and SpidR, we fit a k-means model to representations extracted from the `train-clean-100` subset of LibriSpeech, where the number of clusters is set to 256, unless otherwise noted. The DinoSR and SpidR 2.0 units are obtained directly from their learned prediction heads. To resynthesize speech from deduplicated unit sequences, we train the duration predictor on the unit sequences extracted from the `unlab-600` subset.

Libriheavy. For the Libri-Light experiments, the text LMs are trained on Libriheavy (Kang et al., 2024), which provides transcripts aligned with the Libri-Light recordings. At each scale, we use the transcripts corresponding to the respective Libri-Light subset. The text LM and SLM therefore use the same underlying audiobook data and approximately matched amounts of linguistic content, enabling a controlled comparison of their scaling behavior.

We use the official long-form resegmentation of Libriheavy,[3] in which consecutive transcripts are grouped into multi-sentence segments to preserve greater cross-sentence continuity. In preliminary experiments, we found this longer context to be important for training effective text LMs, particularly for sentence-level evaluations such as tSC and sSC. The released transcripts are used without additional normalization or filtering. For each training-data scale, we train a separate BPE tokenizer (Sennrich et al., 2016) with a vocabulary of 4,096 tokens on the corresponding training transcripts.

LJ Speech. The unit HiFi-GAN vocoder (Polyak et al., 2021) is trained on the 24-hour LJ Speech corpus (Ito & Johnson, 2017). We hold out 250 utterances for validation and testing, leaving approximately 23 hours for training.

[2] https://github.com/facebookresearch/libri-light

[3] https://huggingface.co/datasets/pkufool/libriheavy_long

VoxPopuli. For the multilingual experiments, we use the English, French, and German portions of VoxPopuli (Wang et al., 2021a), which contain approximately 24K, 23K, and 23K hours of speech respectively. VoxPopuli consists of European Parliament recordings and exhibits substantially more heterogeneous acoustic conditions than Libri-Light, including greater variation in speakers, recording environments, channels, and background noise.

For each language, we construct training examples from the original unsegmented VoxPopuli recordings[4]. We first apply voice activity detection using pyannote (Bredin, 2023)[5] to identify contiguous speech regions. We then perform language identification (LID) on each region using both Whisper large-v3 (Radford et al., 2023)[6] and MMS-LID (Pratap et al., 2024)[7]. We retain speech regions of at least 0.5 seconds and merge adjacent regions separated by less than 5 seconds of silence, provided that the merged interval does not contain a region rejected by LID. Any resulting example longer than 60 seconds is recursively split at the longest interior silence until the maximum-duration constraint is satisfied. Intervening nonspeech intervals are removed, resulting in speech-only examples between 0.5 and 60 seconds in duration. After filtering and segmentation, the English, French, and German collections contain approximately 16K, 23K, and 23K hours of speech, with mean example durations of 21.6, 26.2, and 27.1 seconds, respectively.

For each language, we independently sample a 6K-hour subset and use it to train both the speech encoder and the SLM. For the SpidR-based SLM, we fit a k-means model to representations extracted from a randomly sampled 100-hour subset of the corresponding training data, with the number of clusters set to 256. To construct the text LM training data, we transcribe the same 6K-hour speech subsets using Whisper large-v3. For each language, we train a separate BPE tokenizer with a vocabulary of 4,096 tokens on the resulting ASR transcriptions and use the tokenized sequences to train the corresponding text LM.

Conversational Speech. To study scaling beyond Libri-Light, we use English conversational speech sourced from the internet, totaling approximately 6M hours. We combine the public Emilia corpus (He et al., 2024) with off-the-shelf conversational audio licensed from two commercial data providers. We construct subsets of 600, 6K, 60K, and 600K hours by randomly sampling segments from the 6M-hour superset, keeping the proportion of the three sources constant across subsets.

We use the English portion of Emilia as released, together with its published ASR transcripts. The Emilia-Pipe preprocessing pipeline has already filtered the data for quality and segmented them into single-speaker utterances lasting 3-30 seconds.

For the licensed audio, we first apply quality-based filtering to the source recordings. We estimate signal-to-noise ratio (SNR) using brouhaha (Lavechin et al., 2023), perform LID with Whisper large-v3, and apply audio event detection classification to discard recordings dominated by non-speech content. We then construct training examples from the retained recordings. We apply voice activity detection to identify contiguous speech regions, retaining regions of at least 0.5 seconds and merging adjacent regions separated by less than 5 seconds of silence. Any resulting example longer than 90 seconds is recursively split at the longest interior silence until the maximum-duration constraint is satisfied. Intervening nonspeech intervals are removed, yielding speech-only examples between 0.5 and 90 seconds in duration. To build the corresponding text corpus, we transcribe these examples with faster-whisper,[8] using the large-v3 checkpoint.

SLM training data is constructed as per Libri-Light: we pass the segmented waveforms through the pretrained speech encoder and extract sequences of discrete speech units. For text LM training, we train a separate BPE tokenizer with a vocabulary of 4,096 tokens on the ASR transcriptions.

B.2 Model Architecture and Design Choices

Speech encoder. All encoders used in our experiments, including HuBERT, DinoSR, SpidR and SpidR 2.0, use the same Transformer backbone architecture, based on Base wav2vec 2.0 (Baevski

[4] https://github.com/facebookresearch/voxpopuli
[5] https://huggingface.co/pyannote/segmentation-3.0
[6] https://huggingface.co/openai/whisper-large-v3
[7] https://huggingface.co/facebook/mms-lid-1024
[8] https://github.com/SYSTRAN/faster-whisper

Table 7: Transformer backbone configurations for the speech encoders and spoken/text LMs. Scale labels denote nominal backbone sizes, and Table 8 reports the exact total parameter counts for the LMs.

	Encoder	Language model				
Hyperparameter	90M	90M	300M	1.2B	2.5B	6.5B
Transformer layers	12	12	24	24	32	32
Hidden dimension	768	768	1,024	2,048	2,560	4,096
FFN dimension	3,072	3,072	4,096	8,192	10,240	16,384
Attention heads	12	12	16	32	32	32

Table 8: Total parameter counts for the LMs in the main comparison (Table 2 and Figure 2). Each count includes the Transformer backbone and its associated input and output modules.

	LM backbone scale				
Model	90M	300M	1.2B	2.5B	6.5B
Text LM	92.9M	312.8M	1.23B	2.54B	6.49B
SpidR SLM	87.0M	304.9M	1.21B	2.52B	6.45B
SpidR 2.0 SLM	87.0M	304.9M	1.21B	2.52B	6.45B
GSLM2	96.3M	315.5M	1.23B	2.54B	6.48B

et al., 2020). The Transformer configuration is summarized in Table 7. A convolutional feature extractor downsamples the 16 kHz input waveform to a sequence of 50 Hz features, which are processed by the Transformer layers. DinoSR and SpidR associate each of the top eight teacher layers with an online codebook containing 256 codewords. SpidR 2.0 replaces the separate online codebooks with layer-specific heads shared between the student and teacher at the same eight layers, each producing a softmax distribution over 256 codes.

Language model. We instantiate the SLMs and text LMs at several parameter scales using decoder-only Transformer configurations based on OPT (Zhang et al., 2022), as summarized in Table 7. We retain the standard OPT architecture to facilitate controlled comparisons across model scales and with prior SLM work.[9] The models are implemented using the official SLM training and evaluation pipeline released with SpidR.[10]

For GSLM2, the temporal Transformer uses the same backbone configuration as the single-stream SLMs. The generative head is implemented as a six-layer depth Transformer with a hidden dimension of 256, four attention heads, and a feed-forward network (FFN) dimension of 1,024. The generative head contains approximately 6M parameters and uses the same configuration at every backbone scale. GSLM2 additionally introduces stream-specific input embeddings and output heads, whose parameter counts grow slightly with the hidden dimension of the temporal backbone. Table 8 reports the total parameter counts for the models compared in Table 2 and Figure 2.

Duration predictor. Because the SLMs operate on deduplicated unit sequences, we train a separate duration predictor to restore the number of 20 ms frames associated with each generated unit. Each input unit is mapped to a learned embedding and augmented with sinusoidal positional encoding. The resulting sequence is processed by a non-autoregressive Conformer (Gulati et al., 2020) followed by a two-layer classification head over 50 duration classes, corresponding to durations of 1-50 frames. For single-stream SLMs, the Conformer has six layers, a hidden dimension of 128, four attention heads, a convolution kernel size of 17, and an FFN dimension of 512. For GSLM2, we increase its capacity to eight layers, a hidden dimension of 192, and an FFN dimension of 768, while this larger configuration was ineffective for single-stream SLMs. The per-stream embeddings

[9] In preliminary experiments, replacing OPT components with more modern alternatives, such as rotary positional embeddings and RMS normalization, did not yield consistent improvements. We therefore use the basic OPT-style architecture in all reported experiments.

[10] https://github.com/facebookresearch/spoken-lm

Table 9: Training configurations for text LMs, single-stream SLMs, and GSLM2 across data and model scales. Parenthesized values correspond to the parenthesized model size.

Data	Model	#Params	Peak learning rate	Steps	Global batch (units/tokens)	Context length	#GPUs
600 h	Text LM	90M	10^{-2}	10k	81,920	2,048	8 DDP
	SLM	90M	10^{-2}	15k	655,360	2,048	8 DDP
	GSLM2	90M	10^{-2}	15k	1,310,720	4,096	32 DDP
6K h	Text LM	300M	5.0×10^{-3}	20k	491,520	2,048	8 DDP
	SLM	300M (90M)	5.0×10^{-3} (10^{-2})	25k	655,360	2,048	8 DDP
	GSLM2	300M (90M)	5.0×10^{-3} (10^{-2})	25k	1,310,720	4,096	32 DDP
60K h	Text LM	1.2B	10^{-3}	100k	491,520	2,048	16 DDP
	SLM	1.2B	10^{-3}	150k	655,360	2,048	16 DDP
	GSLM2	1.2B	10^{-3}	150k	1,310,720	4,096	64 DDP
600K h	Text LM	2.5B	8.0×10^{-4}	250k	688,128	2,048	32 FSDP
	SLM	2.5B	8.0×10^{-4}	350k	917,504	2,048	32 FSDP
	GSLM2	2.5B	8.0×10^{-4}	350k	1,835,008	4,096	64 FSDP
6M h	Text LM	6.5B	3.0×10^{-4}	600k	688,128	2,048	32 FSDP
	SLM	6.5B	3.0×10^{-4}	750k	917,504	2,048	32 FSDP
	GSLM2	6.5B	3.0×10^{-4}	750k	1,835,008	4,096	112 FSDP

are concatenated and linearly projected to the 192-dimensional Conformer input. We then predict a single duration for each cross-stream-deduplicated tuple and share it across all streams.

Unit vocoder. To convert SLM-generated unit sequences into audio, we use a HiFi-GAN unit vocoder (Kong et al., 2020; Polyak et al., 2021) that maps discrete units at 20 ms frame intervals to a 16 kHz waveform. For single-stream models, each unit is mapped to a learned 128-dimensional embedding, and the resulting sequence conditions the HiFi-GAN generator. The generator upsamples this sequence through a cascade of transposed convolutions with strides $(5, 4, 4, 2, 2)$, yielding a total upsampling factor of 320. This matches the 320 samples corresponding to each 20 ms frame at 16 kHz. Each transposed convolution is followed by a multi-receptive-field fusion module with kernel sizes $(3, 7, 11)$ and dilations $(1, 3, 5)$. For GSLM2, the vocoder is jointly conditioned on all streams: each stream has its own 64-dimensional unit embedding, and the per-stream embeddings are concatenated and linearly projected to the 128-dimensional generator input.

B.3 Training Configurations

Speech encoder. Unless otherwise specified, we pretrain SpidR and SpidR 2.0. Pretraining largely follows the protocols established for DinoSR (Liu et al., 2023) and SpidR (Poli et al., 2025b). We train each encoder on LibriSpeech for 400k steps using 16 NVIDIA H200 GPUs and a global batch size corresponding to 63 minutes of audio. Training takes approximately 18 hours. Optimization uses AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$ and a weight decay of 0.01. The learning rate follows a three-stage schedule consisting of a 12k-step warmup, a 188k-step constant phase, and a 200k-step decay. During warmup, the learning rate increases from 5×10^{-6} to 5×10^{-4} for SpidR, and from 2×10^{-6} to 2×10^{-4} for SpidR 2.0. We freeze the convolutional feature extractor and the teacher encoder after 200k steps. For SpidR 2.0, the conditional-entropy weight is fixed at $\mu = 0.1$, while the marginal-entropy weight is initialized at $\lambda = 0.1$ and reduced to 0.001 after 20k steps. This regularization setup is selected to maximize development-set performance while minimizing bitrate, as detailed in Appendix C.2.4. For encoders trained with this setup, we use the final checkpoint at step 400k for evaluation and unit extraction. We also use the official DinoSR[11] and SpidR[12] checkpoints, along with the pretrained HuBERT BASE model available through TorchAudio (Hwang et al., 2023). All three downloaded models are pretrained on LibriSpeech and use similar architectures.

Language model. We train SLMs and text LMs across multiple model scales and training-data sizes. Table 9 summarizes the key configurations for each setting, including model size, peak learning rate, total training steps, global batch size, context length, number of NVIDIA H200 GPUs, and

[11] https://github.com/Alexander-H-Liu/dinosr
[12] https://github.com/facebookresearch/spidr

Table 10: Training and inference compute across model scales. Training FLOPs include both forward and backward passes, whereas inference FLOPs correspond to a forward pass per second of audio. Inference FLOPs of text LMs assume 4.45 BPE tokens/s, measured on Libriheavy transcriptions.

		Compute (FLOPs)	
Scale	**Model**	Training	Inference / audio s
600 h / 90M	Text LM	5.26×10^{17}	9.52×10^{8}
	SpidR 2.0 SLM	6.13×10^{18}	3.22×10^{9}
	GSLM2	2.34×10^{19}	1.11×10^{10}
6K h / 300M	Text LM	2.10×10^{19}	3.17×10^{9}
	SpidR 2.0 SLM	3.47×10^{19}	1.09×10^{10}
	GSLM2	1.16×10^{20}	3.14×10^{10}
60K h / 1.2B	Text LM	3.88×10^{20}	1.17×10^{10}
	SpidR 2.0 SLM	7.72×10^{20}	4.06×10^{10}
	GSLM2	2.39×10^{21}	1.04×10^{11}
600K h / 2.5B	Text LM	2.78×10^{21}	2.40×10^{10}
	SpidR 2.0 SLM	5.17×10^{21}	8.32×10^{10}
	GSLM2	1.57×10^{22}	2.09×10^{11}
6M h / 6.5B	Text LM	1.67×10^{22}	5.99×10^{10}
	SpidR 2.0 SLM	2.77×10^{22}	2.08×10^{11}
	GSLM2	8.28×10^{22}	5.08×10^{11}

parallelization strategy. In the table, SLM denotes a single-stream model trained on units extracted by HuBERT, DinoSR, SpidR, or SpidR 2.0. Table 10 reports the training and inference compute requirements for the text LM, SpidR 2.0-based SLM, and GSLM2 at each model scale.

The HuBERT- and SpidR-based SLMs use units obtained by applying k-means quantization to representations from encoder layers 11 and 6, respectively, following Poli et al. (2025b). In DinoSR, all prediction heads operate on representations from student encoder layer 12, with each head trained to predict codebook assignments derived from a corresponding teacher layer in the range 5-12. The DinoSR-based SLM uses units predicted by the head corresponding to teacher layer 5. The SpidR 2.0-based SLM uses units predicted from layer 7 of the SpidR 2.0 encoder, whereas GSLM2 uses units predicted from layers 5-10. Appendix C.1.1 describes the layer-selection criteria for the proposed models. For GSLM2, streams are generated in ascending order of bitrate, yielding the layer order $(10, 5, 6, 9, 8, 7)$ for π. Figure 12 reports the layer-wise bitrates, and Appendix C.1.4 compares alternative generation orders.

All models are optimized using AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$. Weight decay is set to 0.1 for all models except text LMs, which use 0.01. Models with 90M, 300M, 1.2B, or 2.5B parameters use a cosine-annealing learning-rate schedule with a 1k-step linear warmup. The 6.5B-parameter SLM instead uses a tri-stage schedule that allocates 10%, 40%, 50% of the total training steps to warmup, hold, and cosine decay, respectively, corresponding to 75k, 300k, and 375k steps. The 6.5B-parameter text LM follows the same allocation, with 60k, 240k, and 300k steps for the respective stages. For GSLM2, we increase the global batch size and context length to account for the reduced compression of cross-stream deduplication. Because a position is retained whenever any stream changes, individual streams may contain consecutive repeated units. With five or more streams, the resulting sequences are approximately twice as long as SpidR 2.0 sequences, as shown in Figure 5. Larger batches and longer contexts provide each stream with comparable coverage of non-repeated tokens. GSLM2 retains its performance advantage even when the batch size and context length are matched, as reported in Appendix C.1.5. For each language model, we select the checkpoint with the lowest validation loss.

Duration predictor. We train the duration predictor on deduplicated unit sequences extracted from the Libri-Light `unlab-600` subset. Training runs for 25k steps on a single NVIDIA H100 GPU with a batch size of 64. Optimization uses AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$ and a weight decay

of 10^{-5}. The learning rate increases linearly to 3×10^{-3} over the first 1k steps and then decays linearly. We select the checkpoint with the lowest validation loss.

Unit vocoder. We train the unit HiFi-GAN vocoder on non-deduplicated unit sequences extracted from LJ Speech, following the HiFi-GAN training procedure (Kong et al., 2020). Training runs up to 2k epochs on randomly sampled 0.56-second segments with a global batch size of 512 across eight NVIDIA H100 GPUs. Optimization uses AdamW with $(\beta_1, \beta_2) = (0.8, 0.99)$ and a weight decay of 0.01. The learning rate is initialized to 2×10^{-4} and decays exponentially by a factor of 0.999 after each epoch. Adversarial training uses a multi-period discriminator with periods $\{2, 3, 5, 7, 11\}$ and a multi-scale discriminator operating at three scales, both trained with a least-squares GAN objective. The generator objective combines a least-squares adversarial loss, a feature-matching loss, and an L_1 log-mel reconstruction loss weighted by 45. We use the final checkpoint for evaluation.

B.4 Evaluation Metrics, Protocols, and Configurations

This section details the metrics and protocols used to evaluate our SLMs and encoder representations. We first describe the data splits, adaptations, and scoring procedures for three linguistic benchmarks: ZeroSpeech'21 (sWuggy and sBLIMP; Appendix B.4.1), StoryCloze (tSC and sSC; Appendix B.4.2), and LongTail-Swap (Appendix B.4.3). We then detail the evaluation of generated continuations using ASR-PPL and LLMJ, together with the decoding-temperature calibration procedure (Appendices B.4.4 and B.4.5). Finally, we describe ABX and MAP evaluations for assessing encoder representation quality (Appendix B.4.6).

B.4.1 ZeroSpeech'21

We evaluate lexical and syntactic competence of our SLMs using the scoring-based tasks from ZeroSpeech'21 (Nguyen et al., 2020).

Tasks. We consider two tasks:

- **sWuggy (lexical)**: A spoken spot-the-word task in which each example pairs a real English word with a nonword (generated using Wuggy (Keuleers & Brysbaert, 2010)) and matched in phonotactics and syllabic structure (e.g., "blick" vs. "brick"). The task comprises *in-vocabulary* and *out-of-vocabulary* subsets, defined by whether the real word appears in the LibriSpeech training vocabulary. We report sWuggy-all, the mean accuracy across the two subsets.
- **sBLIMP (syntax)**: The spoken counterpart of BLiMP (Warstadt et al., 2020), designed to evaluate syntactic acceptability. Each example pairs a grammatical sentence with a minimally different ungrammatical sentence (e.g., "dogs eat meat" vs. "dogs eats meat"). The task comprises 63 subtasks grouped into 12 linguistic categories. We report mean accuracy across all sentence pairs.

Data splits. For both tasks, we use the official development sets for model selection and the official test sets for final evaluation.

Multilingual adaptations. For the VoxPopuli experiments in Appendix C.1.11, we construct French and German sBLIMP from the corresponding multilingual grammaticality pairs from MultiBLiMP (Jumelet et al., 2026). For English, we instead use the original sBLIMP benchmark from ZeroSpeech'21 rather than the English split of MultiBLiMP, as the two are not identical.

Because some written grammatical contrasts are neutralized in speech, we apply an audibility filter before synthesis. Specifically, we prompt Llama 3.1 405B[13] to act as a phonetics expert and identify pairs whose grammatical contrast is not recoverable from speech (e.g., French "aurais" vs. "aurait", which are homophones). To validate the filtering procedure, authors who are native speakers of the respective languages review a random sample of 50 pairs per language. After filtering, the French and German benchmarks contain 1,464 and 2,249 grammaticality pairs, respectively. We synthesize the retained pairs using Gemini 2.5 Pro TTS with six voices: Aoede, Charon, Fenrir, Kore, Puck,

[13] https://huggingface.co/meta-llama/Llama-3.1-405B

and Zephyr.[14] All parameters are set to their default values except temperature, which we set to 0.0 (i.e., greedy decoding).

Scoring. For each stimulus pair, a prediction is counted as correct if the model assigns a higher score to the real word in sWuggy or to the grammatical sentence in sBLIMP. We report pairwise accuracy; chance performance is 50%. For French and German sBLIMP, we average accuracy across the six voices.

B.4.2 StoryCloze

We use the Spoken StoryCloze benchmark developed by Hassid et al. (2023), which is derived from the original Story Cloze Test (Mostafazadeh et al., 2016). Each example consists of a four-sentence commonsense story and two candidate fifth sentences, one of which is the reference ending and the other a distractor.

Tasks. We consider two variants:

- **Topic StoryCloze (tSC; semantics)**: The distractor is a correct ending randomly sampled from a different story. This contrast primarily tests whether the model recognizes topical and global semantic coherence between a story and its ending.
- **Spoken StoryCloze (sSC; semantics)**: The distractor is the adversarial ending from the original Story Cloze Test. Because these endings are generally fluent and superficially plausible, sSC provides a more challenging test of causal, temporal, and discourse-level understanding.

Data splits. We evaluate both tasks on the 1,871 stories in the official spoken test set. Because Spoken StoryCloze does not provide an official development split, we construct a tSC development set for model selection and hyperparameter tuning. We draw its stories from the "test" split of the original textual Story Cloze Test benchmark, whereas the released spoken test set is derived from the original "development" split; the resulting development and test sets are therefore story-disjoint.

To match the released spoken benchmark, we synthesize each sentence independently using FastSpeech 2 (Ren et al., 2021)[15] and a HiFi-GAN vocoder (Kong et al., 2020)[16] trained on LJ Speech, then concatenate consecutive sentences with 100 ms of intervening silence. Preliminary experiments showed substantial variation in tSC accuracy across independently trained model seeds when only one distractor was used per story. We therefore sample ten distractors from other stories for each of the 1,871 development stories, yielding ten correct-incorrect pairs per story and 18,710 pairs in total. This expanded set provides a more stable validation signal.

Multilingual adaptations. For the VoxPopuli experiments in Appendix C.1.11, we construct French and German adaptations of tSC from the original English text benchmark (Mostafazadeh et al., 2016). We translate the stories and candidate endings sentence by sentence using Llama 3.3 70B[17] with structured prompts. To ensure that the translated task remains well formed and solvable, authors who are native speakers of the respective target languages review a random sample of 50 pairs per language, and we correct any issues they identify. We synthesize the translated pairs using the same six-voice Gemini 2.5 Pro TTS configuration as for multilingual sBLIMP.

Scoring. For each stimulus pair, the model scores the two complete five-sentence stories. A prediction is counted as correct if the story containing the reference ending receives the higher score. We report pairwise accuracy; chance performance is 50%. For French and German tSC, we average accuracy across the six voices.

[14] https://ai.google.dev/gemini-api/docs/models/gemini-2.5-pro-preview-tts

[15] https://github.com/facebookresearch/fairseq/tree/main/examples/speech_synthesis

[16] https://github.com/jik876/hifi-gan

[17] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

B.4.3 LONGTAIL-SWAP

We develop a spoken adaptation of LT-Swap (Algayres et al., 2025), a corpus-dependent benchmark that evaluates linguistic knowledge across the word-frequency distribution.

Tasks. LT-Swap comprises three complementary tasks. Each example is a quadruplet (S_1, S_2, S_1^*, S_2^*), where S_1 and S_2 contain two target forms in their intended contexts, and S_1^* and S_2^* are obtained by exchanging those forms.

- **WordSwap (WS; semantics)**: Two target words from the same part of speech and nonzero frequency bin are placed in separate contexts in which they are semantically appropriate. Exchanging the targets yields sentences that remain syntactically well formed but become semantically anomalous. For example,

  S_1 : "The *cat* is sleeping on the mat." S_1^* : "The *boat* is sleeping on the mat."
  S_2 : "The *boat* is sailing on the sea." S_2^* : "The *cat* is sailing on the sea."

- **InflectionSwap (IS; morphosyntax)**: Two inflected forms of the same lemma, such as the bare and present-participle forms of a verb, are placed in contexts that license their respective forms. Exchanging them creates morphosyntactic violations while holding the underlying lemma fixed. For example,

  S_1 : "He couldn't *sleep* last night." S_1^* : "He couldn't *sleeping* last night."
  S_2 : "The baby was *sleeping* peacefully." S_2^* : "The baby was *sleep* peacefully."

- **AgreementSwap (AS; syntax)**: Singular and plural target forms are exchanged between contexts, breaking a targeted agreement relation. The task covers subject-verb, antecedent-anaphor, and determiner-noun agreement, with short- and long-distance variants where applicable. For example, a subject-verb agreement quadruplet is:

  S_1 : "The *strategist* analyzes." S_1^* : "The *strategists* analyzes."
  S_2 : "The *strategists* analyze." S_2^* : "The *strategist* analyze."

Benchmark construction. For each Libri-Light training-data scale, we compute word frequencies from the corresponding subset of Libriheavy transcriptions. Following the original benchmark, we retain nouns and verbs and group their surface forms into power-of-two frequency bins: $\{[0], [1, 2), [2, 4), [4, 8), \ldots, [256, 512), [512, \infty)\}$. The zero-frequency bin, which applies only to IS and AS, contains automatically derived inflected forms that do not occur in the corresponding training corpus.

Following the original LT-Swap pipeline,[18] we use the instruction-following Llama 3.3 70B model to generate candidate sentences and filter ambiguous examples. A quadruplet is retained only if the filtering model reliably distinguishes both intended sentences from their corrupted counterparts. For WS, the two targets are drawn from the same frequency bin. For IS and AS, each quadruplet is assigned to the bin of its less frequent target form. For WS, IS, and AS, respectively, this yields 31,952, 33,218, and 40,928 sentence pairs at 600 hours; 41,418, 69,834, and 73,132 at 6K hours; and 39,166, 105,624, and 99,860 at 60K hours.

To create the spoken benchmark, we synthesize all four sentences in each quadruplet using Piper TTS[19] with the Kathleen voice. Each waveform is tokenized separately by the speech encoder associated with the evaluated SLM, whereas text LMs are evaluated on the corresponding text.

Scoring. Each quadruplet yields two binary comparisons: S_1 vs. S_1^* and S_2 vs. S_2^*. A comparison is counted as correct if the intended sentence receives a higher score than its corrupted counterpart. We compute accuracy separately for each task and word-frequency bin; chance performance is 50%. For each task, Table 2 reports the macro-average across its applicable frequency bins. A breakdown by word-frequency bin is provided in Appendix C.1.9.

[18] https://github.com/facebookresearch/lt-swap
[19] https://github.com/OHF-Voice/piper1-gpl

B.4.4 ASR-PPL

We assess the linguistic plausibility of generated speech continuations using the perplexity of their ASR transcripts. Perplexity alone can be misleading, however, because repetitive, low-temperature generations may achieve artificially favorable scores. Following the motivation of auto-BLEU (Lakhotia et al., 2021), we therefore compare models using a common calibration target for lexical repetition. For each model, we calibrate the decoding temperature on a development set to target the ground-truth baseline under the combined normalized unigram-bigram repetition score defined below, and then evaluate the model on the test set using the calibrated temperature. The complete evaluation protocol is described below.

Speech-continuation data. We sample 1,024 LibriSpeech utterances with durations between 6 and 15 seconds. We report results on utterances from `test-clean` and calibrate temperatures on an identically constructed set of 1,024 utterances from `dev-clean`. Each utterance is divided into a prompt and a continuation at approximately the 3-second mark. Using forced alignments, we shift the split point to the nearest word boundary to avoid splitting a word. Note that the transcripts in both `dev-clean` and `test-clean` are normalized (i.e., punctuation and casing removed).

Unit generation. For each utterance, we encode the prompt with a speech encoder to obtain ground-truth speech units. Conditioned on these units, each SLM autoregressively generates a continuation of up to 10 seconds. We use nucleus sampling with $p = 0.9$ and calibrate the temperature separately for each model. For GSLM2, the generative head predicts the streams autoregressively within each time step. To mitigate the accumulation of sampling noise across successive stream predictions, we linearly anneal the temperature from its calibrated value for the first stream to 50% of that value for the final stream, resetting the schedule at every time step. Appendix C.1.5 evaluates this annealing strategy. Text LMs are conditioned on the reference transcript up to the same prompt boundary and generate 48 tokens using the same nucleus-sampling parameter and model-specific calibrated temperatures. This token budget yields approximately 30 words on average, closely matching the average word count of the SLM continuations.

Unit synthesis. Generated units are converted to waveforms using the vocoder corresponding to their speech-unit space. For deduplicated unit streams, a separately trained duration predictor restores the temporal information removed by deduplication by predicting the number of frames associated with each generated unit. The generated continuation is then expanded to the original frame rate, while the prompt units preserve their original frame-level durations. The complete prompt-continuation sequence is synthesized using the unit HiFi-GAN vocoder trained on the corresponding unit representation.

ASR and text normalization. We split each synthesized waveform at the known prompt boundary and transcribe only the generated continuation using Qwen3-Omni[20]. For ASR-PPL, we condition on the ground-truth prompt transcript and normalize both prompt and continuation by lowercasing, removing punctuation, expanding contractions, and spelling out numerals. For LLMJ, we instead use a semantics-preserving normalization that lowercases text while retaining apostrophes and sentence-boundary punctuation. For each evaluation, we apply the corresponding task-specific normalization to both ASR transcripts and direct text-LM outputs, making the textual representations of speech and text systems as comparable as possible.

Conditional perplexity. We compute ASR-PPL as the conditional perplexity of the normalized ASR transcript of each continuation, conditioned on the normalized ground-truth prompt. Let W^{p} and W^{c} denote the corresponding prompt and continuation token sequences, respectively. Both sequences are tokenized using the tokenizer associated with Llama-3.2-1B (Grattafiori et al., 2024)[21]. Frozen Llama-3.2-1B scores each continuation conditioned on its prompt as

$$\text{ASR-PPL}\left(W^{\mathsf{c}} \mid W^{\mathsf{p}}\right) = \exp\left[-\frac{1}{|W^{\mathsf{c}}|}\sum_{u=1}^{|W^{\mathsf{c}}|}\log p_{\text{LLM}}\left(w_u^{\mathsf{c}} \mid W^{\mathsf{p}}, w_{<u}^{\mathsf{c}}\right)\right]. \tag{9}$$

[20] https://huggingface.co/Qwen/Qwen3-Omni-30B-A3B-Instruct
[21] https://huggingface.co/meta-llama/Llama-3.2-1B

The prompt is provided only as left context and is excluded from the loss, so ASR-PPL is computed solely over continuation tokens.

Temperature calibration. Decoding temperature controls the trade-off between generation quality and diversity. Evaluating all models at a common temperature could therefore compare systems operating at substantially different levels of lexical repetition. We instead calibrate each model's decoding temperature, for both SLMs and Text LMs, to target the ground-truth repetition baseline. Appendix C.1.10 reports additional results under a fixed decoding temperature.

Because auto-BLEU depends on sequence length, e.g., longer transcripts offer more opportunities for repetition, we compute it over the first 15 words of each normalized continuation transcript. Prior work commonly uses bigram auto-BLEU (Lin et al., 2025; Maimon et al., 2025a), but we find bigram repetition too sparse for reliable calibration in our setting: only 3.2% of ground-truth development continuations contain a repeated bigram. We therefore combine unigram and bigram repetition, with the unigram component providing a denser calibration signal. Let $\text{auto-BLEU}_n(\tau)$ denote the mean n-gram auto-BLEU of continuations generated at temperature τ, and let $\text{auto-BLEU}_n^{\text{gt}}$ denote the corresponding mean for ground-truth continuations. We define the unigram-bigram normalized repetition score (NRS) as

$$\text{NRS}(\tau) = \frac{1}{2}\left(\frac{\text{auto-BLEU}_1(\tau)}{\text{auto-BLEU}_1^{\text{gt}}} + \frac{\text{auto-BLEU}_2(\tau)}{\text{auto-BLEU}_2^{\text{gt}}}\right). \tag{10}$$

Thus, $\text{NRS} = 1$ indicates a match to the ground-truth baseline under this combined unigram-bigram repetition measure, while values above and below one indicate greater and lower repetition according to this measure, respectively.

For each model, we evaluate NRS over a temperature grid with increments of 0.05. We then linearly interpolate between the two adjacent grid points that bracket $\text{NRS} = 1$ to obtain the calibrated temperature $\hat{\tau}$ at a resolution of 0.01. The resulting development-set calibration curves are shown in Appendix C.1.3. No test-set statistics are used for calibration. Once selected, the model-specific temperatures are fixed and used unchanged for test-set generation.

For both development and test sets, we evaluate each model over ten runs. Within each run, we summarize ASR-PPL by the median and NRS by the mean across examples. We then report the mean and standard deviation of these per-run statistics over the ten runs, as in Table 4.

B.4.5 LLM-JUDGE

We use Prometheus 2 (Kim et al., 2024) as an LLM-based judge to evaluate generated continuations. The grading prompt below follows the official absolute-grading format[22], with adaptations for open-ended continuation evaluation.

[22] https://huggingface.co/prometheus-eval/prometheus-8x7b-v2.0

###Task Description:
An instruction (might include an Input inside it), a response to evaluate, a reference answer that gets a score of 5, and a score rubric representing a evaluation criteria are given.

1. Write a detailed feedback that assess the quality of the response strictly based on the given score rubric, not evaluating in general.
2. After writing a feedback, write a score that is an integer between 1 and 5. You should refer to the score rubric.
3. The output format should look as follows: "Feedback: (write feedback for the criteria) [RESULT] (an integer number between 1 and 5)"
4. Please do not generate any other opening, closing, and explanations.

###The instruction to evaluate:
Evaluate the passage below. <CONTINUATION_START> separates the given prompt from the generated continuation. Score only the continuation, but read both sides together. The boundary may be inside a sentence, name, quotation, or question, and the continuation may end at a generation limit.
The prompt and response use the same semantic normalization: lowercase text with apostrophes and sentence boundaries preserved. They may contain minor rendering or transcription artifacts. Judge the intended content when it remains understandable.

###Response to evaluate:
{prompt_transcript}<CONTINUATION_START>{continuation_transcript}

###Reference Answer (Score 5):
Any natural, coherent, non-degenerate continuation that fits the marked boundary. Multiple answers are valid; it need not match a particular event, add detail, or finish the passage.

###Score Rubrics:
[Semantic continuation quality: Judge boundary fit, internal coherence, and non-degeneration from the actual words. Missing boundary punctuation and newly introduced entities are not errors when the connection is clear. Ordinary word reuse is cohesion; claim copying only for an actual repeated multiword phrase. Forgive isolated understandable transcription artifacts and ignore only the final unfinished fragment. Do not invent missing links or let a coherent opening excuse later unrelated, contradictory, repetitive, or uninterpretable text. Choose the highest fully satisfied score. Give at most two feedback sentences and 60 words, cite the strongest evidence, and end with [RESULT] N.]
Score 1: No connected meaning is recoverable, or most of the continuation is unrelated, contradictory, uninterpretable, copied, boilerplate, or looping.
Score 2: A local connection exists, but major or repeated failures make the overall meaning difficult to recover.
Score 3: The main meaning is recoverable and mostly connected, but a substantive defect or several local errors materially weaken it.
Score 4: The continuation fits and remains coherent, with only localized minor errors and no substantive defect.
Score 5: Every complete span fits, is coherent and natural, and has no identified semantic defect.

###Feedback:

For SLMs, we reuse the ASR transcripts generated for the ASR-PPL evaluation; for text LMs, we evaluate the generated text directly. We apply the same semantic normalization to both modalities, preserving apostrophes and sentence boundaries. The prompt and continuation are presented as a single passage separated by the explicit boundary marker "<CONTINUATION_START>", and only the continuation is scored. Prometheus 2 assigns an integer score from 1 to 5 based on boundary compatibility, internal coherence, and non-degeneration. The rubric allows the coherent introduction of new entities and ordinary lexical reuse, while penalizing contradictions, unrelated content, multiword phrase copying, and repetitive loops. It also tolerates isolated, intelligible transcription errors and disregards an unfinished final fragment caused by the generation limit. We evaluate each model using the temperature calibrated during the ASR-PPL evaluation. We conduct ten runs for each system and report the mean and standard deviation, as in Table 4.

B.4.6 ABX AND MAP

We compute ABX (Schatz, 2016) and MAP (Carlin et al., 2011) scores on both the `dev-clean` and `dev-other` subsets of the ZeroSpeech'21 development set (Nguyen et al., 2020), using the provided phoneme- and word-level alignments, respectively.

ABX is a discrimination task involving a triplet of representations (a, b, x), where a and b belong to known classes A and B, respectively, and the class of x is unknown but belongs to $\{A, B\}$. x is assigned to class A if $d(x, a) < d(x, b)$ and to class B if $d(x, b) < d(x, a)$. We report the error rate for discriminating between triphone classes, where a triphone is a sequence of three consecutive phonemes. We evaluate both within-speaker and across-speaker conditions. In the within-speaker condition, all three tokens are produced by the same speaker. In the across-speaker condition, a and b are produced by the same speaker, while x is produced by a different speaker. For continuous ABX, we measure local distances between continuous representations using cosine distance. For discrete ABX, we represent the argmax code assignments as one-hot vectors and use cosine distance. In both cases, we aggregate local distances into sequence-level distances using dynamic time warping. We

use the fastabx library (Poli et al., 2025a) for scoring, with chance performance corresponding to a 50% error rate.

For MAP, we represent each word token by a single embedding obtained by mean-pooling frame representations over its aligned word segment. Word classes correspond to orthographic word types from the forced alignments. MAP evaluates a retrieval task: given any query word token q of class W, all other word tokens in the corpus are ranked by cosine similarity $\cos(q, \cdot)$. Over this ranking, we compute average precision as $\mathrm{AP}(q) = \frac{1}{R}\sum_{k=1}^{N} \mathrm{Precision@}k\,\mathrm{rel}_k$, where R is the number of other tokens of class W, N is the total number of retrieval candidates, $\mathrm{Precision@}k$ is the proportion of relevant tokens among the top k results, and rel_k equals 1 if the token at rank k belongs to class W and 0 otherwise. MAP is the mean of $\mathrm{AP}(q)$ over all query tokens. We use the speech-map library (Algayres et al., 2020) for scoring, with a chance MAP score of 0.56%.

C Additional Experimental Results

This section presents additional experimental results that provide further insight into the design of GSLM2 and complement the main findings in Section 3. The remainder of the section is organized as follows:

- Appendix C.1: Supplementary SLM evaluation
- Appendix C.2: Supplementary encoder evaluation
- Appendix C.3: Speech resynthesis evaluation
- Appendix C.4: Statistical significance analysis

C.1 Supplementary SLM Evaluation

This section supplements the main SLM results in Section 3.2 with additional experiments and analyses. We first describe how we select encoder layers and the number of streams (Appendix C.1.1), and estimate scaling efficiency relative to text LMs (Appendix C.1.2). We then examine how decoding temperature and stream generation order affect continuation quality and repetition (Appendices C.1.3 and C.1.4). We assess the contributions of self-distillation and single-stream supervision through ablations (Appendix C.1.5), compare scoring heads and stream aggregation strategies (Appendix C.1.6), and test whether GSLM2 benefits from complementary information across encoder layers (Appendix C.1.7). Further evaluations compare GSLM2 with speech-capable LLMs (Appendix C.1.8), analyze LT-Swap performance by word frequency (Appendix C.1.9), and assess the robustness of generation rankings to decoding temperature (Appendix C.1.10). Finally, we report results for English, French, and German models trained on VoxPopuli (Appendix C.1.11) and compare direct speech modeling with cascaded ASR-LM systems to examine how transcription affects linguistic benchmark scores (Appendix C.1.12).

C.1.1 Layer Selection Strategy

We select the encoder layers for SpidR 2.0 SLM and GSLM2 based on aggregate development-set performance, defined as the mean accuracy across the development splits of sWuggy, sBLIMP, and tSC.

Figure 4 compares SpidR 2.0 SLM performance using speech units extracted from different encoder layers. To enable faster iteration, we conduct these experiments in the 6K-hour Libri-Light setting with the 90M-parameter configuration. Although the three tasks favor different layers, mean accuracy is highest at layer 7. We therefore use units extracted from layer 7 for SpidR 2.0 SLM.

Figure 5 shows the effect of varying the number of streams in GSLM2 under the same experimental setup. Streams are added in descending order of their mean development-set accuracy in Figure 4. Thus, the single-stream model uses layer 7, the two-stream model combines layers 7 and 8, and each subsequent model adds the next-best layer. Accuracy improves as streams are added, peaking at six streams before declining. We define the compression ratio as the sequence length before (cross-stream) deduplication divided by the sequence length afterward; thus, higher values indicate greater compression. The ratio decreases steadily as more streams are added, indicating that cross-stream

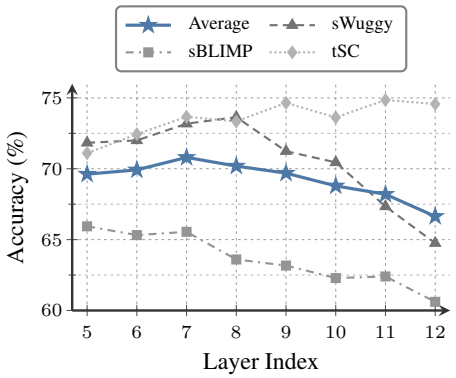


Figure 4: Development-set accuracy of SpidR 2.0 SLM (Libri-Light 6K h; 90M parameters) using units extracted from different encoder layers. Average denotes the mean across sWuggy, sBLIMP, and tSC.

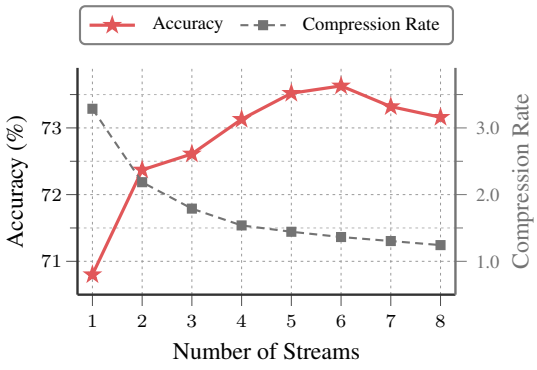


Figure 5: Development-set mean accuracy and compression ratio of GSLM2 (Libri-Light 6K h; 90M parameters) for different numbers of streams. Streams are added using encoder layers in descending order of the mean accuracy reported in Figure 4.

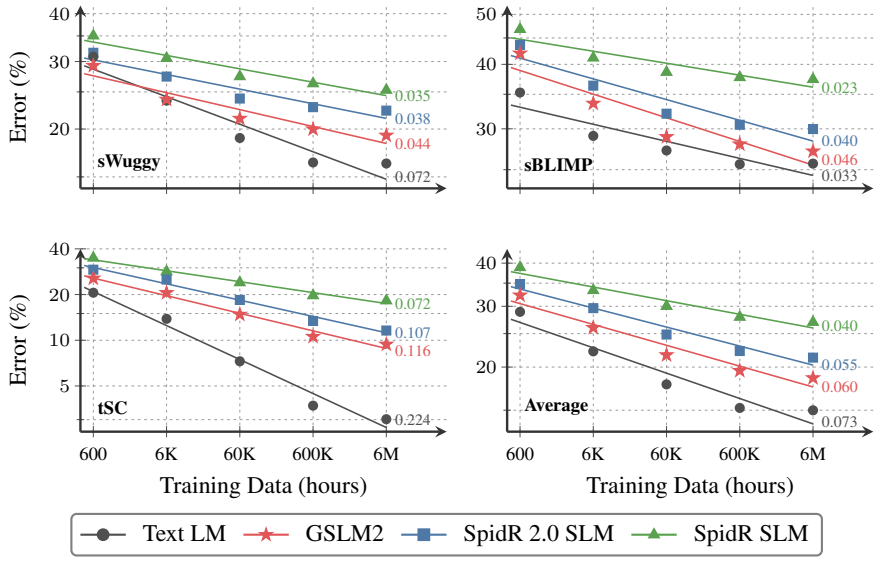


Figure 6: Error scaling on conversational speech, both axes logarithmic. Each marker is a trained model; lines are the fitted power law $\text{error} = a\,N^{-\gamma}$, whose slope is $-\gamma$. Fitted exponents γ are given in each panel; a larger γ means error falls faster as training data grows.

deduplication becomes less effective and yields longer sequences. We use six streams with layers $\mathcal{J} = \{5, 6, \ldots, 10\}$, which achieves the highest development-set accuracy with a compression ratio of 1.36. However, this layer-selection heuristic is optional, and GSLM2 can use all eight layers and still substantially outperform the single-stream model.

C.1.2 Estimating Scaling Efficiency

To estimate scaling efficiency in Figure 2, we fit the error, $100-\text{accuracy}$, as a power law in training hours, $\text{error} = a\,N^{-\gamma}$. Taking logarithms of both quantities makes this a straight line of slope $-\gamma$, which we estimate by ordinary least squares over the five measured scales. Here, γ measures error reduction in relative terms—what fraction of the remaining error is removed per unit of data. It therefore does not shrink as the curves flatten near the ceiling (as can be seen in Figure 2), making it more appropriate than a slope fitted directly to raw accuracy.

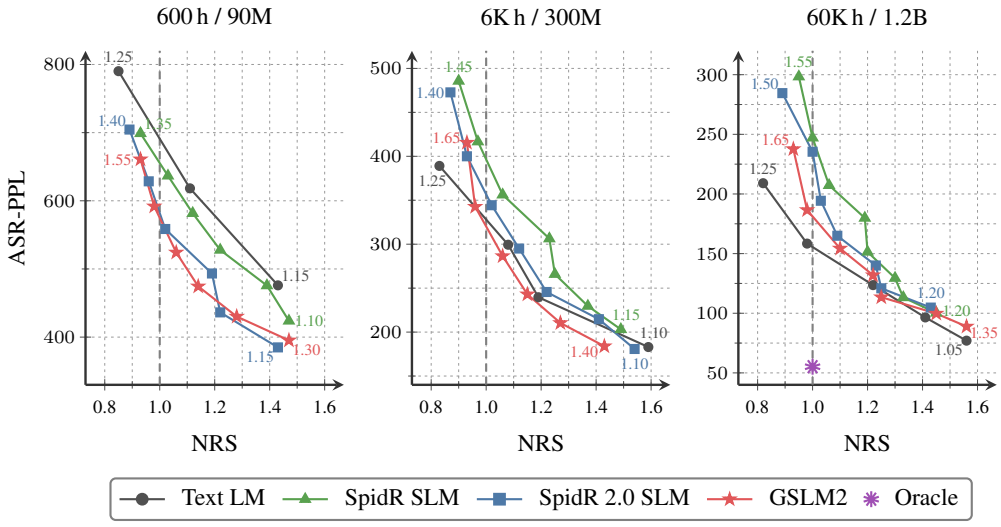


Figure 7: Quality-diversity sweeps for temperature calibration. Each point represents an evaluated temperature, and endpoint labels indicate the temperature range of each sweep. The dashed line at NRS $= 1$ indicates the calibration target, where generated continuations match the repetition level of ground-truth continuations under combined unigram-bigram auto-BLEU. Values above and below one indicate greater and lower repetition, respectively. Results are averaged over ten random runs. The oracle is computed directly from the ground-truth continuations.

Figure 6 shows the resulting fits with both axes logarithmic; the measured points fall close to straight lines (all R^2 values are above 0.92 on the average accuracy). On the average metric, we obtain $\gamma = 0.060$ for GSLM2, 0.055 for SpidR 2.0 SLM and 0.040 for SpidR SLM. The same ordering holds on each of sWuggy, sBLIMP and tSC individually. The Text LM remains ahead at $\gamma = 0.073$, so GSLM2 narrows the efficiency gap. We also see that every model scales fastest on tSC ($\gamma = 0.116$ for GSLM2, against 0.044 on sWuggy), indicating that semantic competence is where continued scaling pays most.

The two modeling changes can be separated by comparing consecutive pairs of models in log-error space, where a vertical offset corresponds to a constant multiplicative reduction in error. Replacing SpidR units with SpidR 2.0 widens the gap steadily with scale, from 0.048 to 0.103 log units on the average metric. Adding multi-stream modeling on top of SpidR 2.0 units instead produces a gap that is mostly flat. The exponents show the same split: of the total difference in γ between SpidR and GSLM2 ($+0.021$), the unit upgrade accounts for $+0.016$ and multi-stream modeling for $+0.005$. The unit encoder change therefore alters the rate at which error falls with data, whereas multi-stream modeling shifts the curve down by a constant factor.

C.1.3 Calibration Curves for Generative Evaluation

Figure 7 shows the development-set quality-diversity sweeps used to calibrate the decoding temperature (see Appendix B.4.4). Across models, increasing the temperature generally raises both NRS and ASR-PPL over the evaluated range.

Each sweep brackets the target NRS $= 1$. For each model, we estimate the calibrated temperature $\hat{\tau}$ by linearly interpolating between the two adjacent points that cross this target and rounding the result to the nearest 0.01. The resulting temperatures, reported in Table 4, enable comparisons using a common calibration target rather than at a fixed decoding temperature. ASR-PPL is reported to characterize generation quality and does not influence temperature selection. All temperatures are calibrated on the development set and then held fixed for test-set generation.

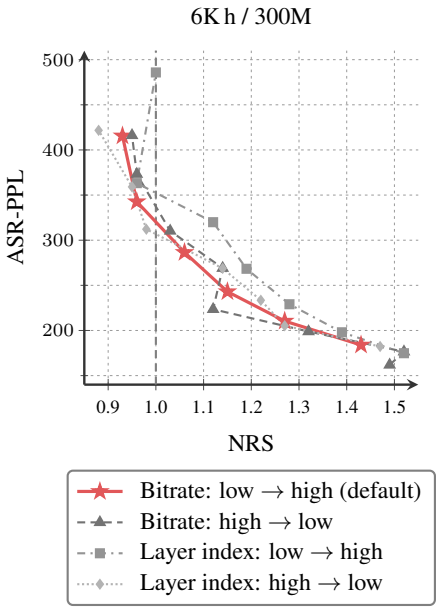


Figure 8: Effect of layer generation order on the quality-diversity curve for GSLM2 (Libri-Light 6K h; 300M parameters). The adopted low-to-high bitrate order is highlighted in red, while the alternative orders are shown in gray.

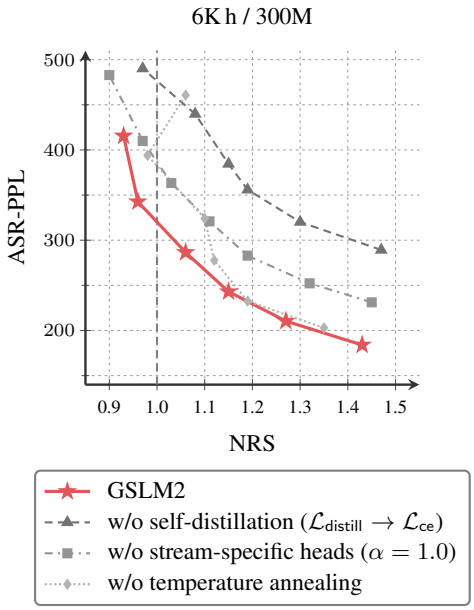


Figure 9: Ablation study of GSLM2 on generative performance. The default model (Libri-Light 6K h; 300M parameters) is highlighted in red, while the ablated variants are shown in gray.

C.1.4 Effect of Layer Order on Generation Performance

Figure 8 compares the development-set quality-diversity curves obtained with different generation orders for 300M-parameter GSLM2 trained on 6K-hour Libri-Light. We consider two bitrate-based orders for π (see Figure 12 for the layer-wise bitrates): ascending bitrate with $(10, 5, 6, 9, 8, 7)$, and descending bitrate with $(7, 8, 9, 6, 5, 10)$. We also evaluate ascending and descending layer-index orders, given by $(5, 6, 7, 8, 9, 10)$ and $(10, 9, 8, 7, 6, 5)$, respectively.

Among the bitrate-based orders, descending bitrate shows higher ASR-PPL near the calibration target at $\text{NRS} = 1$ and exhibits a less smooth trajectory in the low-temperature range. In contrast, ascending bitrate produces a smoother and more predictable trade-off, making temperature calibration more reliable. The layer-index orders exhibit a larger difference. Ascending layer order causes ASR-PPL to increase sharply near the calibration target and produces an irregular curve in this region. Descending layer order performs substantially better and remains close to the ascending-bitrate configuration. However, layer index depends on the set of modeled layers, whereas bitrate directly reflects the information rate of each stream.

We therefore adopt ascending bitrate as the generation order for GSLM2. This choice combines competitive generation performance with stable calibration behavior and remains applicable when the set of layers used for multi-stream modeling changes. By generating lower-bitrate streams before higher-bitrate ones, it also induces a coarse-to-fine schedule, which is consistent with prior work demonstrating the effectiveness of coarse-to-fine generation (Copet et al., 2023; Borsos et al., 2023; Défossez et al., 2024). Across the generation orders considered, performance on the other benchmarks (i.e., ZS21, SC, and LT-Swap) remains essentially unchanged; the observed differences are confined to generative evaluation.

C.1.5 Ablation Studies

We ablate training objectives and configuration of GSLM2 to assess their contributions to the primary metrics, i.e., sWuggy, sBLIMP, and tSC (Table 11). We also evaluate the effects of stream-specific supervision and temperature annealing on generative performance (Figure 9).

Table 11: Ablations of GSLM2 (Libri-Light 6K h; 300M parameters) on the primary evaluation metrics, including sWuggy, sBLIMP, and tSC. A1 and A3 modify the full model (A0), while A2 builds on A1. Values are accuracies (%; chance level: 50%).

		ZS21 (↑)		SC (↑)
ID	Model	sWuggy	sBLIMP	tSC
A0	GSLM2	**76.3**	**67.2**	**78.9**
A1	↪ w/o self-distillation ($\mathcal{L}_{\text{distill}} \rightarrow \mathcal{L}_{\text{ce}}$)	68.5	57.3	70.1
A2	↪ w/o single-stream loss ($\beta = 0.0$)	67.9	55.8	65.4
A3	↪ Halve context length & batch size	75.7	66.8	78.3
A4	SpidR 2.0 SLM	73.9	65.1	74.7
A5	↪ matched GSLM2 training FLOPs	73.3	64.5	75.1

Self-distillation is critical to performance gains. In A1 (Table 11), we replace the self-distillation loss with a cross-entropy loss ($\mathcal{L}_{\text{distill}} \rightarrow \mathcal{L}_{\text{ce}}$ in $\mathcal{L}_{\text{slm}}$), training the stream-specific heads on hard speech-unit targets at the same transition positions. Relative to the full model (A0), this reduces sWuggy, sBLIMP, and tSC by 7.8, 9.9, and 8.8 percentage points, respectively. Building on A1, A2 sets $\beta = 0.0$, removing the single-stream loss $\mathcal{L}_{\text{single}}$ and reallocating its weight to the cross-entropy loss. Training then uses only the cross-entropy and generative losses, and scores decline by a further 0.6, 1.5, and 4.7 points, respectively. The substantial degradation in A1 highlights the importance of transition-level self-distillation. The additional decline in A2 suggests that retaining the single-stream objective remains beneficial even without self-distillation, particularly on tSC.

Performance is largely retained with shorter contexts and smaller batches. GSLM2 uses longer contexts and larger batches to maintain coverage of non-repeated speech units (see Appendix B.3). Halving both settings (A3) reduces sWuggy, sBLIMP, and tSC by only 0.6, 0.4, and 0.6 points, respectively, relative to the full model (A0). The resulting model still outperforms the single-stream baseline in A4. It also requires 10% fewer training FLOPs than the full configuration (Table 10), offering lower computational cost with only modest performance degradation. In addition, increasing the batch size of SpidR 2.0 SLM to match GSLM2's training FLOPs (A5) does not improve its overall performance.

Self-distillation also improves generative performance. For generation, we evaluate the same replacement of self-distillation with cross-entropy used in A1. We also evaluate a separate variant that removes the stream-specific heads entirely and sets $\alpha = 1.0$ in $\mathcal{L}_{\text{slm}}$, retaining only the multi-stream generative loss $\mathcal{L}_{\text{gen}}$. The full model achieves lower ASR-PPL than both variants at comparable NRS values across the evaluated range (Figure 9), supporting the benefit of self-distillation over cross-entropy supervision and of stream-specific supervision more broadly. These gains are consistent with the auxiliary objectives regularizing the shared temporal representations used by the generative head through per-stream transition modeling and consistency between single-stream and multi-stream predictions.

Temperature annealing improves and stabilizes generation. During generation, GSLM2 anneals the sampling temperature across successive stream predictions within each tuple (see Appendix B.4.4). Without annealing, ASR-PPL is higher at comparable NRS values, including the calibration target $\text{NRS} = 1$. At higher decoding temperatures, NRS decreases after exceeding this target, while ASR-PPL continues to rise. With annealing, NRS increases monotonically over the evaluated temperature range, yielding a more predictable quality-diversity curve. These observations are consistent with annealing helping to limit the accumulation of sampling noise across stream predictions within each tuple.

C.1.6 Scoring Choices of GSLM2

We examine how the choice of output head affects paired-stimulus evaluation of GSLM2, using the same model trained at 6K hours in Table 2. Our default scoring protocol uses the stream-specific heads: for each stream, we average log probabilities over its unit-change positions, including the first unit, and then average the resulting scores across streams. This is consistent with the event selection used by independently deduplicated single-stream baselines, while retaining the full multi-stream

Table 12: Performance of GSLM2 (Libri-Light 6K h; 300M parameters) using different heads for scoring. Stream-specific heads score unit-change positions, whereas the generative head scores all retained positions. Individual rows select one stream's score while retaining all input streams; "All streams" averages per-stream scores before making paired comparisons. Scores are accuracies (%; chance level: 50%).

		Stream-specific head			**Generative head**		
Order j	Layer $\pi(j)$	sWuggy	sBLIMP	tSC	sWuggy	sBLIMP	tSC
1	10	70.1	62.8	77.2	57.2	53.6	59.3
2	5	71.7	64.8	73.7	61.8	53.5	55.9
3	6	72.1	64.7	77.6	57.5	52.0	53.2
4	9	71.5	64.0	79.3	60.9	52.2	61.8
5	8	72.3	63.0	77.7	55.6	50.8	51.8
6	7	72.8	65.2	77.2	56.2	51.5	54.6
All streams		76.3	67.2	78.9	66.7	56.3	61.7

history as context. For comparison, we evaluate the generative head by averaging log probabilities over all retained positions, including per-stream repetitions. Averaging these scores across streams gives the joint sequence log-likelihood normalized by the number of predicted units.

Table 12 reports accuracies obtained by scoring individual streams while retaining the full multi-stream input. Stream-specific predictions condition only on the preceding multi-stream tuples. For the generative head, each row instead scores the corresponding conditional factor, which additionally conditions on the observed units of earlier streams within the current tuple. Stream-specific scoring outperforms generative scoring for every stream on all three tasks, supporting our default transition score for these linguistic comparisons. The gap persists at $j = 1$, where both heads condition only on temporal context. Thus, within-tuple conditioning at evaluation cannot alone explain the difference.

These findings are consistent with prior work reporting benefits from removing consecutive unit repetitions in spoken language modeling (Lakhotia et al., 2021). Related approaches separate content and duration modeling: Kharitonov et al. (2022) model deduplicated units and their durations separately, while Nguyen et al. (2023) show that supervision at unit-change positions improves unit prediction even when repeated inputs are retained for cross-stream alignment. The stream-specific heads similarly emphasize transitions in both training and scoring. The generative head additionally predicts and scores the persistence events retained by cross-stream deduplication. Including these repetition terms in the averaged score is likely to reduce its sensitivity to the linguistic contrasts being evaluated.

Aggregating stream scores improves sWuggy and sBLIMP over the best individual stream-specific results. On tSC, aggregation remains close to the strongest individual result from layer 9. We examine this ensembling effect further in Appendix C.1.7. Our main evaluations use the same unweighted all-stream transition score across benchmarks, avoiding task-specific output-stream selection. These results assess paired-stimulus discrimination; generation uses the generative head and is evaluated separately through prompt-conditioned continuations.

C.1.7 Effect of Layer Diversity

We investigate whether GSLM2's gains arise from complementary information across encoder layers or from aggregating multiple noisy realizations of the same representation. Here, we compare the standard six-layer configuration (from the 6K h / 300M setup in Table 2) with two six-stream controls derived exclusively from layer 7, which is used by the single-stream SpidR 2.0 SLM. All three configurations use independently initialized stream-specific heads, trained jointly with a shared temporal backbone. The first control repeats the layer-7 argmax sequence across streams, testing score aggregation across independently initialized heads over identical units. The second independently samples units from the layer's categorical distributions for each stream, introducing variation in unit identities and stream boundaries. This sampling control is motivated by the benefits of stochastic tokenization in neural machine translation (Kudo, 2018; Provilkov et al., 2020) and consistency training across alternative tokenizations in cross-lingual transfer (Wang et al., 2021b). We adjust

Table 13: Effect of layer diversity. All GSLM2 variants use six streams and cross-stream deduplication. The repeated-layer control copies the layer-7 argmax sequence across streams; the sampled control independently samples units from the same layer's categorical distributions for each stream. Scores are accuracies (%; chance level: 50%).

		ZS21 (↑)		**SC** (↑)
Model	**Unit layers**	sWuggy	sBLIMP	tSC
SpidR 2.0 SLM	7	73.9	65.1	74.7
GSLM2	5–10	**76.3**	**67.2**	**78.9**
	7 (repeated)	73.7	65.3	74.4
	7 (sampled)	69.9	61.0	68.4

the sampling temperature to match GSLM2's cross-stream deduplication rate (i.e., 1.36 from Appendix C.1.1).

As shown in Table 13, the standard multi-layer configuration outperforms both controls on all three benchmarks. Repeating layer 7 yields performance close to the single-stream baseline, while categorical sampling reduces accuracy. Thus, neither score aggregation across independently initialized heads over identical units nor stochastic tokenization at a matched deduplication rate reproduces GSLM2's gains in these experiments. These findings support the importance of layer diversity and are consistent with our hypothesis that different encoder layers provide complementary information.

C.1.8 Comparison with Speech-Capable LLMs

To compare GSLM2 with recent speech-capable LLMs, we evaluate instruction-tuned models that accept speech directly on identical paired-stimulus benchmarks. These models do not expose likelihood scores for their speech inputs, and the likelihood-based evaluation used for conventional SLMs is not applicable. Therefore, we reformulate each benchmark item as a forced-choice question between two recordings: the benchmark's correct candidate and its incorrect counterpart.

Prompting. All models receive the following system prompt:

> You are participating in a speech understanding benchmark.
> You will hear spoken audio options labeled A and B. Your task is to choose the better option according to the instruction.
> Answer with exactly one character: A or B. Do not explain your answer. Do not transcribe the audio.

The two recordings are provided as options A and B jointly with the task-specific instruction:

- sWuggy:

  > You are given two spoken word candidates, Option A and Option B.
  > Choose the option that sounds more like a real English word.

- sBLIMP:

  > You are given two spoken English sentence candidates, Option A and Option B.
  > Choose the option that sounds more grammatical, natural, and well-formed in English.

- tSC and sSC:

  > You are given two spoken stories, Option A and Option B.
  > Each story contains the same context but a different ending.
  > Choose the story whose ending is more coherent, plausible, and natural given the preceding context.

Scoring. We use greedy decoding with a 256-token output limit configured to each model's native multimodal interface. A strictly valid response yields an unambiguous A or B choice. Invalid responses are simply counted as incorrect. To measure positional bias, we doubly evaluate each pair, reversing the order of the correct and incorrect recordings. The reported score pools decisions

Table 14: Results for open-weight and proprietary speech-capable LLMs on ZeroSpeech'21 (ZS21) and StoryCloze (SC). Speech LLMs use prompted forced-choice evaluation. GSLM2 is included as a speech-only reference evaluated using likelihood-based scoring. Scores are accuracies (%; chance level: 50%).

		ZS21 (↑)		SC (↑)	
Model	**#Params**	sWuggy	sBLIMP	tSC	sSC
Ours					
GSLM2 (Libri-Light 60K h)	1.2B	79.3	71.8	84.5	57.6
GSLM2 (Conversational Speech 6M h)	6.5B	**80.7**	**72.9**	**90.5**	**63.2**
Open-weight speech LLMs					
Gemma 4 (Gemma Team, 2026)	4B	72.6	67.7	88.6	80.4
	12B	43.8	71.2	97.4	92.4
Qwen3-Omni (Xu et al., 2025b)	30B	84.0	**81.5**	97.8	92.1
Phi-4 (Abouelenin et al., 2025)	5.6B	66.3	66.9	79.9	70.6
Voxtral (Liu et al., 2025)	3B	79.1	73.8	84.8	79.7
	24B	81.1	79.9	**98.6**	**97.3**
MiniCPM-o 4.5 (Cui et al., 2026)	9B	**84.7**	78.4	97.3	93.1
Nemotron 3 Nano Omni (Deshmukh et al., 2026)	30B	75.8	64.9	55.5	54.8
Proprietary speech LLMs					
Gemini 2.5 Flash	–	68.7	80.8	99.4	98.3
Gemini 3.5 Flash	–	**86.2**	88.1	**99.8**	99.4
GPT-Realtime-2	–	83.5	**91.0**	98.8	**99.6**

across both orientations. Consequently, a model that always selects the same position regardless of configuration receives an accuracy of 50%.

For GSLM2, we retain the conventional scoring procedure. Therefore, the speech LLMs are scored differently, and these results should not be interpreted as a controlled comparison of their underlying speech modeling capabilities. Instead, the speech-LLM scores measure end-to-end task performance, jointly reflecting speech perception, linguistic reasoning, instruction following, and response generation. We report them as a practical reference for how frontier speech LLMs perform and to contextualize the extent to which GSLM2 remains competitive.

Speech LLMs. We evaluate the following open-weight checkpoints: Gemma 4 (4B[23] and 12B[24]), Qwen3-Omni[25], Phi-4[26], Voxtral (3B[27] and 24B[28]), MiniCPM-o 4.5[29], and Nemotron 3 Nano Omni[30]. We also evaluate the proprietary models via their respective APIs: Gemini 2.5 Flash[31], Gemini 3.5 Flash[32], and GPT-Realtime-2[33].

Results. Table 14 presents speech LLM results on the ZS21 and SC benchmarks, alongside GSLM2 results from the 60K-hour Libri-Light (Table 2) and 6M-hour conversational speech experiments (Figure 2). Among open-weight speech LLMs, performance varies substantially across models and benchmarks: MiniCPM-o 4.5 leads on sWuggy, Qwen3-Omni on sBLIMP, and Voxtral 24B on both SC tasks. Proprietary models achieve the highest scores across all four benchmarks, with all three exceeding 98% on both tSC and sSC. Given the scoring differences discussed above,

[23] https://huggingface.co/google/gemma-4-E4B-it
[24] https://huggingface.co/google/gemma-4-12B-it
[25] https://huggingface.co/Qwen/Qwen3-Omni-30B-A3B-Instruct
[26] https://huggingface.co/microsoft/Phi-4-multimodal-instruct
[27] https://huggingface.co/mistralai/Voxtral-Mini-3B-2507
[28] https://huggingface.co/mistralai/Voxtral-Small-24B-2507
[29] https://huggingface.co/openbmb/MiniCPM-o-4_5
[30] https://huggingface.co/nvidia/Nemotron-3-Nano-Omni-30B-A3B-Reasoning-BF16
[31] https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash
[32] https://ai.google.dev/gemini-api/docs/models/gemini-3.5-flash
[33] https://developers.openai.com/api/docs/models/gpt-realtime-2

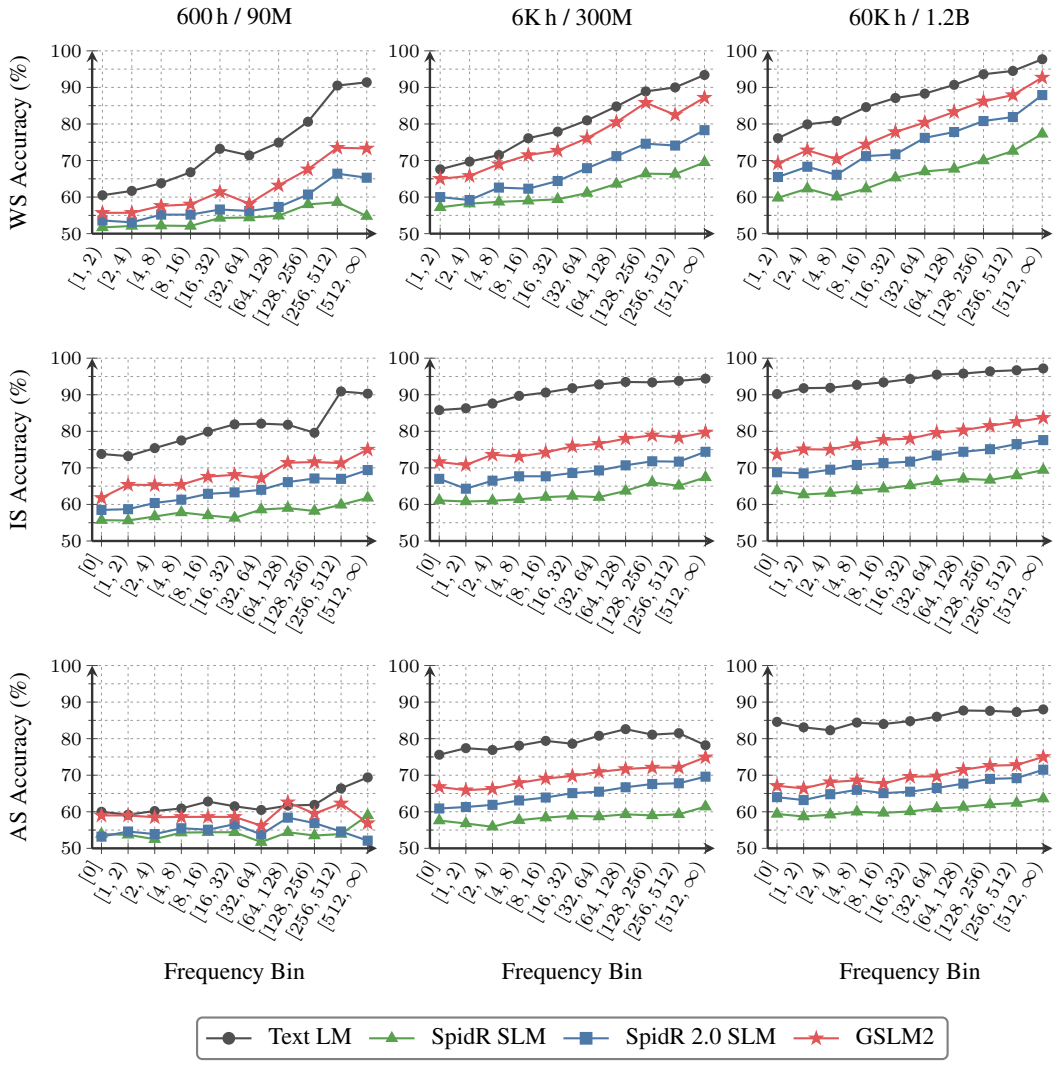


Figure 10: LT-Swap accuracy on WordSwap (WS), InflectionSwap (IS), and AgreementSwap (AS) by word-frequency bin at the three Libri-Light training scales. Chance level: 50%.

GSLM2 provides a practical reference: its ZS21 scores are comparable to those of several speech LLMs of similar size (e.g., Gemma 4, Phi-4, and Voxtral), while the strongest speech LLMs obtain higher SC scores, particularly on sSC. Near-saturation on SC should also be interpreted in light of frontier speech LLMs' substantially larger training corpora, including the text used for pretraining. These benchmarks remain informative at the speech-only scales studied here, but extending speech-text data-scaling studies to larger models and datasets will require more challenging evaluations.

C.1.9 LongTail-Swap Results by Word-Frequency Bin

To better understand the remaining LT-Swap gaps to Text LM in Table 2, we examine whether these gaps are concentrated on low-frequency words. Figure 10 presents WS, IS, and AS performance by word frequency across all three Libri-Light training-data scales. The aggregate SLM ranking largely holds within bins: SpidR 2.0 SLM generally outperforms SpidR SLM, while GSLM2 surpasses SpidR 2.0 SLM in every bin across all tasks and scales. Thus, the gains from multi-stream modeling extend across the frequency distribution.

On WS, accuracy generally increases with word frequency at all three scales. This association is weaker for SpidR SLM than for the other SLMs, particularly at 600 hours. At this scale, the gap to

Table 15: Speech-continuation performance at fixed decoding temperatures $\tau \in \{1.0, 0.8\}$, complementing the results with calibrated temperatures in Table 4. Oracle rows evaluate ground-truth continuations.

		ASR-PPL (↓)		**LLMJ** (↑)	
	Model	$\tau = 1.0$	$\tau = 0.8$	$\tau = 1.0$	$\tau = 0.8$
Oracle	Text LM	55.2		4.55	
	SpidR SLM	60.1		4.54	
	SpidR 2.0 SLM	70.1		4.41	
	GSLM2	59.5		4.53	
600 h / 90M	Text LM	229.1 ± 5.8	76.8 ± 2.4	2.08 ± 0.01	2.22 ± 0.02
	SpidR SLM	355.8 ± 9.3	232.6 ± 4.0	1.91 ± 0.01	2.01 ± 0.01
	SpidR 2.0 SLM	264.2 ± 6.0	163.4 ± 3.1	1.99 ± 0.02	2.10 ± 0.02
	GSLM2	244.9 ± 7.9	188.0 ± 4.2	2.02 ± 0.01	2.08 ± 0.03
6K h / 300M	Text LM	114.7 ± 3.1	45.3 ± 1.0	2.66 ± 0.05	2.87 ± 0.02
	SpidR SLM	143.2 ± 3.8	87.0 ± 2.4	2.81 ± 0.03	3.00 ± 0.03
	SpidR 2.0 SLM	141.9 ± 1.4	79.9 ± 2.7	2.74 ± 0.04	2.94 ± 0.03
	GSLM2	89.5 ± 2.4	69.0 ± 1.6	3.01 ± 0.04	3.01 ± 0.02
60K h / 1.2B	Text LM	66.2 ± 1.5	30.3 ± 0.5	3.27 ± 0.04	3.51 ± 0.02
	SpidR SLM	67.7 ± 1.4	45.8 ± 0.8	3.53 ± 0.02	3.65 ± 0.04
	SpidR 2.0 SLM	63.6 ± 1.3	40.7 ± 1.2	3.55 ± 0.03	3.66 ± 0.03
	GSLM2	51.1 ± 1.6	40.6 ± 0.8	3.60 ± 0.03	3.59 ± 0.02

Text LM generally widens with frequency for all three SLMs. At 6K and 60K hours, the gap varies less across frequency bins.

IS also shows a positive association between frequency and accuracy, but substantial gaps to Text LM persist throughout the distribution. The IS deficit highlighted in the main results therefore extends beyond low-frequency items.

AS exhibits a less consistent frequency trend at 600 hours, with SLM scores fluctuating across bins. At 6K and 60K hours, accuracy generally increases with frequency, but Text LM remains ahead in every bin. Its advantage over SLMs is consistently larger at 60K than at 6K hours, showing that the wider aggregate gap extends throughout the frequency distribution.

These results suggest that the limitations exposed by IS and AS are not confined to the lexical long tail. Although sBLIMP also evaluates syntax and includes similar inflection and agreement contrasts, its aggregate score combines a broader range of grammatical phenomena. Moreover, the cascaded ASR-LM evaluation in Appendix C.1.12 reveals TTS artifacts that can bias performance on several sBLIMP subtasks. Our spoken LT-Swap benchmark is less affected by these specific artifacts and contains substantially more sentence pairs on the specific categories. Its results therefore provide complementary evidence that inflectional and agreement judgments remain challenging for SLMs, even for frequent word items.

C.1.10 Generative Evaluation at Fixed Temperatures

The main results in Table 4 use model-specific temperatures calibrated to target $\mathrm{NRS} = 1$ on the development set (see Appendix B.4.4). Although this uses a common repetition target, temperature calibration is not standard practice in typical LM inference. Moreover, the resulting temperatures are relatively high, ranging from 1.20 to 1.59. We therefore complement the calibrated evaluation with results at $\tau \in \{1.0, 0.8\}$, two settings closer to conventional inference, in Table 15.

At both fixed temperatures, every model and data scale obtains lower ASR-PPL and higher LLMJ than when using calibrated temperatures. The differences can be substantial: at 60K hours, GSLM2 improves from an ASR-PPL of 189.1 and an LLMJ score of 3.05 at its calibrated temperature $\hat{\tau} =$

Table 16: Results for SLMs trained on 6K-hour English, French, and German data from VoxPopuli. Text LMs serve as toplines, which are trained using ASR transcriptions. Scores are accuracies (%; chance level: 50%).

Language	Model	sWuggy	sBLIMP	tSC
English	Text LM	77.0	63.9	76.8
	SpidR SLM	60.3	53.3	69.3
	SpidR 2.0 SLM	62.9	55.3	69.6
	GSLM2	**65.6**	**57.2**	**72.7**
French	Text LM	–	95.0	75.7
	SpidR SLM	–	58.8	63.9
	SpidR 2.0 SLM	–	66.0	70.2
	GSLM2	–	**69.4**	**72.0**
German	Text LM	–	88.9	74.8
	SpidR SLM	–	56.2	63.4
	SpidR 2.0 SLM	–	67.1	68.9
	GSLM2	–	**72.3**	**71.4**

1.59 to 51.1 and 3.60, respectively, at $\tau = 1.0$. At this temperature, Text LM performs best on both metrics in the 600-hour setting, while GSLM2 performs best at 6K and 60K hours, showing that its advantage at larger scales persists under a conventional decoding setup. At $\tau = 0.8$, Text LM achieves the lowest ASR-PPL at every scale, whereas the highest LLMJ score is obtained by the Text LM at 600 hours, GSLM2 at 6K hours, and SpidR 2.0 at 60K hours. Thus, the fixed-temperature results provide a practically relevant complement to the comparison using calibrated temperatures while also showing that model rankings depend on the decoding temperature.

We also report oracle results obtained by evaluating the ground-truth continuations. For Text LM, the ground-truth transcript is scored directly. For each SLM, in contrast, the ground-truth speech is converted into the corresponding unit representation, resynthesized with its unit vocoder, and transcribed by ASR before scoring. Consequently, the SLM oracles perform consistently worse than the Text LM oracle, with higher ASR-PPL and lower LLMJ scores. This gap captures degradation introduced by the multi-stage SLM evaluation pipeline, meaning that direct comparisons between SLMs and Text LM are not strictly equivalent. Oracle performance also varies across SLMs because their unit vocoders differ in resynthesis quality, although SpidR and GSLM2 achieve comparable oracle scores. Resynthesis artifacts increase ASR errors (see Appendix C.3), which propagate to both evaluation metrics.

A notable limitation of ASR-PPL appears at the lower fixed temperatures. At 60K hours and $\tau = 0.8$, every model obtains a lower ASR-PPL than its corresponding ground-truth oracle. Because the oracle contains the actual ground-truth continuations, these results cannot meaningfully indicate that the generated continuations are better than the ground truth, particularly as their LLMJ scores remain substantially below the oracle scores. Instead, they show that ASR-PPL can strongly favor the more predictable outputs produced by low-temperature decoding. ASR-PPL should therefore be interpreted together with semantic-quality and diversity measurements, even when evaluating models under conventional inference settings.

C.1.11 Multilingual evaluation on VoxPopuli

To examine how speech-data quality and language affect SLM performance, we train English, French, and German models on VoxPopuli. For each language, all SLM variants are trained on 6K hours of speech, using the same procedure as the 6K-hour Libri-Light models. The corresponding Text LM is trained on ASR transcripts of the same data (see Appendix B.1 for details). The English models are evaluated on the same benchmarks used in the Libri-Light experiments, whereas the French and German models are evaluated on our language-specific adaptations of sBLIMP and tSC (see Appendices B.4.1 and B.4.2).

Table 17: Performance of Text LMs and GSLM2 (Libri-Light 6K h; 300M parameters) under different training and evaluation conditions. Text LMs are trained on either ground-truth or ASR transcripts. In the cascaded setting, spoken stimuli are transcribed and the resulting transcripts are scored by the Text LM. GSLM2 is trained and evaluated directly on speech. Scores are accuracies (%; chance level: 50%).

			ZS21 (↑)		**SC** (↑)
Model	**Training Input**	**Evaluation Input**	(s)Wuggy	(s)BLIMP	tSC
Text LM	Ground-truth text	`B1`: Ground-truth text	79.5	66.8	84.8
		`B2`: ASR transcript of speech	70.6	71.0	75.4
	ASR transcripts	`B3`: Ground-truth text	81.2	66.9	85.1
		`B4`: ASR transcript of speech	71.5	69.1	77.4
GSLM2	Speech	Speech	76.3	67.2	78.9

Table 16 reports the results. Across all three languages, the SLMs exhibit the same ranking observed on Libri-Light: SpidR 2.0 consistently outperforms SpidR, while GSLM2 yields further gains. This consistency supports the generalizability of the SpidR 2.0 encoder and the multi-stream modeling approach across languages. Appendix C.2.6 further investigates the multilingual capabilities of the SpidR 2.0 encoder.

Compared with the corresponding 6K-hour Libri-Light results in Table 2, the English VoxPopuli setting exhibits a wider performance gap between GSLM2 and Text LM, particularly on sWuggy and sBLIMP. The gap increases from 3.2 to 11.4 percentage points on sWuggy and from -0.4 to 6.7 percentage points on sBLIMP, whereas the tSC gap changes only slightly, from 5.9 to 4.1 percentage points. Because sWuggy and sBLIMP may be particularly sensitive to the preservation of fine-grained phonetic information, the selective widening of these gaps suggests that noisier training speech can reduce data efficiency in spoken language modeling. We attribute this pattern to the more challenging recording conditions and greater acoustic heterogeneity of VoxPopuli relative to the comparatively clean Libri-Light corpus.

C.1.12 Cascaded ASR-LM Evaluation

We evaluate a cascaded ASR-LM approach to spoken language modeling, in which spoken evaluation stimuli are first transcribed by an ASR system and the resulting transcripts are scored by Text LM. We consider Text LM trained on either ground-truth or ASR-generated transcripts and compare them with GSLM2, which is trained and evaluated directly on speech. We use Whisper large-v3 as the ASR frontend. Table 17 reports results for the 6K-hour Libri-Light setting with 300M parameters. The results for Text LM trained and evaluated on ground-truth text (`B1`), as well as those for GSLM2, are identical to the corresponding results in Table 2. Evaluation on ASR transcripts (`B2` and `B4`) measures performance on spoken stimuli from sWuggy, sBLIMP, and tSC. Evaluation on ground-truth text (`B1` and `B3`) measures performance on the corresponding textual inputs: Wuggy and BLiMP items and ground-truth tSC transcripts built from StoryCloze.

For the Text LM trained on ground-truth text, replacing ground-truth evaluation text with ASR transcripts substantially degrades performance on sWuggy and tSC (`B1` vs. `B2`). On sWuggy, this is partly due to the difficulty of transcribing nonwords: Whisper may map an unfamiliar nonword to a plausible word or phrase, thereby changing the lexical contrast presented to the LM. On tSC, Whisper can produce different transcriptions of the first four sentences shared by the positive and negative stories when each full story is transcribed independently. These inconsistencies introduce score variation unrelated to the final continuation, making it harder for the LM to identify the more plausible story. In contrast, performance improves on sBLIMP when the same model is evaluated on ASR transcripts. We find that this improvement is partly driven by several artifacts introduced by the TTS system used to construct sBLIMP. For example, certain words in the ungrammatical distractors, such as "WHO," are retained in uppercase in the TTS input and pronounced letter by letter. The resulting unnatural pronunciations provide unintended cues associated with negative examples. When reflected in ASR transcripts, these cues can make the discrimination task easier,

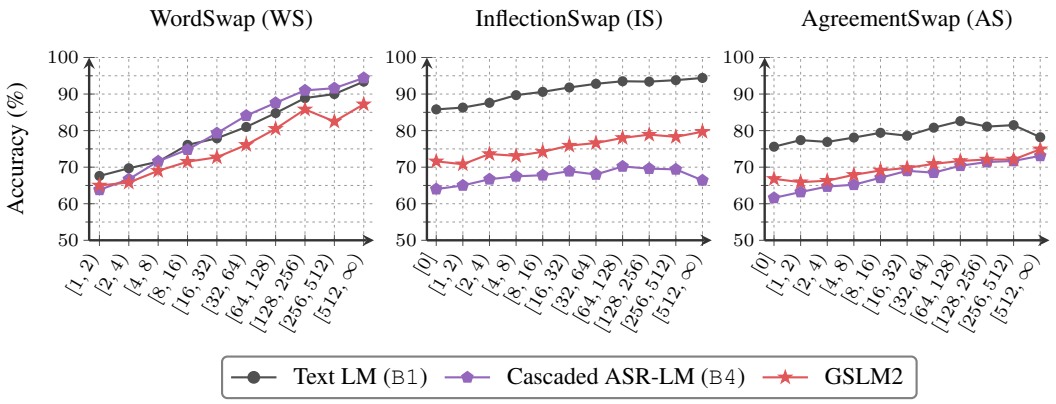


Figure 11: LT-Swap accuracy by word-frequency bin, comparing cascaded ASR-LM system (B4 in Table 17) with Text LM and GSLM2. Chance level: 50%.

independently of the grammatical contrast being tested. Such artifacts may also explain why GSLM2 outperforms Text LM on sBLIMP in Table 2.

Training Text LM on ASR transcripts yields a modest improvement on sWuggy, while performance on sBLIMP and tSC remains similar (B1 vs. B3). When the LM is trained on ASR transcripts and evaluated on ASR-transcribed speech (B4), the resulting system forms a fully cascaded ASR-LM pipeline, with ASR serving as the frontend to the LM. Compared with this cascaded system, GSLM2 performs substantially better on sWuggy and tSC. Figure 11 extends this comparison to LT-Swap, showing results for the cascaded system (B4), Text LM (B1), and GSLM2 across word-frequency bins. Average WS accuracy is similar for Text LM and the cascade (80.1% and 80.5%). On IS and AS, however, the cascade performs substantially worse than Text LM and is also outperformed by GSLM2. This degradation is partly attributable to Whisper altering the intended minimal contrasts, particularly in ungrammatical examples involving inflection or agreement. Together, these results suggest an advantage to modeling speech directly, as a direct SLM avoids errors and biases introduced by an intermediate ASR representation.

C.2 Supplementary Encoder Evaluation

This section presents supplementary experiments and analyses of SpidR 2.0's continuous representations and learned discrete units. We characterize codebook usage, unit duration, bitrate, and phone-code alignment (Appendices C.2.1 and C.2.2), and examine how regularization and training dynamics shape the compact code vocabularies of SpidR 2.0 (Appendices C.2.4 and C.2.5). We perform progressive ablations to assess the contributions of individual components (Appendix C.2.3) and compare learned codes with post-hoc k-means quantization (Appendix C.2.8). We also analyze similarities between streams from different encoder layers (Appendix C.2.9), compare a broad range of speech encoders on ABX and MAP (Appendix C.2.10), and examine how proxy metrics correlate with SLM performance (Appendix C.2.11). Finally, we extend the evaluation to a multilingual setup on DiscoPhon (Appendix C.2.6) and supervised probing tasks on SUPERB (Appendix C.2.7).

C.2.1 Codebook Metrics

For each encoder layer, we align the discrete codes with forced phone alignments on the ZeroSpeech'21 dev-clean set. Let P and C denote the corresponding frame-level phone and code random variables, respectively. We compute their empirical joint distribution $\mathbf{P} = [P(p, c)]$, where $P(p, c) = \Pr(\mathsf{P} = p, \mathsf{C} = c)$. The resulting unit assignments are characterized using six metrics in three groups:

- **Diversity.** Codebook perplexity is $\mathrm{PPL} = 2^{H(\mathsf{C})}$, where $H(\mathsf{C}) = -\sum_c P(c) \log_2 P(c)$ is the marginal entropy of code usage on this evaluation split, measured in bits, and $P(c) = \sum_p P(p, c)$. It measures the effective number of codes used and is upper-bounded by the codebook size (256 here).

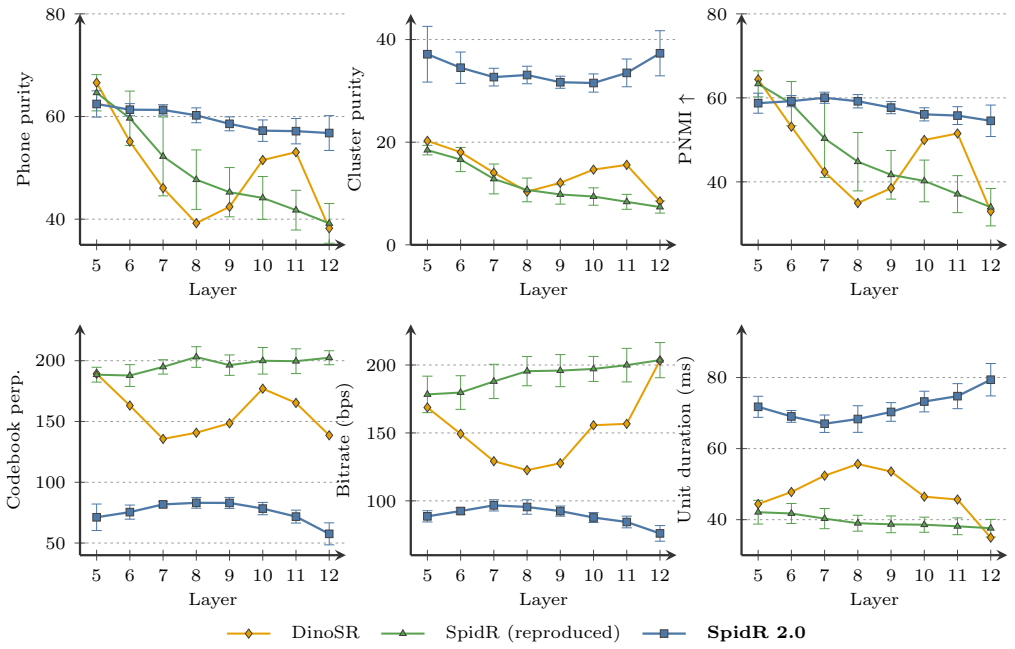


Figure 12: Layerwise codebook quality on ZeroSpeech'21 `dev-clean`. DinoSR uses the official released checkpoint; SpidR and SpidR 2.0 show mean ± standard deviation over eight training seeds. Purity and PNMI are in %.

- **Phone-code alignment (purity, PNMI).** Phone purity, $\sum_c \max_p P(p,c)$, is the fraction of frames whose phone label matches the most frequent phone associated with their assigned code (how consistently a code is assigned to a single phone). Cluster purity, $\sum_p \max_c P(p,c)$, is the fraction of frames assigned to the most frequent code for their phone (how strongly a phone is concentrated in a single code). Phone-normalized mutual information (PNMI), $I(\mathsf{P};\mathsf{C})/H(\mathsf{P})$, is the mutual information between phones and codes, normalized by phone entropy. It is important to note that all three metrics are heavily dependent on codebook usage: using more codes can increase phone purity and PNMI while reducing cluster purity. They should therefore be interpreted alongside perplexity.

- **Compression (bitrate, unit duration).** Both compression metrics are computed on the run-length deduplicated code sequence (consecutive identical codes collapsed to a single unit segment). Using a definition similar to that in Chang et al. (2025b), we compute bitrate as $(N/T)\,H_{\mathsf{dedup}}$ bits per second (bps), where N is the number of segments, T is the total duration in seconds, and H_{dedup} is the entropy, in bits, of the deduplicated code distribution. Unit duration (in ms) is the average span length (i.e., feature stride × total frames / total segments).

Observations. Figure 12 reveals three complementary trends. **(i) Diversity**: SpidR 2.0 uses less than half as many effective codes per layer as SpidR, matching the drop in active codes observed during training (see Figure 16). **(ii) Purity / PNMI**: Although SpidR keeps roughly constant codebook usage over layers, both purity metrics and PNMI degrade with depth. This resonates with the poorer performance on downstream metrics observed in Figure 3. SpidR 2.0 achieves substantially higher cluster purity at every layer, indicating that a larger fraction of frames map to the dominant code for their phone. Together with its lower codebook perplexity, this suggests a more compact representation. Although SpidR 2.0 has lower phone purity and PNMI than SpidR at their respective best layers, it outperforms SpidR on all other downstream metrics. **(iii) Compression**: SpidR 2.0 produces longer units (∼70 to 90 ms vs. ∼40 ms) at roughly half the bitrate. This is closer to mean phone duration in this dataset: 93 ms including silence and 82 ms excluding silence. Together, these trends indicate that SpidR 2.0 learns coarser units without sacrificing phone-level identifiability.

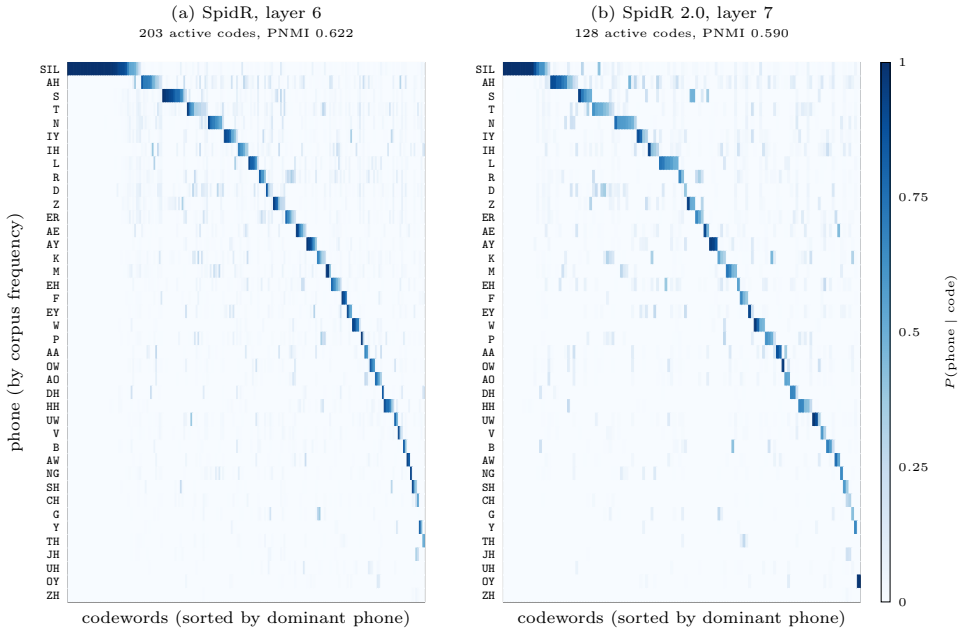


Figure 13: Phone content of the discrete units. Conditional distribution $P(\text{phone} \mid \text{code})$ on LibriSpeech `dev-clean`, computed using forced alignments at 10 ms and discrete codes at 20 ms. Rows are the 40 phone labels ordered by corpus frequency (`SIL` first); columns are the active codewords only (203 in (a) SpidR and 128 in (b) SpidR 2.0), grouped by their dominant phone in row order and sorted within each group by the conditional probability of that phone. Concentration on a single phone within each column indicates high phone purity; the column ordering arranges these peaks into a staircase pattern.

Table 18: ABX ablation results (%; lower is better), averaged over `dev-clean` and `dev-other` and over within- and across-speaker conditions. Modifications are cumulative. Values are mean ± standard deviation across eight seeds of encoder training. For each variant, l^* denotes the layer with the lowest task-averaged ABX error. $l = 5$ and $l = 10$ provide fixed-layer comparisons.

Variant	**Best layer** l^*	$l = l^*$	$l = 5$	$l = 10$
SpidR (reproduced)	6	5.16 ± 1.37	5.25 ± 0.76	10.90 ± 2.97
+ shared heads	6	4.82 ± 0.57	5.75 ± 0.74	6.78 ± 1.02
+ regularization	9	4.12 ± 0.34	8.75 ± 0.75	4.29 ± 0.42
+ normalizations	10	3.58 ± 0.31	5.88 ± 0.27	3.58 ± 0.31
+ LR $= 2.0 \times 10^{-4}$ (**SpidR 2.0**)	10	$\mathbf{2.88 \pm 0.12}$	$\mathbf{4.73 \pm 0.22}$	$\mathbf{2.88 \pm 0.12}$

C.2.2 Phone-Code Assignments

Figure 13 shows the conditional distribution $P(p \mid c)$ over phone labels p for each code c. The concentration of these distributions around dominant phones is consistent with the high phone purity of SpidR 2.0 reported in Figure 12. If we assign each code to its dominant phone, nine phones account for 50% of the total codebook usage in SpidR 2.0, compared with ten in SpidR. Codes assigned to silence constitute over 20% of SpidR's codebook, compared with 13% for SpidR 2.0, indicating that SpidR 2.0 allocates a smaller share of its codebook to silence.

C.2.3 Ablation Studies

The transition from SpidR to SpidR 2.0 consists of three successive modifications, followed by hyperparameter tuning. Throughout this process, we guide encoder improvements using ABX as our primary proxy for downstream SLM performance.

Table 19: MAP ablation results (%; higher is better). Modifications are cumulative. Values are mean ± standard deviation across eight seeds of encoder training. For each variant, all scores are reported at the layer with the highest average MAP score.

Variant	Best layer	Average	dev-clean	dev-other
SpidR (reproduced)	4	58.3 ± 2.5	65.2 ± 3.0	51.4 ± 3.9
+ shared heads	5	61.3 ± 3.4	66.2 ± 5.4	56.4 ± 4.1
+ regularization	7	68.0 ± 1.6	73.3 ± 2.2	62.7 ± 2.4
+ normalizations	8	$\mathbf{72.3 \pm 0.3}$	$\mathbf{77.0 \pm 0.3}$	$\mathbf{67.5 \pm 0.4}$
+ LR $= 2.0 \times 10^{-4}$ (**SpidR 2.0**)	7	69.2 ± 0.3	75.3 ± 0.4	63.1 ± 0.5

Table 20: Ablation results for downstream SLM metrics: sWuggy, sBLIMP, and tSC (%; higher is better). Modifications are cumulative. The SLM has 90M parameters and is trained on 6K hours of Libri-Light. Values are mean ± standard deviation across eight seeds of encoder training. For each variant, all scores are reported at the layer that maximizes average accuracy across the three metrics.

Variant	Best layer	Average	sWuggy	sBLIMP	tSC
SpidR (reproduced)	5	63.9 ± 0.6	66.4 ± 1.4	56.4 ± 1.0	68.9 ± 0.8
+ shared heads	5	58.3 ± 0.7	59.9 ± 1.4	51.6 ± 0.8	63.3 ± 1.3
+ regularization	7	66.7 ± 0.9	67.5 ± 2.3	60.4 ± 1.0	72.3 ± 0.9
+ normalizations	8	69.1 ± 0.7	70.1 ± 1.6	63.0 ± 1.2	74.1 ± 0.7
+ LR $= 2.0 \times 10^{-4}$ (**SpidR 2.0**)	7	$\mathbf{70.1 \pm 0.7}$	$\mathbf{72.1 \pm 1.2}$	$\mathbf{64.0 \pm 1.2}$	$\mathbf{74.2 \pm 1.3}$

Shared heads. As the main change resides in the tokenizer, we first replace the online codebooks on the teacher branch, updated through iterative clustering (Liu et al., 2023; Poli et al., 2025b), with the student's heads. Without regularization, this change admits stable collapsed solutions. This can be seen in Figure 14, where all phone-code alignment metrics (i.e., phone/code purity and PNMI) suddenly drop. However, continuous representations retain their performance on ABX and MAP (Tables 18 and 19, respectively), whereas discrete units perform worse on downstream SLM tasks (Table 20).

Regularization. We next introduce both regularization terms (i.e., Eq. (3)) using the baseline weights λ and μ and their associated schedules. Adding both of them jointly is necessary, since they are both heavily co-dependent: changing λ requires adjusting μ to keep similar codebook usage and prediction confidence. Not only does this improve performance, already surpassing the SpidR baseline on most metrics, but it also reduces the standard deviation of ABX and MAP across training runs. In addition, the resulting units tend to be longer and less diverse, with a codebook perplexity of 100 units and a mean duration of 55 ms, yielding a lower bitrate.

Normalizations. We then add layer normalization before each linear projection layer inside the shared heads and normalize the projection weights to unit norm during the forward pass. The projection weights are also initialized on the unit sphere. These changes further increase unit duration while slightly increasing codebook diversity. This yields better results in all metrics, with the largest improvements in MAP. Interestingly, the standard deviation of MAP also decreases by a factor of five, while ABX variance seems unaffected.

Hyperparameter tuning. Finally, we perform a small sweep over optimization hyperparameters. In our experiments, none of those yielded significantly beneficial changes but lowering the learning rate to 2.0×10^{-4}. With this setting, the prediction heads result in a more compact codebook and improve performance on ABX proxy and the target SLM metrics, at the cost of lower MAP.

C.2.4 Regularization

The objective of the entropy regularization terms (in Eq. (3)) is to prevent the shared codebooks from collapsing towards two stable but uninformative equilibria. The most classic case is codebook collapse: the student and teacher branches collapse onto a single arbitrary code regardless of the

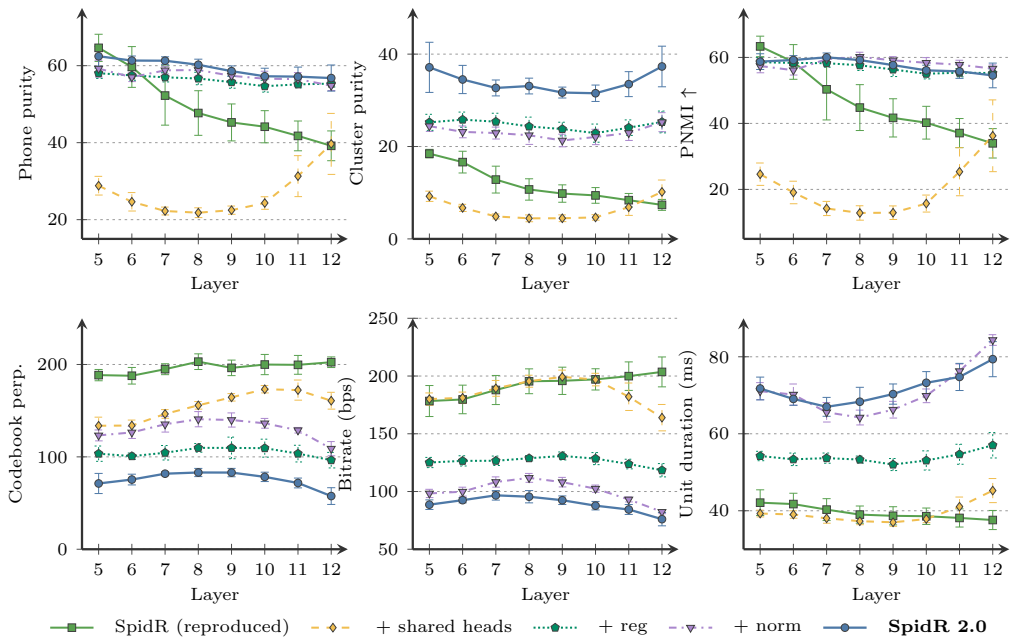


Figure 14: Layerwise codebook metrics for the cumulative ablations from SpidR to SpidR 2.0, evaluated on ZeroSpeech'21 `dev-clean`. Panels show the same metrics as Figure 12. Results are mean ± standard deviation across eight seeds.

input. To avoid this, we hypothesize that, over a batch of sequences, any informative sequence of discrete units uses a significant number of codes, though the appropriate level of diversity is task-dependent. Hence, we use the entropy of the batch-averaged probability distributions $H_c(\mathbb{E}_t[\mathbf{p}_{l,t}])$ (*marginal entropy* or ME) as a proxy to be maximized. Then, the heads in both student and teacher branches may converge to a near-uniform distribution over codes, regardless of the input. As a differentiable surrogate, we minimize the mean predictive entropy $\mathbb{E}_t[H_c(\mathbf{p}_{l,t})]$ (*conditional entropy* or CoE) to keep individual predictions confident.

Setup. In the default configuration, the marginal-entropy weight λ is set to 0.1 for the first 20k steps and then reduced to 0.001, while the conditional-entropy weight μ remains fixed at 0.1. We apply both terms to the teacher distribution only. We initially choose this configuration using ABX, MAP, prediction confidence, and codebook usage as proxies. We then conduct a sweep targeting development-set SLM performance and bitrate (defined in Appendix C.2.1). We compare the default (reported in Table 5) with four alternative configurations:

- **Teacher-only regularization** (`A` and `B`). We give λ and μ the same schedule: both remain fixed at 0.1 in setup `A`, or both decrease from 0.1 to 0.001 after 20k steps in setup `B`. We also tested constant ME regularization with stepped CoE regularization, but omit this configuration because it collapsed.
- **Teacher and student regularization** (`C` and `D`). Setup `C` additionally applies regularization to the student, using the baseline weights and schedules for both branches. Because the SSL objective tends to produce teacher and student entropies of similar magnitude and with similar dynamics, this approximately doubles the regularization contribution to the total loss. Setup `D` therefore applies regularization to both branches with halved baseline weights.

Results. Table 21 compares the above regularization configurations using ABX and MAP on continuous representations. We first notice that setup `B` suffers from a strong decrease in performance compared to the baseline. Though it reaches values of ABX and MAP in the same orders of magnitude as the baseline, setup `A` yielded larger codebook usage, as depicted in Figure 15. At layer 7, it uses all 256 active codes (vs. 128 for SpidR 2.0). Its units are also shorter on average (48 ms vs. 66 ms), resulting in a higher bitrate (163 bps vs. 97 bps).

Table 21: Effect of entropy regularization on ABX and MAP (%) for continuous representations from SpidR 2.0. All variants share the same base recipe as SpidR 2.0 and differ only in the regularization schedule/target. For each variant, ABX and MAP are reported at their respective best layers l^*, selected by task-averaged performance.

		ABX (↓)						**MAP** (↑)		
		Within		Across						
Variant	l^*	cln	oth	cln	oth	Avg.	l^*	cln	oth	Avg.
SpidR 2.0 (ME step, CoE constant)	10	2.33	2.74	2.56	3.70	**2.83**	7	75.6	63.6	**69.6**
A: ME constant, CoE constant	8	2.61	3.00	2.87	3.93	3.10	7	75.4	63.3	69.4
B: ME step, CoE step	10	2.57	3.18	2.79	4.10	3.16	7	72.1	60.0	66.0
C: Baseline + student reg. (1×)	9	2.36	2.77	2.58	3.73	2.86	6	75.2	62.2	68.7
D: Baseline + student reg. (0.5×)	10	2.37	2.82	2.61	3.91	2.93	7	74.7	62.3	68.5

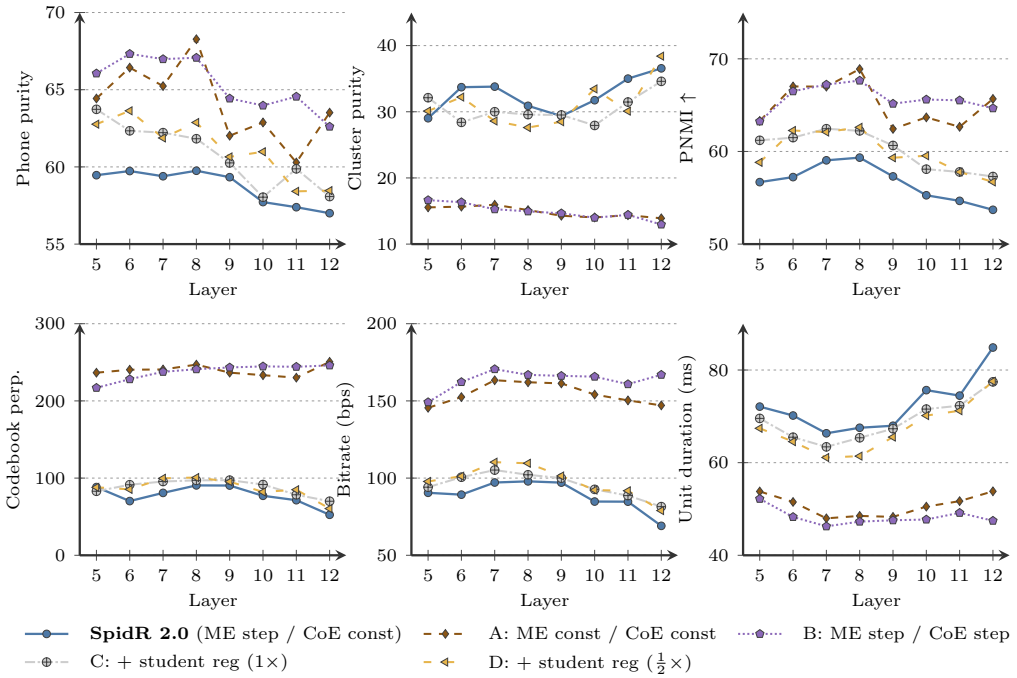


Figure 15: Layerwise codebook metrics on ZeroSpeech'21 `dev-clean` for the regularization setups in Appendix C.2.4. Panels show the same metrics as Figure 12. All runs use the SpidR 2.0 recipe and seed 0, differing only in regularization. The baseline reduces the ME weight λ from 0.1 to 0.001 after 20k steps and keeps the CoE weight $\mu = 0.1$. `A` keeps both weights at 0.1; `B` reduces both to 0.001. `C` and `D` regularize both teacher and student using full and half baseline weights, respectively. Purity and PNMI are in %.

Setups `C` and `D` show small differences to baseline, around the standard deviation variations. Note that, because these experiments are limited to a single seed due to time constraints, we cannot establish whether these differences are robust across training runs.

Importantly, these results also inform the broader design choices in GSLM2. Our regularization settings prioritize MAP, bitrate, and unit duration as proxies for downstream SLM performance (see Appendix C.2.11). Other applications may favor shorter units and greater codebook usage, which can be encouraged by adjusting the regularization weights. Setup `A` illustrates this trade-off: it achieves nearly maximal codebook usage and unit durations of 45-55 ms, with only a small reduction in MAP.

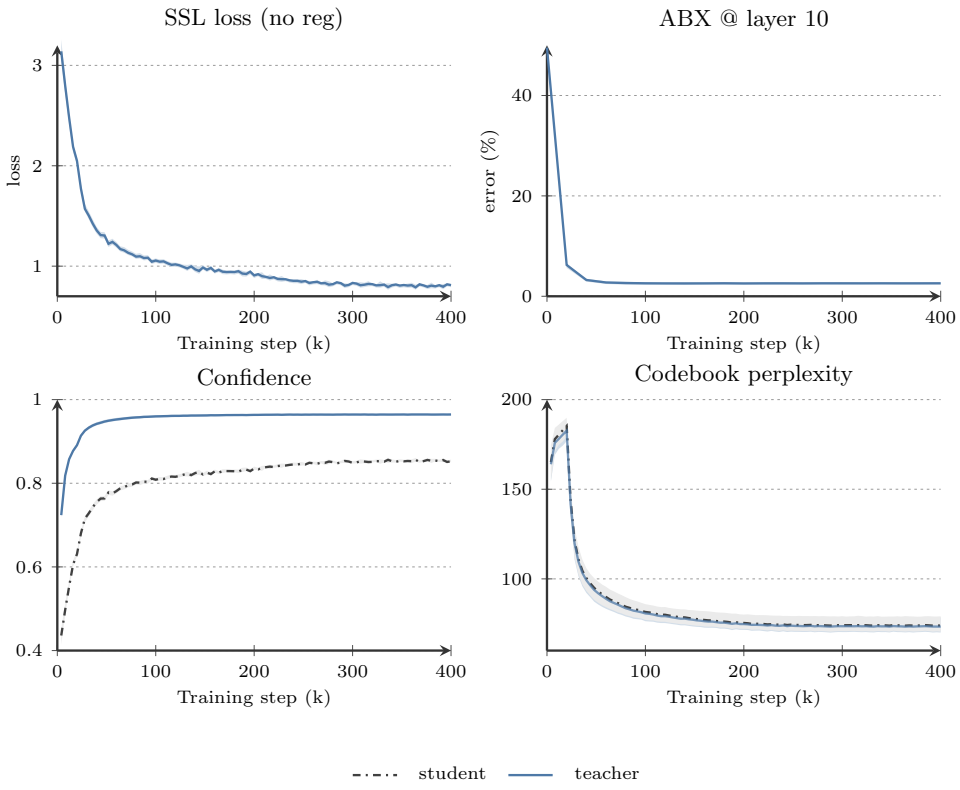


Figure 16: Training dynamics of SpidR 2.0. We report the mean (line), as well as the minimum and maximum for each training step (shaded area) across eight seeds. *Top row:* SSL loss and task-averaged ABX error at layer 10. *Bottom row:* predictive confidence, $\mathbb{E}_{l,t}[\max_c \mathbf{p}_{l,t}[c]]$, and codebook perplexity, $\mathbb{E}_l[2^{H_c(\mathbb{E}_t[\mathbf{p}_{l,t}])}]$, for the teacher (solid) and student (dash-dotted).

C.2.5 Training Curves

To facilitate reproducibility, Figure 16 shows the SpidR 2.0 training curves aggregated over eight seeds. The SSL loss (top left) decreases smoothly throughout training. ABX at layer 10 (the best-performing ABX layer at the final step) decreases exponentially through the first half of training, and plateaus at around 200k steps. In our experiments, while ABX and other proxy metrics plateau at this point, downstream SLM performance continues to improve up to 400k steps.

The confidence trajectories (bottom left) show an interesting phenomenon: the student and teacher branches sharpen their assignment distributions at different rates. While the teacher confidence rises quickly to $\sim$0.96, an almost certain confidence, the student confidence climbs gradually to $\sim$0.85. The persistent gap of about 0.11 reflects the gap in available context between the teacher and the student, where, under the masking policy, the student sees only 50% of the frames in each training sequence.

Codebook perplexity (bottom right) makes the ME regularization schedule visible: it grows for 20k steps when $\lambda = 0.1$, then drops sharply when λ is reduced. Perplexity subsequently declines more gradually, reaching approximately 73 by the end of training, which gives a total of 129 active codes on the development sets (cf. Figure 12). The continued decline in perplexity during the second half of training indicates that the model progressively commits to a compact unit set after the diversity-enforcing pressure is relaxed, while downstream SLM performance continues to improve.

Table 22: DiscoPhon cross-lingual evaluation. All metrics are reported at each model's best-PNMI layer, selected by averaging PNMI across dev and test languages. Scores are averaged over languages within each split and expressed in %. ABX is computed on continuous triphone representations and averaged over within-/across-speaker conditions. Dev languages: German, Swahili, Tamil, Thai, Turkish, Ukrainian. Test languages: Basque, English, French, Japanese, Mandarin, Wolof.

Model	**Layer**	**F1** (↑)	**R-val** (↑)	**PER** (↓)	**ABX** (↓)	**PNMI** (↑)
Dev languages						
DinoSR (LibriSpeech)	5	61.1	48.9	82.4	10.8	49.0
SpidR (LibriSpeech)	5	62.5	50.4	79.1	11.8	52.8
SpidR (VP19)	5	63.0	44.8	82.8	**8.7**	**55.5**
SpidR 2.0 (LibriSpeech)	6	58.8	61.6	65.2	9.1	48.6
SpidR 2.0 (VP19)	6	**64.3**	**65.3**	**61.1**	8.9	50.0
Test languages						
DinoSR (LibriSpeech)	5	61.4	45.6	83.4	8.4	51.3
SpidR (LibriSpeech)	5	**64.2**	45.2	80.8	9.5	56.2
SpidR (VP19)	5	63.4	36.9	88.6	**6.4**	**58.1**
SpidR 2.0 (LibriSpeech)	6	59.6	58.6	66.7	7.1	51.3
SpidR 2.0 (VP19)	6	63.7	**60.5**	**65.4**	6.8	51.6

C.2.6 DISCOPHON

We report zero-shot results for SpidR 2.0 on DiscoPhon (Poli et al., 2026), a multilingual benchmark evaluating unsupervised phoneme discovery across 12 languages. Its tasks assess complementary properties of speech representation:

- *Unit quality*, measured by PNMI, captures how much information discrete units retain about reference phone identities.
- *Phone recognition*, measured by phone error rate (PER), evaluates recovery of the phone sequence through substitution, deletion, and insertion errors, independently of precise boundary timing.
- *Phone segmentation* evaluates boundary placement: F1 summarizes boundary precision and recall, while R-value penalizes over-segmentation more strongly.
- *ABX* measures how well continuous representations distinguish phonemes. We average within- and across-speaker ABX errors.

For recognition and segmentation, we use a many-to-one mapping that assigns each codeword to the reference phone with which it most frequently co-occurs, allowing multiple codewords to represent the same phone. We evaluate the same encoders reported in Table 5, all trained on LibriSpeech. We also train SpidR and SpidR 2.0 on a VoxPopuli subset comprising 300 hours of speech from each of 19 languages (VP19), extracted using the VoxPopuli procedure described in Appendix B.1.

Table 22 reports the DiscoPhon results. Consistent with the results in Figure 12, SpidR 2.0 yields lower PNMI than SpidR across the dev and test language splits for both pretraining datasets. In addition, it is important to note that we are not reporting the results coming from the best ABX layer, but rather best PNMI, as it is our closest discrete phoneme mapping similarity evaluation metric. This choice has little effect on SpidR's ABX scores, as its phonetic information is concentrated in layers 5-6. For SpidR 2.0 trained on VP19, however, ABX errors decrease to 7.6% and 5.8% on dev and test languages at layer 10, compared with 8.9% and 6.8% at the reported layer 6.

We then notice that SpidR 2.0 substantially improves phone recognition over SpidR, even when pretrained exclusively on English speech from LibriSpeech. Multilingual pretraining on VP19 further reduces its PER. The segmentation results depend on the metric. Relative to SpidR, SpidR 2.0 obtains lower F1 scores with LibriSpeech pretraining and slightly higher scores with VP19 pretraining. R-value, however, improves consistently, by 11.2-13.4 absolute points with LS pretraining and 20.5-23.6 points with VP19 pretraining. Since R-value penalizes over-segmentation more strongly than

Table 23: Downstream evaluation on SUPERB. For DinoSR, SpidR, and SpidR 2.0, we evaluate five learning rates for each trained downstream probe and report the test score of the run with the best development-set performance. For WavLM, we report the scores from the official SUPERB leaderboard. ASR reports word error rate without LM decoding; PR reports phone error rate. KS, IC, and ER report accuracy; SF reports slot-type F1 and slot-value character error rate (CER). QbE reports maximum term weighted value, scaled by 100.

	Recognition		**Detection**		**Semantics**			**Paraling.**
Model	ASR (↓)	PR (↓)	KS (↑)	QbE (↑)	IC (↑)	SF-F1 (↑)	SF-CER (↓)	ER (↑)
DinoSR	**4.78**	**3.66**	96.17	4.99	**98.63**	88.85	23.66	**66.15**
SpidR	7.23	5.17	95.39	8.02	95.28	86.92	27.54	65.52
SpidR 2.0	6.61	4.34	96.27	7.68	97.94	86.29	27.29	64.03
WavLM (BASE)	6.21	4.84	**96.79**	**8.70**	**98.63**	**89.38**	**22.86**	65.94
WavLM LARGE	3.44	3.06	97.86	8.86	99.31	92.21	18.36	70.62
SpidR 2.0 LARGE	4.71	2.72	97.05	8.84	98.68	88.90	22.55	65.28

F1, these gains suggest reduced over-segmentation. It also agrees with the longer units produced by SpidR 2.0, whose durations more closely match typical phone durations (see Appendix C.2.1).

C.2.7 SUPERB

To complement our zero-shot encoder evaluations in Table 5, we report downstream results on SUPERB (Yang et al., 2021), excluding speaker-related tasks. The evaluation covers ASR, phone recognition (PR), keyword spotting (KS), spoken-query detection (QbE), intent classification (IC), slot filling (SF), and emotion recognition (ER). For each supervised probe, we train a task-specific predictor and learn layer-mixing weights while keeping the encoder frozen.

Table 23 summarizes the results. Among DinoSR, SpidR, and SpidR 2.0, DinoSR achieves the best recognition, semantic, and ER scores, while SpidR and SpidR 2.0 lead on QbE and KS, respectively. This pattern is consistent with the divergence between spoken language modeling and supervised downstream performance reported in Poli et al. (2025b). Nevertheless, SpidR 2.0 outperforms SpidR on ASR and PR while also improving SLM performance. The reversal in rankings between DiscoPhon's zero-shot evaluation and SUPERB's supervised probing is particularly evident for PR. On the English subset of DiscoPhon, DinoSR obtains 45.0% PER, compared with 35.6% for SpidR and 35.0% for SpidR 2.0, with all three models pretrained exclusively on LibriSpeech. On SUPERB, DinoSR instead achieves the lowest PER of these models at 3.66%, compared with 5.17% and 4.34% for SpidR and SpidR 2.0, respectively.

These results suggest that the quality of direct code-to-phone assignments does not necessarily predict performance with a supervised probe. However, differences in data and evaluation protocols prevent this comparison from isolating the cause of the reversal. One possible explanation is that SpidR's intermediate prediction heads shape the representation geometry around the code-prediction objective. This specialization may make phonetic information readily accessible through the pretrained projections while limiting the flexibility of the learned features for newly trained downstream probes.

Scaling SpidR 2.0 to the LARGE scale following Hsu et al. (2021) improves all reported SUPERB metrics relative to its BASE model. WavLM remains stronger on most directly comparable tasks at both scales, but SpidR 2.0 achieves lower PER: 4.34% versus 4.84% at BASE, and 2.72% versus 3.06% at LARGE. The latter is the lowest PER among the models reported in Table 23.

Figure 17 illustrates the learned softmax weights used to combine encoder layers for the downstream probes. For DinoSR, the mixtures concentrate on layer 11 for ASR, PR, and SF, coinciding with its peak MAP scores. SpidR places the greatest weight on layer 5, with substantial weights also assigned to layers 4 and 6 for PR. SpidR 2.0 exhibits a more even distribution across layers: the largest weight exceeds 20% on only two tasks (i.e., PR and IC). This broader distribution motivates combining representations from multiple layers in GSLM2.

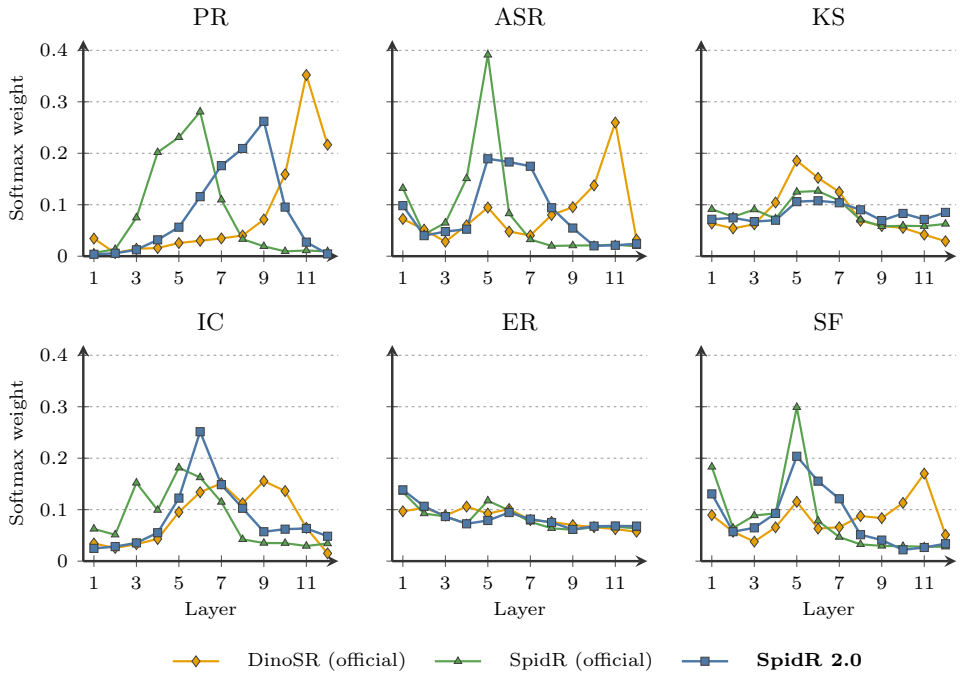


Figure 17: Learned layer-mixing weights for the SUPERB probes. Softmax-normalized weights combine intermediate encoder representations into the features used by each downstream probe.

Table 24: Discrete ABX and SLM results for SpidR and SpidR 2.0. *Codes* denotes units obtained from the prediction heads; *k-means* denotes clustering of continuous encoder representations. For learned codes, k denotes the number of active codes; for k-means, it denotes the number of clusters. k-means is fitted on LibriSpeech `train-clean-100`, except for LS-960, which uses the full 960-hour corpus. Results are reported at the layer with the highest average development score across sWuggy, sBLIMP, and tSC. ABX errors are averaged over `dev-clean` and `dev-other` and over within- and across-speaker conditions.

Model	**Unit type**	**Layer**	k	**ABX** (↓)	**sWuggy** (↑)	**sBLIMP** (↑)	**tSC** (↑)
SpidR	k-means	6	256	7.20	71.0	59.1	70.2
	Codes	6	203	6.31	69.8	58.1	70.1
SpidR 2.0	k-means	8	256	6.63	72.5	61.6	73.6
	k-means	8	129	6.15	71.8	62.7	73.7
	k-means (LS-960)	7	128	6.75	72.4	62.6	74.1
	Codes	7	128	**5.56**	**73.2**	**65.6**	**75.8**

C.2.8 Discrete Unit Quality

We compare discrete unit quality across encoders, quantization methods, and vocabulary sizes. For SpidR 2.0, we apply k-means to the representations from every layer under two settings: a fixed vocabulary of $k = 256$, matching the total codebook capacity, and a vocabulary size set separately for each layer to match the number of learned codes activated on the ZeroSpeech'21 `dev-clean` set. Table 24 compares these configurations with the learned codes from SpidR 2.0 and with both k-means units and learned codes from SpidR. We evaluate ABX computed on the discrete units and SLM performance with the 90M-parameter model trained on 6K-hour Libri-Light. For each configuration, we select the layer that maximizes the mean accuracy across the development splits of sWuggy, sBLIMP, and tSC.

With a fixed vocabulary of $k = 256$, SpidR 2.0 outperforms SpidR on discrete ABX and all three SLM metrics. These gains extend to an external quantizer, suggesting that the improvements arise

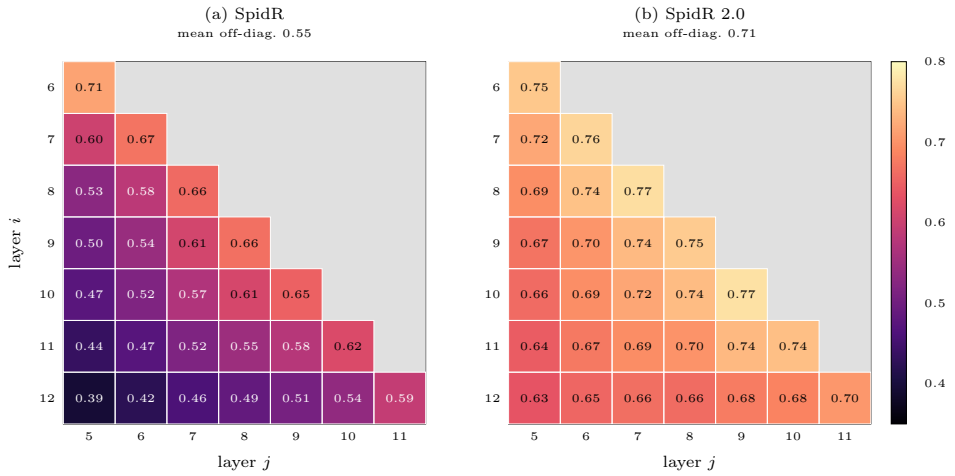


Figure 18: Cross-layer unit-stream similarity. Pairwise NMI between unit streams extracted from encoder layers 5-12 of (a) SpidR and (b) SpidR 2.0, computed on LibriSpeech `dev-clean` and averaged over eight encoder training seeds. Since the matrices are symmetric, only entries below the diagonal are displayed; the diagonal (NMI $= 1$) and upper triangle are greyed out.

partly from the encoder representations themselves. Reducing k to match the number of active learned codes lowers SpidR 2.0's discrete ABX error from 6.63% to 6.15%, while leaving mean SLM performance nearly unchanged (69.2% vs. 69.4%). The smaller vocabulary appears to improve sBLIMP but reduces sWuggy, indicating different effects on syntactic and lexical evaluation. Fitting k-means on the full LibriSpeech dataset (LS-960) achieves slightly higher mean SLM performance (69.7% vs. 69.4%) but higher ABX error, compared to the default setup using `train-clean-100`. However, ABX errors here must be put into context, as the layer with best SLM development scores is not necessarily the best layer for continuous ABX, as shown in Figure 3.

At layer 7 with 128 active units, SpidR 2.0's learned codes outperform the LS-960 k-means configuration on discrete ABX and all three SLM metrics, despite using the same encoder representations, effective vocabulary size, and training corpus. Thus, for SpidR 2.0, learning the quantization module during encoder pretraining both avoids a post-hoc clustering stage and yields better downstream performance. We attribute this to the gradients of $\mathcal{L}_{\text{ssl}}$ backpropagated through the prediction heads. The codebooks' centroids $\mathbf{c}_{l,i}$ are then not only trained to discretize the latent space with coverage constraints as in the k-means algorithm, but also to encourage code assignments to be predictable from masked context in the student branch.

C.2.9 Stream Similarities

To help interpret the benefits of multi-stream SLM, we examine the similarity between unit streams from different encoder layers. Figure 18 reports pairwise normalized mutual information (NMI) across layers 5-12 on LibriSpeech `dev-clean`, averaged over eight encoder-training seeds. Mutual information is normalized by the arithmetic mean of the two marginal entropies. Adjacent layers generally exhibit higher similarity than more distant layers, and SpidR 2.0 has a higher mean off-diagonal NMI than SpidR (0.71 vs. 0.55). NMI quantifies statistical overlap, but does not reveal whether the differences between streams are useful for language modeling. The layer-diversity ablation (Appendix C.1.7) provides evidence of their practical value. These findings support the value of combining representations from different layers despite substantial statistical overlap.

C.2.10 Comparative Evaluation of Speech Encoders

Table 25 compares triphone ABX and word MAP scores across a broad range of speech encoders, extending the results reported in Table 5. We group the models into five categories: self-supervised encoders, self-supervised encoders with additional robustness objectives, ASR-supervised encoders, audio encoders from speech-capable LLMs, and neural codecs. Because both metrics evaluate con-

Table 25: ABX and MAP (%) for SpidR 2.0 and publicly available speech encoders on the ZeroSpeech'21 development set. Each metric is reported at the encoder layer l that achieves its best average score. ABX is evaluated within and across speakers; *cln*/*oth* denote the clean/other subsets. Rows are ordered by decreasing average MAP within each group. In the first model group, bold and underlined scores indicate the best and second-best averages for each metric, respectively.

			ABX (↓)						**MAP** (↑)		
			Within		Across						
Model	**Params** [M]	l	cln	oth	cln	oth	Avg.	l	cln	oth	Avg.
Self-supervised (masked / predictive)											
data2vec 2.0 Large (Baevski et al., 2023)	313	2	6.20	5.92	6.92	7.75	6.70	12	76.1	72.6	**74.4**
data2vec (Baevski et al., 2022)	93	4	4.41	5.51	5.12	7.40	5.61	10	72.5	66.9	<u>69.7</u>
SpidR 2.0	93	10	2.33	2.74	2.56	3.70	**2.83**	7	75.6	63.6	69.6
DinoSR (Liu et al., 2023)	93	5	4.05	5.11	4.74	7.30	5.30	11	74.9	64.1	69.5
MR-HuBERT mono-large 50 Hz (Shi et al., 2024a)	320	1	3.41	3.73	3.93	5.11	4.04	1	73.0	64.2	68.6
MR-HuBERT mono-base 50 Hz (Shi et al., 2024a)	97	1	3.63	3.99	4.21	5.40	4.31	1	70.7	63.1	66.9
USAD (Chang et al., 2025a)	97	10	3.55	4.19	4.08	5.77	4.40	9	72.5	61.2	66.8
data2vec 2.0 (Baevski et al., 2023)	93	1	5.13	5.77	5.71	7.57	6.04	5	70.4	62.9	66.7
MR-HuBERT mono-base 25 Hz (Shi et al., 2024a)	97	1	3.48	3.85	4.01	5.14	4.12	1	68.5	61.6	65.0
HuBERT Large (Hsu et al., 2021)	315	24	3.90	4.30	4.55	5.86	4.65	24	69.0	59.1	64.0
SpidR (Poli et al., 2025b)	93	6	3.32	3.74	3.66	4.95	<u>3.92</u>	5	69.5	57.3	63.4
data2vec Large (Baevski et al., 2022)	313	7	4.51	5.46	5.24	7.18	5.60	10	64.2	56.3	60.2
HuBERT XL (Hsu et al., 2021)	962	48	4.04	4.38	4.80	6.16	4.84	48	64.6	54.6	59.6
XEUS (Chen et al., 2024)	574	16	3.32	4.20	4.19	6.32	4.51	18	61.8	52.4	57.1
MR-HuBERT mono-large 25 Hz (Shi et al., 2024a)	320	1	4.11	4.27	4.74	5.64	4.69	1	56.5	49.9	53.2
HuBERT (Hsu et al., 2021)	94	11	3.38	4.26	4.04	6.45	4.53	10	58.9	47.2	53.1
wav2vec 2.0 Large (Baevski et al., 2020)	315	16	4.34	5.14	5.22	7.27	5.49	13	52.8	42.1	47.4
wav2vec 2.0 (Baevski et al., 2020)	94	6	4.51	5.67	5.27	7.83	5.82	7	49.8	38.6	44.2
NEST-L (Huang et al., 2025)	115	10	5.16	5.70	5.77	7.31	5.98	10	43.7	39.1	41.4
w2v-BERT 2.0 (Barrault et al., 2023)	580	17	5.52	7.37	8.64	14.21	8.94	11	16.0	8.9	12.5
SSL + finetuning robustness / speaker-invariance loss											
WavLM + Spin2048 (Chang et al., 2023)	94	12	3.09	3.48	3.52	4.45	3.63	11	77.0	68.7	72.9
WavLM Base+ (Chen et al., 2022)	94	12	3.54	4.08	4.10	5.83	4.39	10	72.9	63.7	68.3
WavLM Base (Chen et al., 2022)	94	11	3.01	3.70	3.47	5.23	3.85	10	73.6	62.1	67.9
HuBERT + Spin2048 (Chang et al., 2023)	94	12	2.69	3.23	3.05	4.07	3.26	11	70.8	62.8	66.8
ContentVec100 (Qian et al., 2022)	94	12	3.30	4.04	3.83	5.49	4.17	11	69.6	58.9	64.2
WavLM Large (Chen et al., 2022)	315	24	3.94	4.32	4.65	5.98	4.72	24	67.6	57.5	62.5
Supervised (ASR)											
wav2vec 2.0 Large (ASR) (Baevski et al., 2020)	315	17	3.91	4.14	4.53	5.21	4.45	17	76.0	69.8	72.9
HuBERT Large (ASR) (Hsu et al., 2021)	315	14	4.69	5.38	5.48	7.25	5.70	22	75.6	70.1	72.8
OWSM v3.1 (Peng et al., 2024)	29	4	3.50	4.39	4.10	5.98	4.49	5	72.9	61.2	67.0
wav2vec 2.0 (ASR) (Baevski et al., 2020)	94	6	3.94	4.90	4.56	6.49	4.97	9	70.8	62.7	66.8
SenseVoice (An et al., 2024)	158	32	5.37	6.29	6.27	8.36	6.57	50	63.2	54.5	58.8
Moonshine (Jeffries et al., 2024)	20	6	3.62	4.75	4.22	6.61	4.80	7	65.1	49.1	57.1
FastConformer (Rekesh et al., 2023)	109	9	2.92	3.63	3.21	4.68	3.61	11	52.9	49.9	51.4
Canary (Puvvada et al., 2024)	609	10	3.46	4.05	3.91	5.04	4.12	11	50.3	47.8	49.1
Whisper-large-v3 (Radford et al., 2023)	637	26	6.35	6.99	7.51	9.98	7.71	27	40.6	31.6	36.1
Whisper small.en (Radford et al., 2023)	88	8	7.01	8.28	8.28	11.75	8.83	11	31.1	24.3	27.7
Speech-LLM audio encoders											
Kimi-Audio (KimiTeam, 2025)	637	26	6.13	6.81	7.37	9.71	7.50	32	56.8	50.8	53.8
Audio Flamingo 3 (Ghosh et al., 2026)	637	25	6.50	7.69	7.96	10.78	8.23	27	51.2	40.9	46.0
Qwen2.5-Omni (Xu et al., 2025a)	640	25	5.54	6.52	6.27	8.93	6.81	27	47.9	35.8	41.8
Voxtral (Liu et al., 2025)	637	26	6.35	7.00	7.51	9.98	7.71	27	40.6	31.6	36.1
Neural codecs & speech tokenizers (pre-quantizer latent)											
Mimi (Défossez et al., 2024)	39	1	4.78	5.91	5.67	8.47	6.21	1	36.5	26.9	31.7
CosyVoice 2 (Du et al., 2024)	124	1	6.20	8.02	8.45	12.97	8.91	1	15.7	9.9	12.8

tinuous intermediate representations, we exclude quantizers, prediction heads, and decoders. We report encoder parameter counts to contextualize the comparison and, where available, compare our measurements with previously reported scores to identify potential evaluation discrepancies.

SpidR 2.0 achieves the lowest average ABX error among all evaluated encoders (2.83%). Four models exceed its word MAP score: data2vec 2.0 Large, wav2vec 2.0 Large ASR, HuBERT Large ASR, and WavLM + Spin2048. Its MAP score (69.6%) is close to those of data2vec (69.7%) and DinoSR (69.5%); we interpret these small differences cautiously given the observed variation across encoder training runs. Larger models and ASR supervision can yield higher word MAP, but their effects on ABX are mixed.

C.2.11 Correlations Between Proxy and Target Metrics

We report Pearson correlation coefficients and corresponding p-values between six proxy metrics (continuous ABX, MAP, phone purity, PNMI, unit duration, and bitrate) and three target SLM metrics (sWuggy, sBLIMP, and tSC) in two settings. Figure 19 presents correlations across model variants from the progressive development of SpidR into SpidR 2.0, as described in Appendix C.2.3. Each point represents a model-layer pair, with layers ranging from 5 to 12 and scores averaged over eight encoder training seeds. Figure 20 displays correlations across layers and seeds within SpidR 2.0. For the latter analysis, we use a nominal sample size of 64.

Across model variants, MAP and bitrate show the strongest correlations with SLM performance, whereas ABX has the highest p-values. Within SpidR 2.0, tSC shows the weakest associations with the proxies, with R^2 exceeding 0.10 only for ABX. The ranking of proxies by the p-values of their correlations with the average SLM score remains unchanged between the two settings. The correlations for ABX, bitrate, and unit duration nevertheless reverse sign between settings. Across model variants, longer units (lower bitrate) and lower ABX error rates are associated with better SLM performance. Within SpidR 2.0, earlier layers generally have higher ABX error rates and produce shorter units, yet achieve better sWuggy and sBLIMP scores. This pattern results in positive correlations between averaged SLM metrics and both bitrate and ABX error rate.

Overall, our findings support reporting word MAP as a primary proxy rather than ABX (Poli et al., 2025b) or PNMI (Chang et al., 2025b).

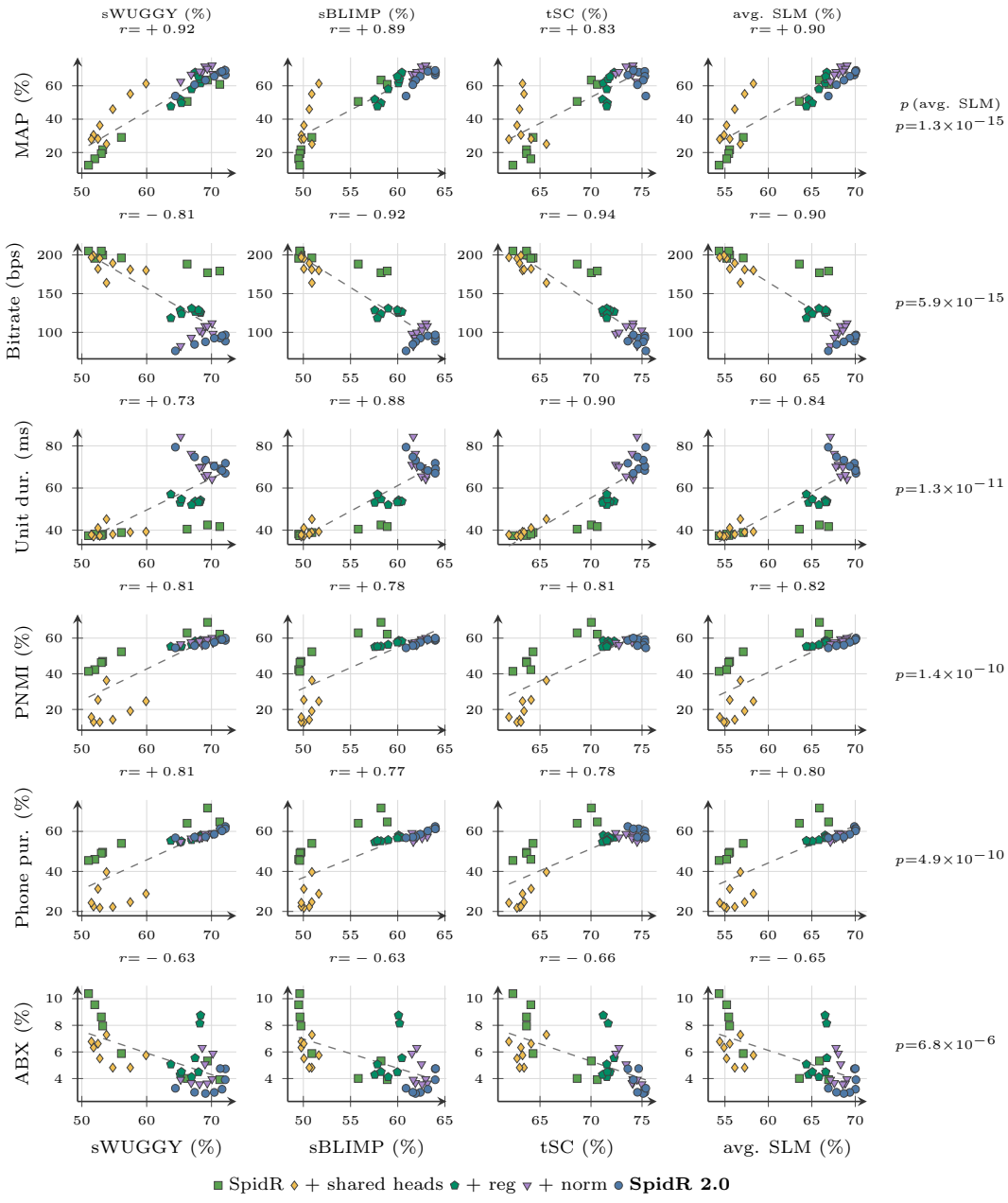


Figure 19: Pearson correlations between proxy metrics (rows) and target SLM metrics (columns: sWuggy, sBLIMP, tSC, and their average, denoted avg. SLM) *across* the progressive-ablation models from SpidR to SpidR 2.0 (Appendix C.2.3). Each point represents a model-layer pair (layers 5-12), with scores averaged over eight encoder training seeds (i.e., 40 points per panel). Dashed lines show OLS fits, and each panel reports Pearson's r. Two-sided p-values for correlations with avg. SLM are shown to the right of each row, with rows ordered by increasing p-value (i.e., the most strongly correlated metric at the top).

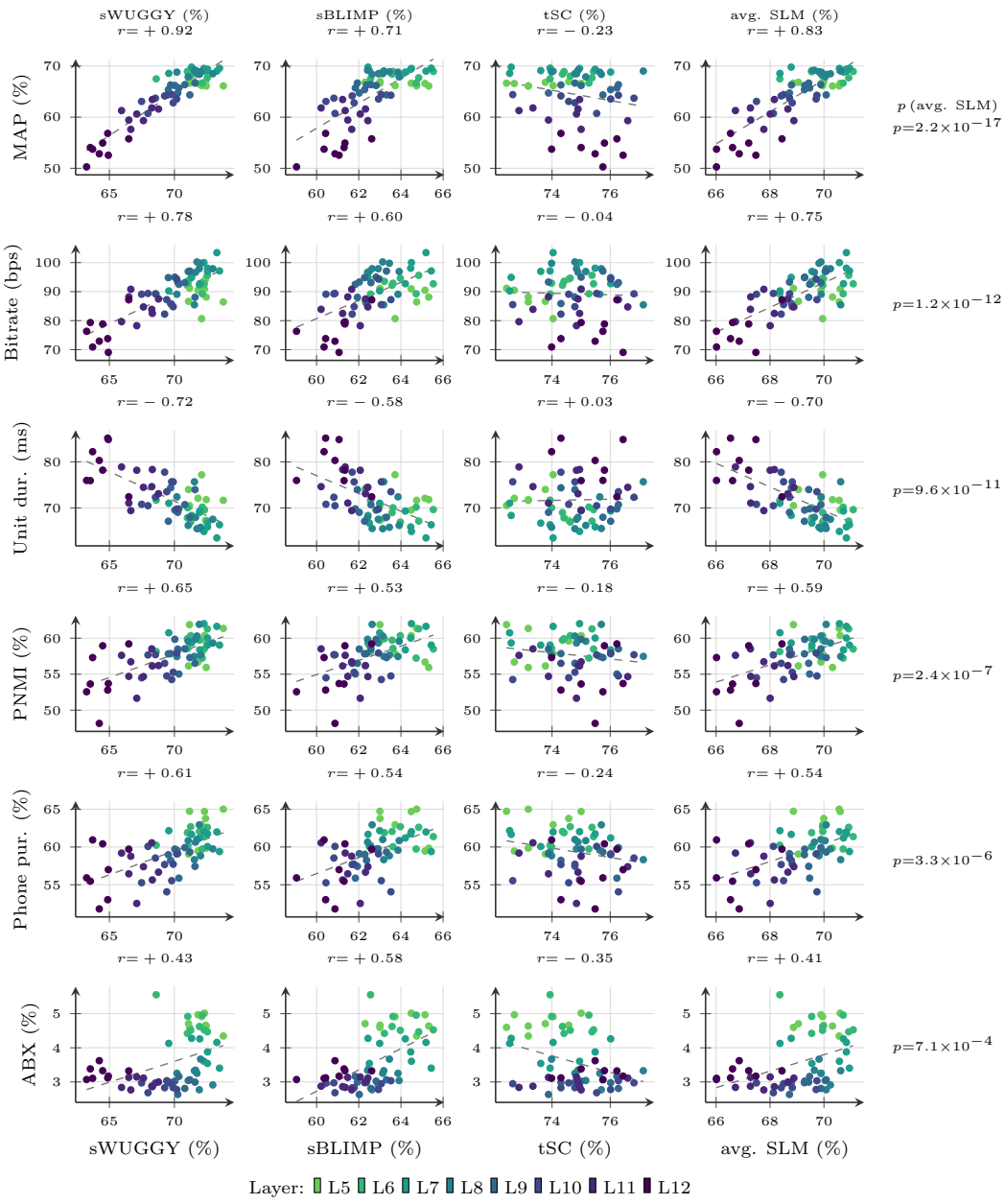


Figure 20: Pearson correlations between proxy metrics (rows) and target SLM metrics (columns: sWuggy, sBLIMP, tSC, and their average, denoted avg. SLM) *within* SpidR 2.0, across eight training seeds and eight encoder layers (5-12). Each point represents a layer-seed pair (i.e., 64 points per subplot). Dashed lines show OLS fits, and each panel reports Pearson's r. Two-sided p-values for correlations with avg. SLM are shown to the right of each row, with rows ordered by increasing p-value (i.e., the most strongly correlated metric at the top). The observed correlations primarily reflect variation across layers rather than across seeds.

Table 26: Speech resynthesis quality using oracle and predicted unit durations. Oracle durations are extracted from the ground-truth utterance, while predicted durations are estimated from the deduplicated unit sequence using the corresponding duration predictor.

	Oracle duration		Predicted duration	
Model	WER (↓)	UTMOS (↑)	WER (↓)	UTMOS (↑)
Ground truth	1.79%	4.15	–	–
SpidR	1.92%	2.99	2.11%	3.02
SpidR 2.0	2.68%	3.00	3.69%	3.08
GSLM2	1.95%	3.49	2.01%	3.51

C.3 Speech Resynthesis Evaluation

We evaluate the resynthesis quality of the unit vocoder trained for each GSLM system. Using the speech-continuation test data constructed from LibriSpeech (see Appendix B.4.4), we extract speech units from each utterance and resynthesize the speech with the corresponding unit HiFi-GAN vocoder. We consider two duration conditions: oracle durations extracted from the source utterance and predicted durations generated from deduplicated speech units by the corresponding duration predictor. We evaluate intelligibility using word error rate (WER), computed with Qwen3-Omni, and naturalness with UTMOS (Saeki et al., 2022). Table 26 reports the results, including performance on the unmodified test utterances as a ground-truth reference.

SpidR achieves lower WER than SpidR 2.0 under both oracle durations and predicted durations, while the two models obtain similar UTMOS scores. The increase in WER when replacing oracle durations with predicted durations is also smaller for SpidR than for SpidR 2.0 (0.19 vs. 1.01 percentage points), suggesting that the durations of SpidR 2.0's coarser units are more difficult to predict. Together with the ABX results from Table 5 (3.92% for SpidR vs. 2.83% for SpidR 2.0), these findings suggest a trade-off between phonemic discriminability and resynthesis fidelity: SpidR retains finer-grained acoustic detail that benefits resynthesis, whereas SpidR 2.0 achieves better ABX performance but produces coarser units with fewer active codes (see Appendix C.2.1), potentially discarding acoustic information needed for faithful resynthesis. However, adjusting the regularization weights can produce shorter units and increase codebook usage (see Appendix C.2.4), and we have not fully explored how these adjustments affect resynthesis quality.

GSLM2, by contrast, combines multiple unit levels in its vocoder. It achieves substantially lower WER than SpidR 2.0 under both oracle durations and predicted durations, while obtaining the highest UTMOS scores among the three systems. Moreover, replacing oracle durations with predicted durations increases its WER by only 0.06 percentage points, suggesting that cross-stream deduplication preserves sufficient temporal resolution for accurate duration prediction. These results indicate that multi-level units help preserve fine-grained speech variation captured by the SpidR 2.0 encoder, thereby improving resynthesis quality. This interpretation is consistent with prior resynthesis findings (Shi et al., 2024b; Chang et al., 2025b). In GSLM2, these improvements occur alongside gains in SLM performance.

C.4 Statistical Significance Analysis

C.4.1 Confidence Intervals

Table 27 reports marginal 95% bootstrap confidence intervals (CIs) for the main results in Table 2. We use 10,000 item-level bootstrap resamples. For sWuggy, we first average scores across the four voice realizations of each lexical item and then resample in-vocabulary and out-of-vocabulary items separately. For sBLIMP, items are resampled within each linguistic subtype. For sSC and tSC, story items are resampled. For each LT-Swap task, we resample items separately within each frequency bin. In each bootstrap draw, we compute bin-level accuracies and macro-average over frequency bins. These intervals quantify uncertainty arising from the finite size of the evaluation set for a fixed checkpoint. Appendix C.4.2 further investigates training seed variance, focusing on SpidR 2.0.

Table 27: Accuracy (%) and corresponding 95% bootstrap confidence intervals for the main results reported in Table 2.

		ZeroSpeech'21 (↑)		StoryCloze (↑)		LongTail-Swap (↑)		
	Model	sWuggy	sBLIMP	tSC	sSC	WS	IS	AS
600 h / 90M	Text LM	73.4 [72.9, 73.8]	61.7 [61.3, 62.0]	75.4 [73.4, 77.3]	53.4 [51.2, 55.7]	73.5 [72.9, 74.0]	80.6 [79.9, 81.3]	62.2 [61.3, 63.2]
	SpidR SLM	67.4 [67.0, 67.7]	54.2 [53.9, 54.6]	65.4 [63.2, 67.7]	52.4 [50.1, 54.6]	54.3 [53.6, 55.0]	57.9 [57.0, 59.1]	54.2 [53.4, 55.3]
	SpidR 2.0 SLM	69.2 [68.9, 69.6]	57.7 [57.4, 58.1]	68.1 [66.0, 70.1]	52.7 [50.4, 54.9]	58.0 [57.3, 58.6]	63.5 [62.8, 64.8]	55.0 [54.1, 56.1]
	GSLM2	71.5 [71.1, 71.8]	59.8 [59.4, 60.1]	74.1 [72.1, 76.1]	52.0 [49.9, 54.4]	62.4 [61.8, 63.1]	68.2 [67.5, 69.4]	59.1 [58.1, 60.1]
6K h / 300M	Text LM	79.5 [79.1, 79.9]	66.8 [66.4, 67.1]	84.8 [83.2, 86.4]	52.2 [49.9, 54.4]	80.1 [79.7, 80.5]	90.9 [90.7, 91.1]	79.1 [78.8, 79.4]
	SpidR SLM	71.1 [70.8, 71.4]	59.3 [58.9, 59.6]	71.4 [69.3, 73.4]	50.7 [48.4, 52.9]	62.0 [61.5, 62.4]	63.0 [62.6, 63.4]	58.5 [58.1, 58.8]
	SpidR 2.0 SLM	73.9 [73.6, 74.3]	65.1 [64.7, 65.4]	74.7 [72.6, 76.6]	52.2 [50.0, 54.5]	67.5 [67.0, 67.9]	69.0 [68.7, 69.4]	64.9 [64.5, 65.3]
	GSLM2	76.3 [76.0, 76.6]	67.2 [66.9, 67.5]	78.9 [77.1, 80.8]	52.5 [50.2, 54.8]	75.6 [75.2, 76.1]	75.5 [75.2, 75.9]	69.8 [69.4, 70.2]
60K h / 1.2B	Text LM	83.8 [83.5, 84.1]	71.5 [71.2, 71.8]	89.2 [87.8, 90.6]	54.6 [52.3, 56.8]	87.3 [87.0, 87.7]	94.2 [94.0, 94.3]	85.4 [85.2, 85.7]
	SpidR SLM	73.9 [73.6, 74.2]	62.4 [62.1, 62.8]	75.1 [73.1, 77.0]	52.8 [50.7, 55.1]	66.4 [66.0, 66.9]	65.5 [65.2, 65.8]	60.7 [60.3, 61.0]
	SpidR 2.0 SLM	76.7 [76.4, 77.0]	69.6 [69.2, 69.9]	81.0 [79.2, 82.8]	56.2 [54.0, 58.4]	74.8 [74.3, 75.2]	72.5 [72.2, 72.8]	66.6 [66.2, 66.9]
	GSLM2	79.3 [79.0, 79.6]	71.8 [71.5, 72.1]	84.5 [83.0, 86.2]	57.6 [55.2, 59.7]	79.5 [79.1, 79.9]	78.5 [78.3, 78.8]	69.9 [69.6, 70.3]

We perform pairwise significance tests among the three SLMs using matched evaluation items. We use two-sided label-swap tests with 10^6 random swaps for sWuggy and exact McNemar tests for sBLIMP, sSC, and tSC. For LT-Swap, we use two-sided paired label-swap tests with 10^6 random swaps, computing accuracy within each frequency bin and macro-averaging equally across bins for each swap type. For each model pair and training scale, we apply Holm correction across the overall results for sWuggy, sBLIMP, sSC, and tSC and, separately, across the three LT-Swap types. Across all training scales, the three SLMs differ significantly from one another on both sWuggy and sBLIMP (all $p \leq 3 \times 10^{-6}$). GSLM2 also significantly outperforms both SpidR and SpidR 2.0 on tSC at every scale (all $p < 10^{-5}$). On sSC, no pairwise difference is significant at 600 h or 6K h. At 60K h, SpidR 2.0 and GSLM2 both significantly outperform SpidR ($p \leq 0.0065$), whereas they do not differ significantly from each other (56.2% vs. 57.6%; $p = 0.20$). On LT-Swap, all observed improvements are statistically significant (all $p \leq 3 \times 10^{-6}$), except for the AS comparison between SpidR SLM and SpidR 2.0 SLM at 600 h (54.2% vs. 55.0%), which is not significant ($p = 0.27$).

C.4.2 Training Seed Variance

The confidence intervals in Appendix C.4.1 quantify uncertainty due to the finite evaluation set for a fixed checkpoint. Here, we examine the complementary variation introduced by retraining the speech encoder and the SLM. Training variability can affect model comparisons even when evaluation-set confidence intervals are narrow, as highlighted in other work across deep learning fields (Henderson et al., 2018; Bouthillier et al., 2021; Dufour et al., 2026). We quantify this variability for SpidR 2.0 SLM on the ZeroSpeech'21 and StoryCloze benchmarks, for which spoken language modeling studies typically report results from a single training run.

Experimental setup. We pretrain SpidR 2.0 encoders on LibriSpeech and train 90M-parameter single-stream SLMs on 6K hours of Libri-Light. We conduct three experiments. (1) We train eight encoders with seeds $\{0, \ldots, 7\}$. For each encoder, we train a separate SLM on the speech units extracted from each of layers 5–12, keeping the SLM seed fixed at 0. (2) We use the encoder trained with seed 0 and vary the SLM seed over $\{0, \ldots, 7\}$ for each of the same layers. (3) We repeat SLM training eight times with its seed fixed at 0, using speech units from layer 7 of the seed-0 encoder. This final experiment measures residual variation between nominally identical training runs.

Table 28: Estimated standard deviations for SpidR 2.0 SLM on ZeroSpeech'21 and StoryCloze. σ_{tot} (Eq. (13)) estimates the pooled standard deviation when the encoder and SLM are independently retrained. For Average, the variance estimates are computed from the per-run mean of sWuggy, sBLIMP, and tSC accuracies, excluding sSC in gray. All values are in accuracy percentage points.

Metric	σ_{slm}	σ_{enc}	σ_{res}	σ_{tot}
sWuggy	0.16	0.99	0.02	1.00
sBLIMP	0.38	0.97	0.09	1.04
tSC	0.60	1.05	0.15	1.21
sSC	0.62	0.88	0.07	1.08
Average	0.27	0.75	0.03	0.79

Table 29: Layer-pooled estimates of σ_{enc} across five training recipes, each evaluated with eight encoder seeds and one SLM run per encoder seed and layer. Recipe columns follow the cumulative changes from SpidR to SpidR 2.0, as studied in Appendix C.2.3. Average excludes sSC (gray) and is defined as in Table 28. All values are in accuracy percentage points.

Metric	SpidR	+heads	+reg.	+norm.	**SpidR 2.0**
sWuggy	2.50	1.69	2.53	1.49	0.99
sBLIMP	1.23	0.59	1.36	1.16	0.97
tSC	1.77	1.90	1.28	0.86	1.05
sSC	1.08	1.05	0.91	0.73	0.88
Average	1.63	1.21	1.42	0.73	0.75

Variance estimates. For each metric, we first compute the sample variance across runs separately at each layer. Let $s^2_{1,l}$ and $s^2_{2,l}$ denote this variance for Exps. (1) and (2), respectively, at layer $l \in \{5, \dots, 12\}$. We pool these within-layer variances across the eight layers:

$$\sigma^2_{\text{enc,obs}} = \frac{1}{8}\sum_{l=5}^{12} s^2_{1,l}, \qquad \sigma^2_{\text{slm}} = \frac{1}{8}\sum_{l=5}^{12} s^2_{2,l}. \tag{11}$$

The resulting estimates summarize variability across runs within layers.

Fixing the SLM seed controls seeded random choices, such as weight initialization and data ordering, but does not necessarily guarantee full determinism during SLM training. We estimate σ_{res} as the sample standard deviation across the eight fixed-seed repetitions in Exp. (3), obtaining a nonzero estimate for every reported metric ($\sigma_{\text{res}} > 0$). The observed variance $\sigma^2_{\text{enc,obs}}$ in Exp. (1) therefore includes both encoder variation and residual variation from SLM retraining. Under an additive variance model, we estimate the encoder component by subtracting the residual variance:

$$\sigma^2_{\text{enc}} = \sigma^2_{\text{enc,obs}} - \sigma^2_{\text{res}}. \tag{12}$$

The SLM-training variance σ^2_{slm} observed in Exp. (2) also includes both variation across SLM seeds and residual variation. Retaining both contributions, we estimate the pooled standard deviation for independent encoder and SLM retraining as

$$\sigma_{\text{tot}} = \left(\sigma^2_{\text{enc}} + \sigma^2_{\text{slm}}\right)^{1/2}, \tag{13}$$

which counts residual variation once. This decomposition assumes independent, additive encoder and SLM effects with negligible interactions. Because residual variance is measured only at layer 7 of the seed-0 encoder, applying the same correction to other layers and encoder checkpoints assumes that their residual variances are comparable. For every reported metric, we observe that the estimated residual variance σ^2_{res} is smaller than the estimated SLM-training variance σ^2_{slm} by more than an order of magnitude.

Encoder pretraining accounts for most training variance. Table 28 shows that the estimated encoder component exceeds the SLM-training component for every reported metric. Under the additive variance model, the encoder share of total variance, $\sigma^2_{\text{enc}}/\sigma^2_{\text{tot}}$, ranges from approximately 67% for sSC to 98% for sWuggy. These findings highlight the importance of including encoder retraining when estimating training variability in this setting.

Training variability depends on the model recipe. We estimate σ_{enc} for each configuration in the progressive ablation study in Appendix C.2.3, using the same pooling procedure. For each recipe, we

subtract the residual variance estimated for SpidR 2.0, assuming comparable residual variance across recipes. Table 29 shows substantial differences in encoder variability across recipes, indicating that a single estimate of σ_{enc} may not generalize to another. SpidR 2.0 has lower estimated standard deviations than SpidR, particularly for sWuggy and the average score, although the reductions are not monotonic across intermediate configurations. The differences for sSC are smaller and should be interpreted in light of both models' near-chance performance in this setting, as also observed for the 300M-parameter models in Table 27.

Limitations. This study attempts to give order-of-magnitude estimates of retraining variability rather than point out statistical weaknesses. We acknowledge that, given resource constraints, conducting large-scale variance analyses might not be feasible. We therefore limit each experimental setting to eight runs. The additive model underlying σ_{tot} assumes independent encoder and SLM effects with negligible interactions. The full variance decomposition is evaluated in a single setting: SpidR 2.0 encoders pretrained on LibriSpeech and 90M-parameter SLMs trained on 6K-hour Libri-Light. These estimates may not generalize to larger training datasets. The closest related work (Dufour et al., 2026) reports that their variance estimates are scale-invariant, though the setting differs substantially from ours. We do not isolate separate sources of variance, as was done in that work. In addition, this study only focuses on ZeroSpeech'21 and StoryCloze, although we also report ABX and MAP spreads in Tables 18 and 19.