Minimizing Modality Gap from the Input Side: Your Speech LLM Can Be a Prosody-Aware Text LLM

Wenqian Cui[1]*, **Xiao-Hui Li**[2]†, **Daxin Tan**[2], **Qiyong Zheng**[1], **Irwin King**[1]†
[1]The Chinese University of Hong Kong, [2]Huawei Technologies

Abstract

Speech large language models (SLMs) are typically built from text large language model (TLM) checkpoints, yet they still suffer from a substantial modality gap. Prior work has mainly attempted to reduce this gap from the output side by making speech generation more text-like, but the gap remains. We argue that the key remaining bottleneck lies on the input side. We propose TextPro-SLM, an SLM that makes spoken input more closely resemble that of a prosody-aware text LLM. TextPro-SLM combines WhisperPro, a unified speech encoder that produces synchronized text tokens and prosody embeddings, with an LLM backbone trained to preserve the semantic capabilities of the original TLM while learning paralinguistic understanding. Experiments show that TextPro-SLM achieves the lowest modality gap among leading SLMs at both 3B and 7B scales, while also delivering strong overall performance on paralinguistic understanding tasks. These gains are achieved with only roughly 1,000 hours of LLM training audio, suggesting that reducing the modality gap from the input side is both effective and data-efficient.[3]

1 Introduction

Speech Large Language Models (Speech LLMs, SLMs) have been extensively studied in recent years to enable seamless speech-based interaction [1–4]. Most SLMs are built by continually training from Text Large Language Model (TLM) checkpoints [5–10], with the goal of extending the strong reasoning and language understanding capabilities of TLMs to spoken input and output. However, despite this shared backbone, a major challenge remains: the **modality gap**—speech-based question answering (QA) performance often remains substantially worse than the text-based QA performance of the underlying TLM, limiting the practical usability of SLMs. For example, prior studies [11] report that GLM-4-Voice [12] suffers up to a 20% performance drop on several QA benchmarks.

Most existing studies attempt to bridge the modality gap from the **output side**. Early SLMs use the LLM backbone to generate only speech tokens, enabling fully end-to-end spoken interaction [13–15]. Later work improves performance by having the LLM first generate intermediate text tokens before emitting speech tokens [12, 16–18]. More recent approaches further decouple text generation from speech synthesis through a "thinker-talker" architecture, in which the LLM backbone generates only text tokens while a separate talker module produces speech conditioned on the thinker's hidden states [19–23]. This line of progress reveals a clear pattern: the closer the generation process is to the interaction paradigm of a TLM, the better the resulting SLM tends to perform. Nevertheless, even these advances do not eliminate the modality gap.

This observation suggests that the remaining bottleneck may lie primarily on the **input side**. Under modern thinker-talker designs, the output side already resembles that of a TLM. The more fundamental difficulty is that, unlike a TLM, an SLM must infer both linguistic content and paralinguistic cues

*Contact: wenqian.cui@link.cuhk.edu.hk
†Corresponding author.
[3]Code is available at `https://github.com/dreamtheater123/TextPro-SLM`.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

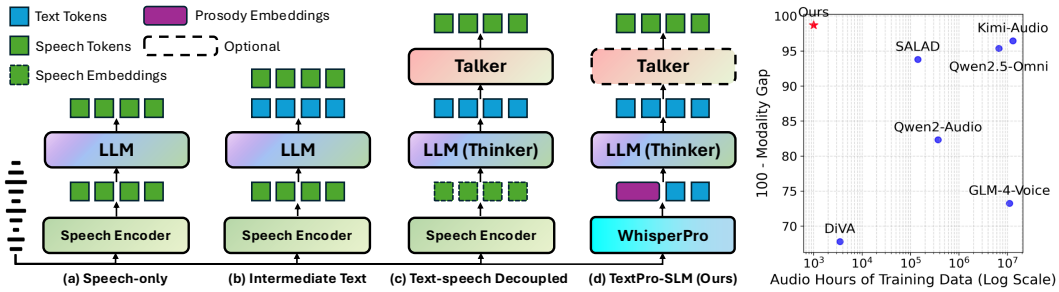

Figure 1: Left: Comparison of the architectural designs used by prior SLMs and our approach. Right: Comparison of modality gap versus the amount of audio training data across different approaches.

directly from speech representations, which are often not presented to the LLM in a form it can fully exploit [24, 25]. Motivated by this perspective, we propose to minimize the modality gap from the input side by making a Speech LLM behave as closely as possible to a **prosody-aware text LLM**. The central idea is simple: if the output side of modern SLMs is already converging toward text-style generation, then the input side should also provide the backbone with a representation that is both TLM-compatible and prosody-preserving. Rather than treating speech purely as an acoustic signal that must be compressed into a single latent stream, we explicitly separate and preserve two complementary aspects of spoken input: *what is said* and *how it is said*.

To achieve this, we propose **TextPro-SLM**, which consists of two key components. **1) Speech encoder.** We introduce **WhisperPro**, a unified **Whisper**-based speech encoder that produces two outputs: text tokens that capture utterance content and **Pro**sody embeddings that encode paralinguistic information such as speaking style, emotion, and speaker timbre. WhisperPro is trained to support both accurate transcription and prosody-rich representation learning, so the downstream LLM receives a richer and more TLM-compatible view of the input speech. **2) LLM backbone.** We train the LLM backbone with two coupled objectives: preserving the semantic capabilities of the original TLM and learning to understand paralinguistics. Specifically, we retain the semantic intelligence of the underlying TLM through knowledge distillation, while jointly optimizing the model on paralinguistic-understanding tasks. In this way, both linguistic meaning and paralinguistic information can propagate through the model, yielding an end-to-end SLM that remains semantically strong while also interpreting how something is spoken. Figure 1 compares the architectural design of TextPro-SLM with prior solutions.

Experimental results show that our method not only achieves the lowest modality gap among the compared approaches but also attains competitive performance on paralinguistic-understanding tasks, while remaining extremely **data efficient**, requiring only roughly 1,000 hours of audio data for LLM training. We attribute this efficiency to the fact that our design stays closer to the original operating regime of the underlying TLM than prior SLM architectures, making adaptation from text to speech more direct and less data-intensive. Overall, our results suggest that an effective route toward stronger Speech LLMs is to turn them into prosody-aware text LLMs.

2 Related Work

2.1 Speech Large Language Models

SLMs aim to enable seamless spoken dialogue interactions [1]. A typical SLM consists of three main components: 1) a speech encoder that converts the input waveform into speech representations, typically either continuous speech embeddings [22, 23] or discrete speech tokens [26–28]; 2) an LLM backbone that processes these representations and generates outputs, which may be text tokens [23], speech tokens [14], or both [12]; and 3) a speech decoder that synthesizes the model output into spoken responses. Early SLMs are trained purely on speech data and typically represent speech as fully discrete tokens [13]. Later work shows that leveraging a pre-trained TLM as the backbone leads to substantially stronger performance on speech tasks [14]. Since then, building SLMs from TLM checkpoints has become standard practice, with much of the research focusing on explicit speech-text

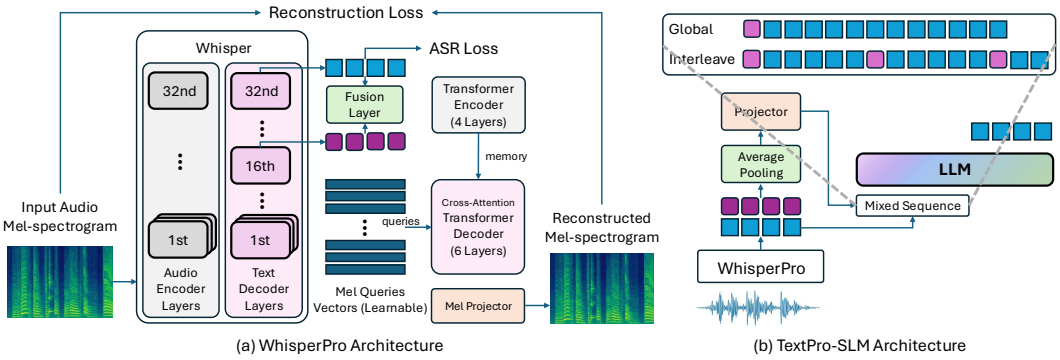

Figure 2: Model architecture of WhisperPro and TextPro-SLM.

alignment leveraging Automatic Speech Recognition (ASR), Text-to-Speech Synthesis (TTS), or interleaved speech-text training data [12, 15, 29].

Recent SLMs can be broadly grouped into two architectural paradigms according to the form of speech input provided to the LLM. **1) Discrete models:** the speech encoder produces discrete speech tokens, and the LLM consumes these tokens directly to generate text tokens, speech tokens, or both [12, 18]. **2) Continuous models:** the speech encoder produces continuous embeddings, which are then mapped into the LLM embedding space through a projector module. Models in this category often adopt a thinker-talker architecture, where the thinker generates text tokens and a separate talker produces speech conditioned on the thinker's hidden states [22, 23]. Our approach differs from both paradigms by providing the LLM with text tokens together with prosody embeddings, rather than asking it to infer both semantic and prosodic information from a single speech representation. This design better preserves the operating regime of the underlying TLM and helps reduce the text-speech modality gap effectively.

2.2 Text-Speech Modality Gap

As discussed in Section 1, recent SLM architectures have narrowed the modality gap mainly by improving the output side. Here, we review work that explicitly studies this gap and work that attempts to reduce it, then clarify how our approach differs.

Some studies focus on understanding the gap itself. Hsu *et al.* [30] attribute it to the longer speech sequence and the sparsity of semantic information in speech. Xiang et al. [24] analyze the gap at the representation level and show that speech and text representations are similar in direction but differ in magnitude. Other studies aim to reduce the gap, mostly through improved training. SALAD [11] and Wang *et al.* [31] use Kullback-Leibler (KL) divergence to align the student's output on speech input with the teacher's output on text input. X-OPD [32] and CORD [33] extend this line with on-policy distillation, while Yang *et al.* [34] perform finer-grained layer-wise distillation. A separate line of work targets catastrophic forgetting during continual training on speech data: DeepOmni [35] uses a Mixture-of-Experts architecture [36] that separates text and speech experts, and Hsiao *et al.* [37] identify experience replay as the most effective continual-learning strategy [38]. However, these methods mainly improve optimization and training stability rather than the interaction paradigm itself, which we view as a more fundamental source of the modality gap.

3 Methodology

Our approach to minimizing the modality gap has two core components. Section 3.1 presents WhisperPro, our speech encoder, and Section 3.2 introduces the prosody-aware LLM backbone.

3.1 WhisperPro: An ASR Speech Encoder that Encodes Everything

We design a speech encoder that outputs both accurate transcripts and prosody-rich embeddings. Broadly, there are three ways to build such a model: starting from an ASR model and adding paralinguistic modeling, starting from a prosody-rich model such as a neural audio codec and adding

transcription ability, or training a new model from scratch for both goals. We adopt the first strategy for two reasons. First, prior work suggests that SLMs rely more heavily on semantic information than on fine-grained acoustic details [39, 40]. Second, strong ASR models are typically trained on much larger datasets than codec models. For instance, Whisper-large-v3 [41] is trained on 680,000 hours of audio, whereas a competitive codec model such as WavTokenizer [42] only requires as few as 585 hours. Starting from a strong ASR backbone therefore fits the downstream LLM well and requires significantly less training data. In practice, we choose Whisper-large-v3 [41] because of its strong transcription performance and robustness across diverse acoustic conditions.

The main challenge of this task is that ASR models are trained to prioritize lexical content, not prosody. To inject paralinguistic information into the representation space, we borrow the core idea behind codec models: preserve acoustic details through reconstruction. Concretely, we use Whisper as the encoder to produce both text tokens and token-aligned prosody embeddings, and we attach an additional decoder that reconstructs the input speech from these two outputs. This design encourages the encoder states to retain not only what was said, but also how it was said. Figure 2 illustrates the overall architecture of WhisperPro.

Architecture. The architecture of WhisperPro consists of two modules: Whisper-large-v3 and a mel-reconstructor. Let the input utterance be a waveform $\mathbf{x} \in \mathbb{R}^T$. We first follow Whisper's standard preprocessing pipeline to obtain a log-mel spectrogram $\mathbf{M} \in \mathbb{R}^{F \times L}$. Whisper-large-v3 then maps $\mathbf{M}$ to both the transcript and the prosody embeddings. Specifically, Whisper autoregressively produces a text token sequence

$$\mathbf{y} = \mathrm{Whisper}(\mathbf{M}) = (y_1, \ldots, y_N). \tag{1}$$

Whisper-large-v3 consists of 32 Transformer encoder layers and 32 Transformer decoder layers. We take the hidden states of each text token from the 16-th Whisper decoder layer as token-aligned prosody embeddings, denoted as

$$\mathbf{P} = \mathrm{WhisperDec}^{(16)}(\mathbf{M}, \mathbf{y}) = (p_1, \ldots, p_N) \in \mathbb{R}^{N \times d}. \tag{2}$$

Because these states are aligned with the generated text tokens, WhisperPro represents speech as two synchronized streams, $\{(y_i, \mathbf{p}_i)\}_{i=1}^N$, capturing *what* was said and *how* it was said. To force the model to preserve prosodic information, we attach a mel-reconstructor that predicts the original mel spectrogram from the transcript-prosody pairs. At the overview level, we denote this module as

$$\hat{\mathbf{M}} = \mathrm{MelReconstructor}(\mathbf{y}, \mathbf{P}). \tag{3}$$

Concretely, the text tokens and prosody embeddings are mapped into a shared space, fused token-wise, contextualized by a Transformer encoder, and decoded into frame-level acoustic features by a Transformer decoder with learnable frame queries. The detailed architecture of the mel-reconstructor is presented in Appendix B. This reconstruction objective encourages the decoder states to retain information not only for transcription, but also for recovering the acoustic realization of the utterance.

During training, WhisperPro is optimized with a multi-task objective consisting of an ASR loss and a Mel reconstruction loss. Let $\mathbf{y}^* = (y_1^*, \ldots, y_N^*)$ denote the ground-truth transcription. The ASR objective is the average token-level cross-entropy:

$$\mathcal{L}_{\mathrm{ASR}} = -\frac{1}{N} \sum_{i=1}^{N} \log p(y_i^* \mid y_{<i}^*, \mathbf{M}), \tag{4}$$

and the mel-reconstructor is trained with a mean squared error loss between the reconstructed and original Mel spectrograms:

$$\mathcal{L}_{\mathrm{mel}} = \frac{1}{FL} \left\| \hat{\mathbf{M}} - \mathbf{M} \right\|_2^2. \tag{5}$$

The final training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{ASR}} + \lambda \, \mathcal{L}_{\mathrm{mel}}, \tag{6}$$

where λ is a hyperparameter and is empirically set to 1.0.

Training. We use the 960-hour LibriSpeech training split [43] as our primary training corpus. Because the original LibriSpeech transcripts are written in uppercase and do not include punctuation, we pair each audio utterance with its punctuated transcription from LibriSpeech-PC [44]. In preliminary experiments, we observe that Whisper becomes less stable when transcribing utterances that contain both questions and answer choices. To improve ASR robustness in QA settings, we further augment

the training data with speech versions of CommonsenseQA [45], synthesized using Kokoro TTS with the af-heart voice [4]. We train WhisperPro for 2 epochs with a global batch size of 16, a learning rate of 1×10^{-5}, and a linear decay scheduler with 500 warmup steps.

3.2 Prosody-Aware LLM

Our LLM backbone is designed to simultaneously minimize the semantic modality gap and preserve strong paralinguistic understanding. We build on instruction-tuned TLMs, specifically Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct at different model scales.

Input Format. Our key design principle is to convert speech into a representation that better matches the native input format of a text LLM, while keeping semantic information dominant. Given the transcript and prosody embeddings produced by WhisperPro, we feed the transcript directly into the LLM as the primary input and inject a compressed prosody representation alongside it. We consider two input organization strategies: **1) Global prepending (default):** We summarize prosody into a single utterance-level vector and prepend it to the textual input. As we show later, this compact representation is already sufficient for strong paralinguistic understanding. **2) Interleaving:** To account for cases where prosodic changes within an utterance may matter, we additionally provide an interleaved formulation that inserts compressed prosody embeddings among text tokens at a fixed ratio. These designs preserve the original capabilities of the LLM while still providing a compact representation of how the utterance was spoken, consistent with prior findings that semantic information plays a more central role than raw acoustic detail in SLM modeling [39, 40].

Table 1: Summary of the datasets used in WhisperPro and LLM training, where Recon, KD, SER, and AGE represent Reconstruction, Knowledge Distillation, Speech Emotion Recognition, and Age Detection, respectively.

Datasets	Objective	Duration (h)
WhisperPro		
CommonsenseQA [45]	ASR + Recon	40
LibriSpeech [43]	ASR + Recon	960
Total	-	1,000
LLM		
CommonsenseQA [45]	KD	40
UltraChat (part) [46]	KD	255
ParaSpeechCaps [47]	Captioning	342
IEMOCAP [48]	SER	12
CREMA-D [49]	SER	5
SAVEE [50]	SER	0.5
TESS [51]	SER	1.6
ESD (Eng) [52]	SER	14.5
Common Voice (part) [53]	AGE	319
Total	-	989.6

Let the WhisperPro produce a transcription token sequence $\mathbf{y}^{\text{enc}} = (y_1^{\text{enc}}, \ldots, y_N^{\text{enc}})$ and the aligned prosody embeddings $\mathbf{P} = (p_1, \ldots, p_N)$. We then tokenize the transcribed text again using the LLM tokenizer, yielding $\mathbf{y}^{\text{llm}} = (y_1^{\text{llm}}, \ldots, y_T^{\text{llm}})$, where in general $T \neq N$ because the tokenization schemes of WhisperPro and the LLM need not coincide. We instantiate two ways of injecting prosody into the LLM, corresponding to the global prepending and interleaving designs introduced above. For **global prepending**, we first aggregate all prosody embeddings from the user question into a single utterance-level vector by average pooling:

$$\bar{\mathbf{p}} = \frac{1}{N} \sum_{i=1}^{N} p_i \in \mathbb{R}^d. \tag{7}$$

We then project the averaged prosody embedding into the LLM embedding space using a two-layer Multi-Layer Perceptron (MLP), and place this projected prosody embedding at the front of the input sequence. Therefore, the final input to the LLM is

$$\mathbf{H}_{\text{global}}^{(0)} = \left[\tilde{\mathbf{p}};\, y_1^{\text{llm}};\, \ldots;\, y_T^{\text{llm}}\right], \tag{8}$$

where $\tilde{\mathbf{p}}$ denotes the projected global prosody embedding. Thus, the backbone receives a sequence that is almost identical to a standard text-only prompt, except for one prepended prosody token summarizing how the question was spoken. For **interleaving**, we preserve finer-grained prosodic variation by first determining how many prosody embeddings will be inserted into the LLM input, and then compressing the original prosody sequence into that many averaged embeddings. Given a fixed insertion ratio r, we calculate $M = \lceil T/r \rceil$ as the number of interleaved groups needed for the LLM token sequence, and then evenly partition the original prosody sequence $\mathbf{P}$ of length N into M consecutive groups. For each group, we average the prosody embeddings within that group and apply the same projector to obtain one interleaved prosody embedding $\tilde{p}_j$, resulting in $\tilde{\mathbf{P}} = (\tilde{p}_1, \ldots, \tilde{p}_M)$.

[4] `https://github.com/hexgrad/kokoro-82M`

Table 2: ASR and paralinguistic probing performance of WhisperPro compared to Whisper-large-v3. Probing results are reported for 1-layer and 2-layer MLP probes.

Model	Probe	WER ↓		Probing ↑				Avg.
		LibriSpeech	GigaSpeech	SER	SI	F0	Energy	
Whisper-large-v3	1-layer	4.43%	13.66%	33.84	38.83	27.45	14.64	28.69
	2-layer			41.02	60.67	34.09	18.35	38.53
WhisperPro	1-layer	**1.96%**	**12.89%**	53.52	79.31	46.81	30.78	52.61
	2-layer			69.63	87.84	50.34	36.56	61.09

Finally, the projected prosody embeddings and text tokens are interleaved, giving the final input sequence represented as

$$\mathbf{H}^{(0)}_{\text{interleave}} = \left[\tilde{p}_1;\, \mathbf{y}^{(1)};\, \tilde{p}_2;\, \mathbf{y}^{(2)};\, \ldots;\, \tilde{p}_M;\, \mathbf{y}^{(M)}\right], \tag{9}$$

where $\mathbf{y}^{(j)}$ denotes the j-th LLM text-token group, containing up to r tokens.

Remark 1 *We omit the training of the speech synthesis module in this work due to the following two reasons. 1) Dominant approaches in this field measure the modality gap using textual outputs. 2) We see speech synthesis based on textual outputs as a largely decoupled problem whose primary goal is to improve speech naturalness and fidelity, rather than to reduce the modality gap between speech understanding and text understanding.*

Training Data and Pipeline. We utilize two sets of training data for different purposes.

1. **Semantic Intelligence Preservation:** Following prior work [11, 31], we construct training data through knowledge distillation from teacher models. Specifically, we use UltraChat[5] [46] to distill general instruction-following ability and CommonsenseQA [45] to distill QA ability. We use the original TLMs as teacher models, and the distillation pipeline is as follows. First, we use Kokoro TTS to convert the text-only datasets into speech. Next, we pass the synthesized speech through WhisperPro to obtain transcriptions and prosody embeddings. We then use the transcriptions to query the teacher models and collect their responses. Finally, we feed the transcription-prosody pairs into the LLM backbone as inputs and use the teacher responses as supervision targets.
2. **Paralinguistic Understanding Capabilities:** To develop paralinguistic understanding, we focus on four representative speech attributes: speaker emotion, gender, accent, and age. We use two types of datasets for paralinguistic training. **1) Speech Captioning:** Speech captioning datasets pair speech utterances with natural-language descriptions that often capture paralinguistic cues. Training on such data helps the model learn to associate speech signals with the paralinguistic information they convey. We use the ParaSpeechCaps dataset [47] for this purpose since it contains the gender and accent attributes. **2) Feature Detection:** Feature detection datasets provide speech utterances together with ground-truth annotations for a target paralinguistic attribute. Because the captions in ParaSpeechCaps do not include emotion or age labels, we use dedicated feature detection datasets to train the model on these two attributes. For emotion, we use IEMOCAP [48], CREMA-D [49], SAVEE [50], TESS [51], and the English portion of ESD [52]. For age, we use the first 1/4 of the Common Voice data [53] with speaker-age labels.

   Remark 2 *We acknowledge that, although the features described above are representative, they do not cover the full paralinguistic space. Nevertheless, we use them as a proof of concept to show that our model can demonstrate strong paralinguistic understanding capabilities.*

We adopt a two-stage training pipeline. In the first stage, only the prosody projector is trained while the LLM backbone remains frozen. In the second stage, we jointly train the projector and the LLM. Each stage trains for 2 epochs with a global batch size of 1,024 and a linear decay scheduler with no warmups. We use a learning rate of 1×10^{-4} in Stage 1 and 1×10^{-5} in Stage 2. Both stages are trained with cross-entropy loss. Table 1 summarizes the datasets used to train WhisperPro and the LLM, along with their training objectives and audio durations.

[5]We use only 1/10 of UltraChat, i.e., `train_2.jsonl`, from `https://huggingface.co/datasets/openbmb/UltraChat`.

Table 3: Various SLM performance ↑ (%) and modality gap ↓ on different benchmarks. Gap cells are color-coded from best (green) to worst (red) for each task, where lower is better. We save the word Instruct for Kimi-Audio, Qwen2.5-3B and 7B to save spaces.

	StoryCloze		MMSU		OBQA		ARC-Easy		ARC-C		PIQA		Avg.	
	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap
Random	50.0	-	25.0	-	25.0	-	25.0	-	25.0	-	50.0	-	33.3	-
*Cascaded **Toplines**: ASR (Whisper-v3-Large) + LLM*														
ASR + Qwen2.5-3B	88.3	4.3	58.9	**2.7**	72.7	4.0	90.5	3.7	80.1	3.7	52.7	-0.6	73.9	3.0
ASR + Qwen2.5-7B	**89.1**	**0.1**	**67.9**	2.9	80.0	5.7	89.9	6.2	83.3	5.8	63.0	5.1	78.9	4.3
End-to-end Systems														
Qwen2-Audio-7B	71.9	9.0	29.5	18.7	39.6	37.1	43.5	28.5	43.5	28.5	73.4	5.4	50.2	21.2
DiVA-Llama3.1-8B	82.1	15.8	28.8	34.1	40.0	35.0	39.7	53.6	33.7	48.9	35.6	30.9	43.3	36.4
GLM-4-Voice-9B	76.4	20.6	39.2	28.3	52.1	32.5	73.2	24.6	59.5	33.8	47.3	30.9	57.9	28.4
Qwen2.5-Omni-7B	83.9	5.4	61.3	9.6	81.5	4.2	94.9	1.2	86.1	3.0	72.8	-4.8	80.1	3.1
Kimi-Audio-7B	66.6	22.6	61.6	9.2	83.7	2.0	95.2	0.9	88.2	1.0	**82.9**	**-14.8**	79.7	3.5
Modality Gap-optimized Systems														
SALAD-3B														
Stage I	75.5	7.4	47.3	14.6	65.5	16.3	-	-	75.6	6.2	78.3	0.3	68.4	9.0
Stage II	75.8	7.1	52.5	9.4	76.7	5.1	-	-	79.9	1.9	78.1	0.5	72.6	4.8
TextPro-SLM-3B	84.2	8.4	57.6	4.0	73.4	3.3	92.9	1.4	80.7	3.1	56.9	-4.8	74.3	2.6
SALAD-7B														
Stage I	81.5	3.5	55.3	15.5	69.7	19.3	-	-	82.3	6.1	80.3	0.4	73.8	9.0
Stage II	81.5	3.5	57.5	13.3	75.1	13.9	-	-	84.0	4.4	80.3	0.4	75.7	7.1
TextPro-SLM-7B	88.6	0.6	66.7	4.2	**85.1**	**0.7**	95.8	0.3	**89.4**	**-0.2**	69.7	-1.6	**82.5**	**0.7**
TextPro-SLM-7B 5:1	87.6	1.6	66.2	4.7	83.3	2.5	**95.9**	**0.2**	88.8	0.4	70.8	-2.7	82.1	1.1
Ablation Studies														
TextPro-SLM-7B w/o KD	84.7	4.5	46.7	24.2	55.8	30.0	89.5	6.6	75.8	13.4	66.6	1.5	69.9	13.3
WhisperPro + Qwen2.5-7B	88.8	0.4	62.6	8.3	80.4	5.4	95.0	1.1	87.8	1.4	63.3	4.8	79.7	3.6

4 Experiments

We conduct experiments to demonstrate that our proposed approach minimizes the modality gap while demonstrating strong paralinguistic understanding abilities. We provide both WhisperPro and LLM evaluation results.

4.1 WhisperPro Evaluations

Before evaluating the full system, we first assess WhisperPro in isolation. Our goal is to verify that it preserves Whisper's strong semantic modeling ability while learning richer prosodic representations. To this end, we compare WhisperPro directly with the original Whisper-large-v3 along two dimensions: ASR quality and paralinguistic information encoded in the prosody embeddings.

To evaluate semantic modeling, we measure ASR performance on the test sets of LibriSpeech [43] and GigaSpeech [54] using word error rate (WER). As shown in Table 2, WhisperPro achieves slightly lower WER than Whisper-large-v3 on both benchmarks. This result indicates that our modifications do not compromise the strong transcription ability of the original Whisper model.

To evaluate prosodic modeling, we conduct probing experiments on four paralinguistic attributes: speech emotion recognition (SER), speaker identification (SI), fundamental frequency (F0), and energy. We use RAVDESS [55] for SER, the VoxCeleb1 test set [56] for SI, and the LibriTTS test set [57] for F0 and energy. Following standard probing practice, we train both 1-layer and 2-layer MLP classifiers on top of the extracted prosody embeddings. Because F0 and energy are continuous-valued signals rather than categorical labels, we discretize each feature into 10 quantile-based bins. More experimental details of the probing experiments are presented in Appendix C. Table 2 shows that the prosody embeddings produced by WhisperPro consistently encode substantially richer information than the original Whisper representations across all four attributes. Together, these findings suggest that our reconstruction objective significantly enhances paralinguistic representation learning while preserving semantic fidelity.

4.2 LLM Evaluations

4.2.1 Experimental Setup

Benchmarks and Evaluations. We evaluate our LLM on two groups of benchmarks: one for measuring the modality gap and the other for assessing paralinguistic understanding.[6] To evaluate the modality gap, we follow the benchmark setup of prior work [11], including Spoken StoryCloze [14], OpenBookQA and MMSU from VoiceBench [58], PIQA [59], and ARC, including both the Easy and Challenge splits [60]. To evaluate paralinguistic understanding, we primarily use the paralinguistic task set from AIR-Bench [61], which covers emotion recognition, speaker gender recognition, and speaker age prediction. All evaluations are conducted in the zero-shot setting.

In addition to the paralinguistic tasks provided by AIR-Bench, we consider speaker accent as another important paralinguistic attribute. Since we did not find a convenient off-the-shelf benchmark for this task, we construct our own accent detection benchmark and evaluate our models on it. Specifically, we use a subset of the Common Voice test set with accent labels and convert it into an AIR-Bench-style four-option multiple-choice benchmark. The four target labels are "United States", "England", "India and South Asia (India, Pakistan, Sri Lanka)", and "Europe". We sample 1,000 examples to build the benchmark. Details of the accent benchmark construction are provided in Appendix D. [7]

Baselines. We compare our model with a diverse set of SLM baselines, including leading industrial systems and methods explicitly designed to reduce the modality gap. We calculate the modality gap as the performance difference between a speech model given spoken input and its backbone TLM given textual input. The compared model pairs are GLM-4-Voice [12] and GLM-4-9B [10], Qwen2-Audio [62] and Qwen-7B-Chat [63], DiVA [64] and Llama-3-8B [65], Qwen2.5-Omni [23] and Qwen2.5-7B-Instruct [66], Kimi-Audio [29] and Qwen2.5-7B-Instruct, and SALAD [11] with Qwen2.5-3B&7B-Base. As a reference point, we also include cascaded systems that combine Whisper-large-v3 with Qwen2.5-3B&7B-Instruct.

4.2.2 Modality Gap Results

Following Cuervo *et al.* [11], we report both accuracy and modality gap for each dataset and model. The results are summarized in Table 3, where TextPro-SLM denotes the global prepending setting and TextPro-SLM 5:1 denotes the interleaving setting with a 5:1 ratio. We make four main observations. **1) TextPro-SLM achieves the lowest average modality gap among all compared SLMs at both the 3B and 7B scales.** In particular, TextPro-SLM-7B attains an average modality gap of 0.7%, substantially lower than SALAD's 7.1% and Qwen2.5-Omni's 3.1%. **2) TextPro-SLM-7B 5:1 achieves performance comparable to that of TextPro-SLM-7B.** This result suggests that, even with more injected prosody embeddings, our method preserves semantic capabilities effectively. **3) TextPro-SLM consistently achieves a slightly lower modality gap than the cascaded toplines.** On the 7B setting, it yields a lower modality gap on 4 of 6 benchmarks. We suspect that this advantage comes from the dedicated knowledge distillation applied to WhisperPro's outputs, and we examine this hypothesis further in the ablation study. **4) TextPro-SLM shows particularly strong gains on reasoning-required benchmarks compared with SALAD.** Datasets such as MMSU and OBQA require stronger reasoning ability, and TextPro-SLM-7B reduces the gap to 4.2% and 0.7%, respectively, compared with SALAD's 13.3% and 13.9%. Overall, these observations suggest that reducing the modality gap from the input side is highly effective, particularly considering TextPro-SLM only requires **fewer than 300 hours of audio** for knowledge distillation.

Table 4: Reasoning-heavy VoxEval math performance ↑ and modality gap ↓ (%).

	Elementary		High School		College		Avg.	
	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap
Text Topline								
Qwen2.5-7B-Instruct	82.1	-	81.3	-	80.4	-	81.3	-
End-to-end Systems								
Qwen2.5-Omni-7B	74.3	7.9	62.6	18.7	47.8	32.6	61.6	19.7
Kimi-Audio-7B	66.1	16.1	59.7	21.6	30.4	50.0	52.1	29.2
TextPro-SLM-7B	**81.4**	**0.7**	**77.8**	**3.5**	**69.6**	**10.9**	**76.3**	**5.0**

Ablation Studies. We study two ablated variants in Table 3. **1) TextPro-SLM-7B w/o KD:** In this setting, we remove knowledge distillation and retain only the paralinguistic-task training. As

[6] All speech inputs used in our evaluations are shorter than 30 seconds.
[7] We will open-source this accent benchmark.

expected, performance drops substantially. **2) WhisperPro + Qwen2.5-7B:** In this setting, we directly feed the text tokens produced by WhisperPro to Qwen2.5-7B-Instruct. We find that performance remains broadly comparable to that of TextPro-SLM, with slight drops on some benchmarks. Because the main differences between this setting and TextPro-SLM are the insertion of prosody embeddings and end-to-end model training, we hypothesize that TextPro-SLM benefits from adapting the LLM to WhisperPro-produced texts, making the backbone more compatible with the speech encoder.

Modality Gap on Reasoning-Heavy Tasks. Although our benchmark choices for measuring the modality gap in Table 3 follow prior work, they mostly involve tasks with low to moderate reasoning demands. We therefore further examine the modality gap on more reasoning-heavy benchmarks. Specifically, we use the three math-related tasks in VoxEval [67]: elementary math, high school math, and college math. Because Qwen2.5-Omni-7B and Kimi-Audio-7B show the lowest modality gap among the baseline SLMs in Table 3, we compare these two models with our approach on these three benchmarks. We cap the computational budget at 512 tokens. The results in Table 4 highlight two main findings. **1) All SLMs exhibit a larger modality gap on reasoning-heavy tasks.** Specifically, the average modality gap of Qwen2.5-Omni-7B increases from 3.1% to 19.7%, while Kimi-Audio-7B shows an average gap above 20%. **2) Our approach performs substantially better on reasoning-heavy tasks.** In particular, TextPro-SLM achieves a modality gap of just 3.5% on high school math, compared with 18.7% for Qwen2.5-Omni-7B and 21.6% for Kimi-Audio-7B. Overall, TextPro-SLM also yields the lowest average modality gap across these benchmarks.

4.2.3 Paralinguistic Understanding Results

Table 5 reports the results on four paralinguistic understanding tasks and reveals two main observations. **1) TextPro-SLM achieves the strongest paralinguistic understanding performance among all baselines.** This result suggests that even with only a single prosody embedding and roughly 1,000 hours of training audio, an SLM can already achieve state-of-the-art paralinguistic understanding. **2) TextPro-SLM-5:1 performs slightly better than TextPro-SLM on paralinguistic understanding.** In particular, it improves performance on three of the four tasks, likely because feeding more prosody embeddings into the LLM exposes richer acoustic cues. However, the overall gain remains modest, indicating that simply increasing the amount of prosodic input does not substantially improve paralinguistic understanding. Taken together, these results support our hypothesis that, for SLMs, semantic information is more critical than additional acoustic detail.

Table 5: SLM performance (%) on various paralinguistic understanding tasks.

Model	Emo	Gender	Age	Accent	AVG
Qwen2-Audio-7B	48.2	64.7	23.1	29.0	41.3
DiVA	30.7	46.8	33.3	32.5	35.8
GLM-4-Voice-9B	23.0	23.9	18.7	32.4	24.5
Qwen2.5-Omni-7B	54.8	**89.8**	44.8	45.3	58.7
Kimi-Audio-7B	**61.1**	75.9	60.5	31.2	57.2
TextPro-SLM-3B	58.8	80.2	63.7	45.0	61.9
TextPro-SLM-7B	60.5	88.6	64.9	45.1	64.8
TextPro-SLM-7B 5:1	57.7	88.8	**66.7**	**50.2**	**65.8**
Ablation Studies					
TextPro-SLM-7B w/o Recon	47.3	77.7	55.5	32.2	53.2
TextPro-SLM-7B w/o Train	51.8	28.5	43.6	16.1	35.0

Ablation Studies. We study two ablated variants in Table 5. **1) Without the reconstruction objective in WhisperPro (TextPro-SLM-7B w/o Recon):** In this setting, we use the transcription and prosody embedding from the original Whisper-large-v3, without our reconstruction-based training. Performance drops on all four paralinguistic tasks, indicating that the reconstruction objective is critical for preserving paralinguistic information throughout the pipeline. **2) Without paralinguistic training (TextPro-SLM-7B w/o Train):** In this setting, we directly feed the prosody embedding from WhisperPro into Qwen2.5-7B-Instruct without training the model on paralinguistic tasks. This variant also shows a substantial performance drop, confirming TextPro-SLM effectively acquires paralinguistic information through task-specific training.

5 Conclusion

In this work, we propose TextPro-SLM, a speech LLM designed to minimize the modality gap from the input side by making spoken input resemble that of a prosody-aware text LLM. Our approach combines WhisperPro, which produces synchronized text tokens and prosody embeddings, with an LLM backbone that preserves the semantic strengths of the original text model while learning to

interpret paralinguistic information. Across experiments at both 3B and 7B scales, TextPro-SLM achieves the lowest modality gap among strong SLM baselines and delivers strong performance on paralinguistic understanding benchmarks, while requiring only about 1,000 hours of audio training data. These results show that aligning speech inputs with text-style representations, with prosodic information preserved, is a promising and data-efficient direction for building more capable and practical SLMs.

Acknowledgments and Disclosure of Funding

The research presented in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 2300246, RGC C1043-24G) and (CUHK 14203425, RGC GRF 2151317).

References

[1] Wenqian Cui, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Steven Y Guo, and Irwin King. Recent advances in speech language models: A survey. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13943–13970, 2025.

[2] Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. Llama-omni: Seamless speech interaction with large language models. *arXiv preprint arXiv:2409.06666*, 2024.

[3] 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.

[4] Wenqian Cui, Lei Zhu, Xiaohui Li, Zhihan Guo, Haoli Bai, Lu Hou, and Irwin King. Think before you talk: Enhancing meaningful dialogue generation in full-duplex speech language models with planning-inspired text guidance. *arXiv preprint arXiv:2508.07375*, 2025.

[5] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[6] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[7] Aaron Adcock, Aayushi Srivastava, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pande, Abhinav Pandey, Abhinav Sharma, Abhishek Kadian, Abhishek Kumawat, Adam Kelsey, et al. The llama 4 herd: Architecture, training, evaluation, and deployment notes. *arXiv preprint arXiv:2601.11659*, 2026.

[8] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[9] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[10] Team Glm, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024.

[11] Santiago Cuervo, Skyler Seto, Maureen de Seyssel, Richard He Bai, Zijin Gu, Tatiana Likhomanenko, Navdeep Jaitly, and Zakaria Aldeneh. Closing the gap between text and speech understanding in llms. *arXiv preprint arXiv:2510.13632*, 2025.

[12] Aohan Zeng, Zhengxiao Du, Mingdao Liu, Kedong Wang, Shengmin Jiang, Lei Zhao, Yuxiao Dong, and Jie Tang. Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot. *arXiv preprint arXiv:2412.02612*, 2024.

[13] 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.

[14] 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. *Advances in Neural Information Processing Systems*, 36, 2024.

[15] Tu Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R Costa-Jussa, Maha Elbayad, Sravya Popuri, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, Itai Gat, et al. Spirit-lm: Interleaved spoken and written language model. *arXiv preprint arXiv:2402.05755*, 2024.

[16] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. SpeechGPT: Empowering large language models with intrinsic cross-modal conversational abilities. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15757–15773, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.1055. URL https://aclanthology.org/2023.findings-emnlp.1055.

[17] Zhifei Xie and Changqiao Wu. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*, 2024.

[18] Zuwei Long, Yunhang Shen, Chaoyou Fu, Heting Gao, Peixian Chen, Mengdan Zhang, Hang Shao, Jian Li, Jinlong Peng, Haoyu Cao, et al. Vita-audio: Fast interleaved audio-text token generation for efficient large speech-language model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[19] Qingkai Fang, Yan Zhou, Shoutao Guo, Shaolei Zhang, and Yang Feng. Llama-omni 2: Llm-based real-time spoken chatbot with autoregressive streaming speech synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18617–18629, 2025.

[20] Qian Chen, Yafeng Chen, Yanni Chen, Mengzhe Chen, Yingda Chen, Chong Deng, Zhihao Du, Ruize Gao, Changfeng Gao, Zhifu Gao, et al. Minmo: A multimodal large language model for seamless voice interaction. *arXiv preprint arXiv:2501.06282*, 2025.

[21] Yuhao Wang, Heyang Liu, Ziyang Cheng, Ronghua Wu, Qunshan Gu, Yanfeng Wang, and Yu Wang. Vocalnet: Speech llms with multi-token prediction for faster and high-quality generation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19595–19612, 2025.

[22] 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*, 2025.

[23] Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. Qwen2.5-omni technical report, 2025. URL https://arxiv.org/abs/2503.20215.

[24] Bajian Xiang, Shuaijiang Zhao, Tingwei Guo, and Wei Zou. Understanding the modality gap: An empirical study on the speech-text alignment mechanism of large speech language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5187–5202, 2025.

[25] Yuhao Zhang, Zhiheng Liu, Fan Bu, Ruiyu Zhang, Benyou Wang, and Haizhou Li. Soundwave: Less is more for speech-text alignment in llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18718–18738, 2025.

[26] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[27] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 244–250. IEEE, 2021.

[28] 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.

[29] Ding Ding, Zeqian Ju, Yichong Leng, Songxiang Liu, Tong Liu, Zeyu Shang, Kai Shen, Wei Song, Xu Tan, Heyi Tang, et al. Kimi-audio technical report. *arXiv preprint arXiv:2504.18425*, 2025.

[30] Ming-Hao Hsu, Xueyao Zhang, Xiaohai Tian, Jun Zhang, and Zhizheng Wu. Anatomy of the modality gap: Dissecting the internal states of end-to-end speech llms. *arXiv preprint arXiv:2603.01502*, 2026.

[31] Enzhi Wang, Qicheng Li, Zhiyuan Tang, and Yuhang Jia. Cross-modal knowledge distillation for speech large language models. *arXiv preprint arXiv:2509.14930*, 2025.

[32] Di Cao, Dongjie Fu, Hai Yu, Siqi Zheng, Xu Tan, and Tao Jin. X-opd: Cross-modal on-policy distillation for capability alignment in speech llms. *arXiv preprint arXiv:2603.24596*, 2026.

[33] Jing Hu, Danxiang Zhu, Xianlong Luo, Dan Zhang, Shuwei He, Yishu Lei, Haitao Zheng, Shikun Feng, Jingzhou He, Yu Sun, et al. Cord: Bridging the audio-text reasoning gap via weighted on-policy cross-modal distillation. *arXiv preprint arXiv:2601.16547*, 2026.

[34] Runyan Yang, Yuke Si, Yingying Gao, Junlan Feng, Chao Deng, and Shilei Zhang. Teaching audio models to reason: A unified framework for source-and layer-wise distillation. *arXiv preprint arXiv:2509.18579*, 2025.

[35] Hang Shao, Heting Gao, Yunhang Shen, Jiawei Chen, Zuwei Long, Dong Yang, Ke Li, and Xing Sun. Deepomni: Towards seamless and smart speech interaction with adaptive modality-specific moe. *arXiv preprint arXiv:2506.21864*, 2025.

[36] Siyuan Mu and Sen Lin. A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications. *arXiv preprint arXiv:2503.07137*, 2025.

[37] Chi-Yuan Hsiao, Ke-Han Lu, Kai-Wei Chang, Chih-Kai Yang, Wei-Chih Chen, and Hung-yi Lee. Analyzing mitigation strategies for catastrophic forgetting in end-to-end training of spoken language models. *arXiv preprint arXiv:2505.17496*, 2025.

[38] Dianzhi Yu, Xinni Zhang, Yankai Chen, Aiwei Liu, Yifei Zhang, Philip S Yu, and Irwin King. Recent advances of multimodal continual learning: A comprehensive survey. *arXiv preprint arXiv:2410.05352*, 2024.

[39] 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.

[40] Amir Hussein, Sameer Khurana, Gordon Wichern, Francois G Germain, and Jonathan Le Roux. Hasrd: Hierarchical acoustic and semantic representation disentanglement. *arXiv preprint arXiv:2506.00843*, 2025.

[41] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[42] Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, et al. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. *arXiv preprint arXiv:2408.16532*, 2024.

[43] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[44] Aleksandr Meister, Matvei Novikov, Nikolay Karpov, Evelina Bakhturina, Vitaly Lavrukhin, and Boris Ginsburg. Librispeech-pc: Benchmark for evaluation of punctuation and capitalization capabilities of end-to-end asr models. In *2023 IEEE automatic speech recognition and understanding workshop (ASRU)*, pages 1–7. IEEE, 2023.

[45] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.

[46] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, 2023.

[47] Anuj Diwan, Zhisheng Zheng, David Harwath, and Eunsol Choi. Scaling rich style-prompted text-to-speech datasets. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3639–3659, 2025.

[48] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335–359, 2008.

[49] Houwei Cao, David G Cooper, Michael K Keutmann, Ruben C Gur, Ani Nenkova, and Ragini Verma. Crowd-sourced emotional multimodal actors dataset (crema-d), 2025.

[50] Philip Jackson and SJUoSG Haq. Surrey audio-visual expressed emotion (savee) database. *University of Surrey: Guildford, UK*, 2014.

[51] Kate Dupuis and M Kathleen Pichora-Fuller. *Toronto emotional speech set (TESS)*. University of Toronto, Psychology Department Toronto, ON, Canada, 2010.

[52] Kun Zhou, Berrak Sisman, Rui Liu, and Haizhou Li. Emotional voice conversion: Theory, databases and esd. *Speech Communication*, 137:1–18, 2022.

[53] Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the twelfth language resources and evaluation conference*, pages 4218–4222, 2020.

[54] Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, et al. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. *arXiv preprint arXiv:2106.06909*, 2021.

[55] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018.

[56] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, 60:101027, 2020.

[57] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. *arXiv preprint arXiv:1904.02882*, 2019.

[58] Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T Tan, and Haizhou Li. Voicebench: Benchmarking llm-based voice assistants. *Transactions of the Association for Computational Linguistics*, 14:378–398, 2026.

[59] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

[60] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[61] Qian Yang, Jin Xu, Wenrui Liu, Yunfei Chu, Ziyue Jiang, Xiaohuan Zhou, Yichong Leng, Yuanjun Lv, Zhou Zhao, Chang Zhou, et al. Air-bench: Benchmarking large audio-language models via generative comprehension. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1979–1998, 2024.

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

[63] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[64] William Held, Yanzhe Zhang, Minzhi Li, Weiyan Shi, Michael J Ryan, and Diyi Yang. Distilling an end-to-end voice assistant without instruction training data. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7876–7891, 2025.

[65] 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.

[66] Qwen Team. Qwen2.5 technical report, 2024. URL `https://arxiv.org/abs/2412.15115`.

[67] Wenqian Cui, Xiaoqi Jiao, Ziqiao Meng, and Irwin King. Voxeval: Benchmarking the knowledge understanding capabilities of end-to-end spoken language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16735–16753, 2025.

[68] 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 the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8666–8681, 2022.

[69] Dong Zhang, Xin Zhang, Jun Zhan, Shimin Li, Yaqian Zhou, and Xipeng Qiu. Speechgpt-gen: Scaling chain-of-information speech generation. *arXiv preprint arXiv:2401.13527*, 2024.

[70] Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Yanqing Liu, Yichong Leng, Kaitao Song, Siliang Tang, et al. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. *arXiv preprint arXiv:2403.03100*, 2024.

[71] Kaizhi Qian, Xulin Fan, Junrui Ni, Slava Shechtman, Mark Hasegawa-Johnson, Chuang Gan, and Yang Zhang. Prosodylm: Uncovering the emerging prosody processing capabilities in speech language models. *arXiv preprint arXiv:2507.20091*, 2025.

[72] Chao-Hong Tan, Qian Chen, Wen Wang, Chong Deng, Qinglin Zhang, Luyao Cheng, Hai Yu, Xin Zhang, Xiang Lyu, Tianyu Zhao, et al. Drvoice: Parallel speech-text voice conversation model via dual-resolution speech representations. In *International Conference on Learning Representations*, volume 2026, pages 121421–121440, 2026.

A Discussions & Limitations

Although we have shown promising results for TextPro-SLM, the current study has several limitations. First, we do not train or evaluate a speech synthesis module, and all comparisons are made on text outputs. We make this choice deliberately because modality-gap studies in recent Speech LLM literature typically compare benchmark performance using text outputs, which has become a standard evaluation protocol. Under the increasingly common decoupled "thinking" and "talking" design, we view text generation as the component that most directly reflects the model's intelligence and understanding ability, while speech synthesis primarily concerns how accurately and naturally the generated text is pronounced. These two aspects are usually measured with different metrics, such as task accuracy for text generation and metrics such as WER and UTMOS for speech synthesis.

Second, although representative, our paralinguistic evaluation does not reflect the whole paralinguistic space. We choose a set of tasks that covers several important paralinguistic phenomena and use them to test whether the model can exploit prosodic information beyond plain lexical content. However, real-world spoken interaction may require understanding more complex and dynamic signals, such as abrupt emotion shifts, mixed affective states, speaker intention changes, or other nuanced discourse-level cues. Therefore, our paralinguistic results should be interpreted as a proof of concept that TextPro-SLM can achieve strong paralinguistic understanding rather than achieving state-of-the-art performance on the whole paralinguistic space.

Third, our current implementation may introduce additional latency. In our setup, the speech encoder runs only after the user has finished speaking, because the Whisper backbone we use does not support streaming inference. As a result, the system is sequential rather than fully streamable, which can slightly increase response latency in interactive scenarios. We view this issue primarily as a limitation of the underlying Whisper backbone rather than of the core idea of our method. In principle, the same input-side modality-gap reduction strategy could be combined with a streaming ASR backbone to support lower-latency processing. We leave this direction to future work.

Fourth, the current implementation of TextPro-SLM only supports speech inputs up to 30 seconds long. This restriction stems from the 30-second audio context window of the Whisper backbone used by WhisperPro [41]. We expect that training WhisperPro with an alternative ASR backbone that supports a longer audio context could address this limitation and enable TextPro-SLM to process longer speech inputs.

Finally, adapting our current framework to non-speech audio is non-trivial. Our current pipeline explicitly decomposes the input into a text transcription stream and a prosody embedding stream, which is well-suited to spoken language but not to general audio events. In more complex settings, a model may need to interpret non-speech sounds or mixed audio inputs that cannot be naturally represented through standard transcription. Extending the framework to such cases will likely require dedicated training data and new modeling techniques. One possible direction is to let the text stream emit placeholder tokens, such as masked tokens, when the input contains non-speech audio, while relying on the prosody or acoustic embedding stream to preserve the relevant information. We consider this extension important but outside the scope of the present paper.

B Detailed Architecture Explanation of WhisperPro

Input and Whisper backbone. The architecture of WhisperPro includes two main components: Whisper-large-v3 (encoder) and Mel Reconstructor (decoder). Let an input utterance be a raw waveform $\mathbf{x} \in \mathbb{R}^T$, where T denotes the number of waveform samples. Following Whisper's standard preprocessing pipeline, the waveform is first resampled and normalized to a fixed 30-second window, and then converted into a log-Mel spectrogram $\mathbf{M} \in \mathbb{R}^{F \times L}$, where $F = 128$ is the number of Mel bins and $L = 3000$ is the number of time frames. Thus, regardless of the original utterance duration, the acoustic input to WhisperPro is always represented as a 128×3000 Mel spectrogram.

Whisper-large-v3 then processes $\mathbf{M}$ using its encoder-decoder Transformer architecture. First, the Whisper encoder maps the acoustic input into a sequence of latent speech representations,

$$\mathbf{H}^{\text{enc}} = \text{Enc}_{\text{whisper}}(\mathbf{M}) \in \mathbb{R}^{L_e \times d}, \tag{10}$$

where L_e denotes the encoder sequence length and d is the Whisper hidden dimension. Conditioned on $\mathbf{H}^{\text{enc}}$, the Whisper decoder autoregressively predicts a text token sequence

$$\mathbf{y} = (y_1, \ldots, y_N), \qquad y_i \in \mathcal{V}, \tag{11}$$

with $\mathcal{V}$ the text vocabulary and N the output token length. Concretely, at decoder layer l, the hidden states are represented as

$$\mathbf{H}^{(l)} = (\mathbf{h}_1^{(l)}, \ldots, \mathbf{h}_N^{(l)}) \in \mathbb{R}^{N \times d}. \tag{12}$$

In our model, the prosody embeddings are taken from the 16-th Whisper decoder layer:

$$\mathbf{P} = \mathbf{H}^{(16)} \in \mathbb{R}^{N \times d}. \tag{13}$$

Because $\mathbf{P}$ is extracted from the decoder states aligned with the generated text sequence, each prosody vector $\mathbf{p}_i$ is naturally paired with exactly one text token y_i. Therefore, the model represents spoken input as two synchronized streams:

$$\{(y_i, \mathbf{p}_i)\}_{i=1}^{N}, \tag{14}$$

where y_i captures the linguistic content and $\mathbf{p}_i$ captures the associated prosodic information.

Mel reconstructor. The Mel Reconstructor is designed to recover the original Mel spectrogram $\mathbf{M}$ from these two token-aligned streams. Its role is to force the decoder hidden states to preserve information about how the utterance was spoken, rather than only what was said. To this end, each text token y_i is first embedded into a continuous vector,

$$\mathbf{e}_i = \text{Emb}(y_i) \in \mathbb{R}^{d_r}, \tag{15}$$

where d_r is the hidden size of the reconstructor. In parallel, each prosody embedding $\mathbf{p}_i \in \mathbb{R}^d$ is projected into the same space:

$$\tilde{\mathbf{p}}_i = \mathbf{W}_p \mathbf{p}_i + \mathbf{b}_p \in \mathbb{R}^{d_r}. \tag{16}$$

The two representations are then fused token-wise. Specifically, they are concatenated first and then projected by a feed-forward layer:

$$\mathbf{z}_i = \phi([\mathbf{e}_i; \tilde{\mathbf{p}}_i]) \in \mathbb{R}^{d_r}, \tag{17}$$

where $[\cdot;\cdot]$ denotes concatenation and $\phi(\cdot)$ denotes a fusion block consisting of a linear layer, layer normalization, GELU activation, and dropout. Stacking the resulting token-level fused vectors yields

$$\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_N) \in \mathbb{R}^{N \times d_r}. \tag{18}$$

To encode contextual dependencies across the whole utterance, $\mathbf{Z}$ is augmented with sinusoidal positional encoding and passed through a Transformer encoder:

$$\mathbf{U} = \text{Enc}_{\text{mel}}(\mathbf{Z}) \in \mathbb{R}^{N \times d_r}. \tag{19}$$

This encoded sequence $\mathbf{U}$ serves as the memory for Mel reconstruction. Intuitively, $\mathbf{U}$ integrates token identity and prosodic realization into a contextualized latent sequence.

The decoder side of the Mel Reconstructor is implemented as a Transformer decoder with cross-attention and a fixed set of learnable frame queries, one for each target Mel frame. This design addresses the mismatch between the variable-length token-level input sequence and the fixed-length frame-level output sequence produced by Whisper's preprocessing. Rather than imposing a one-to-one alignment between tokens and Mel frames, cross-attention allows each frame query to attend over the full token-level memory and extract the information needed to reconstruct its corresponding acoustic frame. Specifically, we introduce

$$\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_L) \in \mathbb{R}^{L \times d_r}, \tag{20}$$

where $L = 3000$ matches the number of Mel frames in Whisper's normalized input. We first add the same sinusoidal positional encoding to $\mathbf{Q}$ and feed it into a Transformer decoder that performs cross-attention over the memory $\mathbf{U}$:

$$\mathbf{D} = \text{Dec}_{\text{mel}}(\mathbf{Q}, \mathbf{U}) \in \mathbb{R}^{L \times d_r}. \tag{21}$$

Here, each query $\mathbf{q}_t$ learns to retrieve from $\mathbf{U}$ the information needed to reconstruct the t-th acoustic frame. This design turns Mel reconstruction into a sequence-to-sequence generation problem from token-aligned text-prosody representations to frame-aligned acoustic features.

Table 7: Common Voice accent strings grouped into the "Europe" category in our accent benchmark.

Common Voice labels mapped to "Europe"
German; French; Russian; German English; Dutch; Icelandic; Dutch English; Polish; German Accent; French Accent; Slovak; Deutsch English; English with a French accent; German native; native Dutch speaking; Swedish; Finnish

Finally, let $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_L)$, where $\mathbf{d}_t \in \mathbb{R}^{d_r}$ denotes the decoder output corresponding to the t-th Mel frame query. Each decoder output vector is then mapped to a Mel frame through a multilayer projection head:

$$\hat{\mathbf{m}}_t = g(\mathbf{d}_t) \in \mathbb{R}^F, \tag{22}$$

where $g(\cdot)$ is a feed-forward network and $F = 128$. Collecting all reconstructed frames gives the predicted Mel spectrogram

$$\hat{\mathbf{M}} = (\hat{\mathbf{m}}_1, \ldots, \hat{\mathbf{m}}_L) \in \mathbb{R}^{F \times L}. \tag{23}$$

Overall, the full computation pipeline can be summarized as

$$\mathbf{x} \to \mathbf{M} \to \mathbf{H}^{\text{enc}} \to \{\mathbf{y}, \mathbf{P}\} \to \mathbf{Z} \to \mathbf{U} \to \mathbf{D} \to \hat{\mathbf{M}}. \tag{24}$$

This formulation explicitly decomposes spoken input into a semantic stream and a prosodic stream, and then requires the model to jointly use both streams to reconstruct the original acoustics.

C Implementation Details of the Probing Experiments

For all probing experiments, we train lightweight MLP classifiers on top of the extracted prosody embeddings. We evaluate both a 1-layer and a 2-layer probe. The 1-layer probe consists of a dropout layer followed by a linear classifier, while the 2-layer probe consists of a linear layer, a ReLU activation, a dropout layer, and a final linear classifier. For the 2-layer probe, the hidden dimension is set to 256. Across all probing experiments, we use the Adam optimizer with learning rate 10^{-3}, dropout rate 0.1, batch size 32, and cross-entropy loss. The maximum number of training epochs is set to 20. We apply early stopping based on validation accuracy with a patience of 3 epochs, and the checkpoint with the best validation accuracy is used for the final evaluation.

Table 6: Number of samples in each label of our accent benchmark.

Target label	Samples
United States English	310
England English	310
India and South Asia (India, Pakistan, Sri Lanka)	309
Europe	71
Total	1,000

For the evaluation protocol, we do not rely on a single train-test split. Instead, we adopt a stratified multi-seed cross-validation setup. Specifically, for each probe architecture, we run 10 random seeds and, under each seed, perform 5-fold stratified cross-validation, resulting in 50 runs in total per probe. In each run, the data split is determined by shuffled stratified folds, and the model is re-initialized independently. We then aggregate the results across all runs. Therefore, the probing numbers reported in the paper are robust estimates obtained from repeated stratified cross-validation.

D Accent Benchmark Construction Details

We construct the accent benchmark from the English split test set of Common Voice 22.0 by selecting four target labels that are both representative of major English accent groups and sufficiently well supported in Common Voice [53]: "United States English", "England English", "India and South Asia (India, Pakistan, Sri Lanka)", and "Europe". Following AIR-Bench [61], the final benchmark contains 1,000 samples, with the per-category distribution summarized in Table 6. Because Common Voice does not provide a single label named "Europe", we map multiple Europe-related accent strings to this category. All the mapped strings are summarized in Table 7.

E WhisperPro Design Choice Investigations

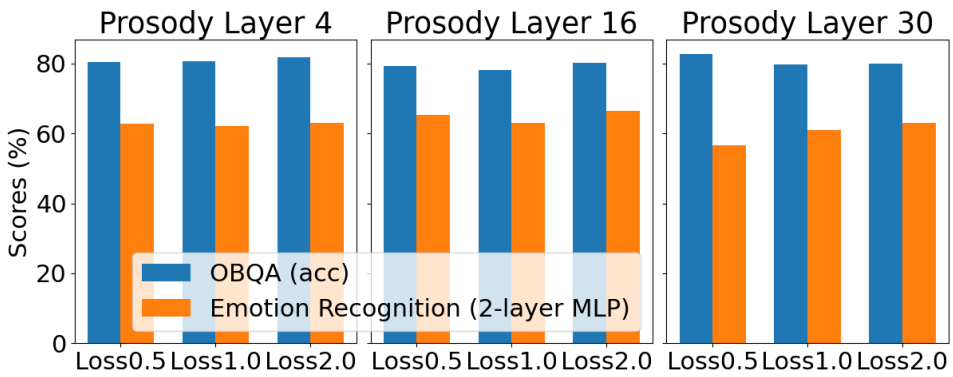

Figure 3: Design choice investigation results for WhisperPro. The blue and orange bars represent OpenBookQA accuracy and the 2-layer MLP probing results on the SER task.

We study two central design choices in WhisperPro. **Prosody layer.** We vary which Whisper decoder layer is used to extract the prosody embedding and test representative early, middle, and late layers (layers 4, 16, and 30 out of 32 decoder layers). The reconstruction is trained on the prosody embeddings produced by the corresponding layer. **Loss ratio (λ).** We vary the weight that balances the ASR loss and the reconstruction loss, considering $\lambda \in \{0.5, 1.0, 2.0\}$. We evaluate each setting on both a semantic task and a prosodic task. For semantic evaluation, we pair the resulting encoder with Qwen2.5-7B-Instruct and measure question-answering accuracy on OpenBookQA. For prosodic evaluation, we use the 2-layer MLP probing setup on SER. As shown in Figure 3, performance is broadly stable across these choices, suggesting that WhisperPro is robust to moderate changes in both the selected prosody layer and the loss ratio.

E.1 Why the 16th Decoder Layer?

We aim to justify the use of the 16th Whisper decoder layer as the source of prosody embeddings. We examine sensitivity to decoder depth and compare encoder-side alternatives to assess whether this choice offers a suitable balance of semantic and paralinguistic information.

The comparison of decoder layers 4, 16, and 30 in Figure 3 shows broadly stable semantic and prosodic performance. We therefore choose layer 16 as a balanced middle-layer default, rather than because it is uniquely optimal. Our motivation is that early decoder layers may be less contextualized and more sensitive to local acoustic variation, whereas late layers are more directly optimized for ASR next-token prediction. An intermediate layer offers a natural compromise between integrating acoustic information through cross-attention and preceding textual context, and specialization toward lexical prediction. This is a motivation for the design; the empirical results suggest that the method is not sensitive to the exact layer choice.

Encoder-side alternatives. We further compare the layer-16 decoder representation with two encoder-side alternatives. **Encoder Global** applies masked mean pooling to the final encoder hidden states over valid audio frames. **Encoder-to-Token Aligned** averages the layer-16 decoder cross-attention weights over attention heads and uses these weights to aggregate the final encoder states for each generated token. Table 8 reports semantic average accuracy and the four AIR-Bench paralinguistic results for the 7B model.

The semantic averages are comparable (81.83% and 82.27% vs. 82.50% for the decoder representation). On AIR-Bench, global encoder pooling performs slightly better on average (66.01% vs. 64.80%), whereas token-aligned encoder features perform slightly worse (63.99%). Encoder-side features thus do not provide a consistent downstream advantage, and the additional alignment procedure does not improve performance in this comparison.

Table 8: Encoder- and decoder-side representations in WhisperPro. Semantic Avg. is the average accuracy on the six semantic benchmarks; the remaining columns report AIR-Bench paralinguistic accuracy. All scores are percentages (↑).

Representation	Semantic Avg.	Emotion	Gender	Age	Accent	Paralinguistic Avg.
Decoder Layer 16	82.50	60.50	88.60	64.90	45.10	64.80
Encoder Global	81.83	59.30	93.22	68.70	42.80	66.01
Encoder-to-Token Aligned	82.27	53.75	93.62	67.30	41.30	63.99

Reconstruction-trained, token-synchronized states. The selected decoder states are not merely off-the-shelf Whisper features optimized for text prediction. In WhisperPro, the Mel reconstruction loss is backpropagated directly through the layer-16 states, encouraging them to retain acoustic and paralinguistic information beyond lexical content. Given comparable downstream performance, natural synchronization with generated text tokens, and no need for an additional alignment procedure, layer-16 decoder states provide a simple and effective default.

F LLM Design Choice Investigations

We investigate the effect of two critical design choices in LLM training.

1) Knowledge Distillation Loss Type: In prior studies that address the modality gap problem, they typically use Kullback-Leibler (KL)-divergence loss for knowledge distillation [11, 31]. Especially, SALAD [11] finds that KL loss achieves significantly better modality alignment results. Therefore, we investigate whether KL loss or cross entropy loss provides better results. Specifically, we use a token-level forward KL objective in which the teacher and student are conditioned on different input sequences:

$$\mathcal{L}_{\mathrm{KL}} = T^2 \sum_{j \in \mathcal{A}} \mathrm{KL}\Big(P_t\Big(\cdot \mid \mathbf{x}^{(t)}_{\le j}\Big) \,\Big\|\, P_s\Big(\cdot \mid \mathbf{x}^{(s)}_{\le j}\Big)\Big), \tag{25}$$

where $\mathcal{A}$ denotes the set of answer-token positions used for distillation, $\mathbf{x}^{(t)}$ and $\mathbf{x}^{(s)}$ denote the teacher and student input sequences, respectively, and T is the distillation temperature. In our setting, the teacher input $\mathbf{x}^{(t)}$ is the clean text-only sequence, while the student input $\mathbf{x}^{(s)}$ is the corresponding sequence containing ASR text tokens and its injected prosody embedding. The predictive distributions are obtained from the teacher and student logits as

$$P_t\Big(\cdot \mid \mathbf{x}^{(t)}_{\le j}\Big) = \mathrm{softmax}\left(\frac{\mathbf{z}^{(t)}_j}{T}\right), \qquad P_s\Big(\cdot \mid \mathbf{x}^{(s)}_{\le j}\Big) = \mathrm{softmax}\left(\frac{\mathbf{z}^{(s)}_j}{T}\right), \tag{26}$$

where $\mathbf{z}^{(t)}_j$ and $\mathbf{z}^{(s)}_j$ are the teacher and student logits at position j. This formulation makes explicit that we distill the teacher's next-token distribution into the student's next-token distribution under different conditioning contexts.

In addition to the KL term, our training objective also includes the standard next-token cross-entropy loss. We balance the two losses using a coefficient $\alpha \in [0, 1]$, yielding the overall objective

$$\mathcal{L} = \alpha \mathcal{L}_{\mathrm{KL}} + (1 - \alpha)\mathcal{L}_{\mathrm{CE}}, \tag{27}$$

where $\mathcal{L}_{\mathrm{CE}}$ denotes the cross-entropy loss on the target tokens and $\mathcal{L}_{\mathrm{KL}}$ is the distillation loss defined above. To study the effect of this trade-off, we evaluate $\alpha \in \{0, 0.5, 1\}$. Here, $\alpha = 0$ corresponds to pure cross-entropy training, $\alpha = 1$ corresponds to pure KL distillation, and $\alpha = 0.5$ gives equal weight to the two objectives. For all distillation experiments, we fix the temperature to $T = 2.0$.

2) Knowledge Distillation Text Token Source: Because WhisperPro internally performs ASR to produce text tokens, an important design question is which text source should be used during distillation. More specifically, we ask whether the student should be trained with ground-truth text, with ASR-transcribed text, or with a mixture in which the student input and teacher target are derived from different text sources. To answer this question, we compare the following three settings:

1. **GTQ_GTA:** The student input uses the original ground-truth text, and the teacher distribution is also computed from the original ground-truth text. This is the cleanest setting, and it corresponds to distillation under an idealized assumption of error-free transcription.

Table 9: Effect of loss weight α and text-token source on MMSU and OBQA (%).

Setting	α	MMSU	OBQA
GTQ_GTA	0.0	67.1	82.9
GTQ_GTA	0.5	63.0	80.9
GTQ_GTA	1.0	66.5	82.2
ASRQ_ASRA	0.0	68.0	84.4
ASRQ_ASRA	0.5	66.5	81.5
ASRQ_ASRA	1.0	63.0	79.8
ASRQ_GTA	0.0	67.6	82.4
ASRQ_GTA	0.5	65.0	81.1
ASRQ_GTA	1.0	63.7	81.1

2. **ASRQ_ASRA:** The student input uses the text transcribed by WhisperPro from the speech-form data, and the teacher distribution is also conditioned on this transcribed text. This setting matches inference-time conditions most closely, because the student is trained on the same type of text tokens it will receive at test time. However, the distillation target is also based on text that may contain ASR errors.
3. **ASRQ_GTA:** The student input uses the text transcribed by WhisperPro, while the teacher distribution is computed from the original ground-truth text. This setting exposes the student to realistic ASR-imperfect inputs, but still provides a clean teacher target. Intuitively, it tests whether the model can learn to recover from typical ASR errors by mapping noisy transcribed inputs toward the behavior induced by the correct text.

For all three settings, the student always receives the same prosody embeddings in addition to the text tokens during training. Additionally, we use only the knowledge distillation datasets (CommonsenseQA and UltraChat) for training. We evaluate all settings on MMSU and OBQA benchmarks. Regardless of the text source used during training, all text tokens at inference time are produced by WhisperPro. As shown in Table 9, we do not observe large performance differences across the three settings. However, the ASRQ_ASRA setting with $\alpha = 0$ achieves the best overall results across the two benchmarks, so we adopt this configuration in the final model training.

G Roles of the Transcript, Semantic Distillation, and Prosody Embeddings

We aim to clarify which components preserve semantic capabilities and which enable paralinguistic understanding. In particular, we test whether the transcript-compatible interface and semantic distillation account for the small semantic modality gap, while prosody embeddings provide complementary non-lexical information.

Tables 10 and 11 isolate the roles of the transcript-compatible interface, semantic knowledge distillation, and prosody embeddings through controlled semantic and paralinguistic evaluations.

Controlled settings. **Original Qwen** is the unmodified Qwen2.5-7B-Instruct model evaluated on text inputs. **Text Semantic Distillation** applies the semantic distillation to text-only Qwen2.5-7B-Instruct using clean text–response data; this control is also referred to as *Text SFT*. **No Prosody Embedding** uses the same data and training protocol as TextPro-SLM, but omits the prosody stream and trains on WhisperPro ASR transcripts with matched ASR-conditioned semantic targets. **Original Whisper** replaces WhisperPro with the original Whisper-large-v3 and trains Qwen using that model's ASR transcripts and layer-16 decoder embeddings, with the same data mixture and training recipe; this is the w/o Recon setting in Table 5. Finally, **w/o Semantic Distillation** removes semantic distillation from TextPro-SLM training while retaining paralinguistic-task training.

The transcript carries most semantic information. Without prosody embeddings, semantic accuracy remains close to the full model (81.9% vs. 82.5%). Replacing WhisperPro with Original Whisper likewise leaves semantic accuracy broadly comparable (82.3% vs. 82.5%). These results suggest that the transcript-compatible interface, rather than the prosody embeddings, is the primary source of content-based QA performance on these benchmarks.

Table 10: Controlled component ablations: semantic accuracy ↑ (%) and gap ↓ (%). Each gap is Original Qwen accuracy minus the row's accuracy; negative values indicate improvement over the text baseline. Original Qwen scores are 89.2%, 70.9%, 85.8%, 96.1%, 89.2%, and 68.1%, respectively (Avg.: 83.2%). Gaps are computed from the reported rounded scores, including the reported averages. For the text-only distillation control, Gap denotes a text-baseline difference, not a speech–text modality gap.

	StoryCloze		MMSU		OBQA		ARC-E		ARC-C		PIQA		Avg.	
Model	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap	Acc.	Gap
Text Semantic Distillation	93.3	-4.1	71.6	-0.7	85.7	0.1	96.8	-0.7	91.6	-2.4	70.9	-2.8	85.0	-1.8
No Prosody Embedding	89.6	-0.4	68.5	2.4	82.4	3.4	95.7	0.4	89.4	-0.2	66.0	2.1	81.9	1.3
Original Whisper	81.9	7.3	67.6	3.3	83.1	2.7	95.8	0.3	88.8	0.4	76.9	-8.8	82.3	0.9
TextPro-SLM-7B w/o Semantic Distillation	84.7	4.5	46.7	24.2	55.8	30.0	89.5	6.6	75.8	13.4	66.6	1.5	69.9	13.3
TextPro-SLM-7B	88.6	0.6	66.7	4.2	85.1	0.7	95.8	0.3	89.4	-0.2	69.7	-1.6	82.5	0.7

Table 11: Controlled component ablations: AIR-Bench paralinguistic accuracy (%, ↑) and the reported four-task average.

Model	Emotion	Gender	Age	Accent	Avg.
No Prosody Embedding	54.10	38.30	29.60	30.50	38.10
Original Whisper	47.25	77.66	55.50	32.20	53.15
TextPro-SLM-7B	60.50	88.60	64.90	45.10	64.80

Semantic distillation preserves and adapts pretrained capabilities. The text-only distillation control achieves 85.0%, compared with 83.2% for Original Qwen. By contrast, removing semantic distillation from TextPro-SLM reduces semantic accuracy from 82.5% to 69.9%. This contrast supports the role of distillation in preventing degradation and adapting the pretrained TLM during speech and paralinguistic training, rather than attributing the reduced modality gap primarily to a large increase in the underlying text model's capability.

Prosody embeddings supply complementary paralinguistic information. Evaluating the same No Prosody Embedding model on AIR-Bench reduces average accuracy from 64.8% to 38.1%, a 26.7-point drop despite the small semantic difference. Original Whisper reaches 53.15% vs. 64.80% for TextPro-SLM, an 11.65-point drop, while maintaining similar semantic accuracy. Thus, transcripts and distillation support semantics, while reconstruction-trained prosody embeddings supply complementary paralinguistic information.

H More Related Works

Several prior works also factorize or decouple different aspects of speech, but they are largely motivated by *generation quality*, *prosody-rich synthesis*, or *prosody control*, rather than by reducing the semantic modality gap between a TLM and an SLM. For example, AudioLM [39], pGSLM [68], SpiritLM [15], Moshi [3], and SpeechGPT-Gen [69] separate semantic/content information from lower-level acoustic, pitch, style, or codec streams so that the model can better generate natural and expressive speech. A related line of work focuses more explicitly on controllable generation. NaturalSpeech 3 [70] factorizes speech into multiple codec subspaces such as content and prosody and then generates them with a cascade of diffusion modules, while ProsodyLM [71] converts each utterance into text together with discrete word-level prosody tokens so that an LLM can better model prosodic phenomena. These approaches show the value of disentangling content from other speech attributes, but their main objective is still to improve spoken generation or prosody control.

Our motivation is different. We are not primarily trying to make the decoder produce more expressive speech; instead, we seek to make spoken *input* look more like the input of a text LLM while preserving paralinguistic information needed for understanding. Concretely, TextPro-SLM uses synchronized text tokens and compact continuous prosody embeddings, so the LLM can retain the semantic strengths of the original TLM while gaining stronger paralinguistic understanding. This differs from

prior token-factorization approaches that represent prosody or acoustics as additional discrete streams or interleaved tokens, which are well suited for generation but can substantially lengthen the sequence and shift the interaction format away from standard TLM processing. In this sense, our method is closer to an input-side alignment strategy: the goal is better semantic preservation under speech input, rather than richer speech synthesis alone.

Comparison with DrVoice. DrVoice [72] is closely related to our input-side perspective: it groups 25 Hz speech representations into a 5 Hz sequence, reducing the temporal-resolution mismatch between speech and text. In contrast, TextPro-SLM explicitly decomposes speech into transcript tokens and compact prosody embeddings, addressing both sequence-length and representation-format mismatches with the native text inputs of the underlying TLM.

On the shared VoiceBench tasks, DrVoice reports 82.20% on OpenBookQA and 67.27% on MMSU, compared with 85.1% and 66.7%, respectively, for TextPro-SLM-7B, indicating broadly comparable performance on these two tasks. These are comparisons of reported results rather than a controlled evaluation under matched training and inference conditions. DrVoice uses approximately 56K hours of speech in post-training and additionally applies DPO and GSPO to improve speech comprehension, instruction following, and reasoning [72]. Such additional optimization can change the capabilities of the underlying text checkpoint, complicating an interpretation solely in terms of modality-gap reduction. By comparison, TextPro-SLM uses roughly 1K hours of LLM training audio with supervised semantic distillation and paralinguistic training, without additional preference optimization or reinforcement learning. These data figures concern model adaptation rather than the total data used to pretrain the underlying components. The results suggest that text–prosody factorization can support competitive speech understanding with limited LLM adaptation data, while the different training pipelines preclude attributing the observed differences solely to the input representation.

I Broader Societal Impact

Our work studies how to reduce the modality gap in SLMs by improving the input-side speech representation rather than relying only on output-side adaptation. A potential positive societal impact of this direction is that it may make speech-based AI systems substantially more semantically capable, allowing users to access stronger reasoning, knowledge use, and instruction-following abilities directly through spoken interaction. In settings such as voice assistants, spoken question answering, accessibility tools, and hands-free interfaces, reducing the modality gap could make speech systems more reliable for general tasks, especially when current models lose important meaning when moving from text input to speech input.

At the same time, making SLMs more semantically powerful can introduce meaningful risks. First, if a model becomes better at extracting task-relevant meaning from speech, users may place greater trust in spoken interfaces even when they still make subtle but consequential mistakes in factual understanding, reasoning, or instruction interpretation; such failures could be especially harmful in high-stakes domains. Second, more capable speech interfaces may accelerate deployment of automated systems in education, customer service, or other communication-heavy settings, potentially amplifying harms from misinformation, over-reliance, or reduced human oversight on a larger scale. Third, although not the primary focus of our work, better use of speech-side cues such as prosody may also increase the possibility that systems infer sensitive speaker attributes or intentions in ways users do not expect, raising privacy and fairness concerns if such capabilities are deployed carelessly.

Table 12: Summary of dataset licenses and whether academic research use is allowed.

Dataset	License	Academic research use
CommonsenseQA	MIT	Yes
LibriSpeech	CC BY 4.0	Yes
UltraChat	MIT	Yes
ParaSpeechCaps	CC BY-NC-SA 4.0	Yes
IEMOCAP	Custom research-only license	Yes
CREMA-D	ODbL 1.0 / DbCL 1.0	Yes
SAVEE	Custom institutional license	Yes
TESS	CC BY-NC-ND 4.0	Yes
ESD	Custom non-commercial license	Yes
Common Voice	CC0	Yes

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's core claim: the remaining modality gap is best addressed from the input side. To support this claim, we design and train both a speech encoder and an LLM backbone, and our experiments show that this approach achieves the lowest modality gap among the compared baselines that try to address the gap only from the output side.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitation section is presented in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be

used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include theoretical results or proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the training details of WhisperPro in Section 3.1 and the training details of the LLM backbone in Section 3.2. We also summarize all training data used for TextPro-SLM and LLM training in Table 1, and describe the benchmarks and evaluation protocols in Section 4 and Appendices C and D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code for model training, model inference, and evaluation are provided in the supplemental material.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the training details of WhisperPro in Section 3.1 and the training details of the LLM backbone in Section 3.2 as well as in the code.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Running the full set of experiments multiple times is extremely time-consuming.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Details of the computing resources, particularly the CPU and GPU types, cannot be disclosed because of institutional restrictions.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have carefully checked the NeurIPS Code of Ethics and confirm that the research conducted in this paper conform with the Code of Ethics in every respect.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses both potential positive and negative societal impacts in Appendix I.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work is mainly for preserving the original semantic capabilities of the base LLMs, so we do not pose new risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited all the code, data, and models used in this paper, and we have listed out all the license of the dataset we use in Table 12 to confirm all datasets are allowed to be used in academic research.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The only new assets introduced in this paper is the newly created accent benchmark dataset, and it has been documented in Appendix D.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [No]

    Justification: LLMs are used only for language editing purposes.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.