Dual-Encoder Fusion with Explicit and Implicit Injection for the Interspeech 2026 Audio Encoder Capability Challenge

Yucong Zhang [1,2], *Zhang Chen* [5], *Juan Liu* [3,1,**], *Wei Ju*[4], *Hongbin Suo*[4], *Ming Li* [2,3,**]

[1] School of Computer Science, Wuhan University, Wuhan, China
[2] School of Artificial Intelligence, The Chinese University of Hong Kong, Shenzhen, China
[3] School of Artificial Intelligence, Wuhan University, Wuhan, China
[4] AI Center, OPPO, Beijing, China
[5] Digital Innovation Research Center, Duke Kunshan University, Kunshan, China

yucong.zhang@whu.edu.cn,mingli369@cuhk.edu.cn

Abstract

We conduct a systematic study of dual-encoder fusion for the Interspeech 2026 Audio Encoder Capability Challenge (AECC), aiming to translate encoder complementarity into consistent gains across various audio classification and understanding tasks. Using Whisper and Dasheng as complementary encoders, we examine two injection strategies: implicit injection via parameter-efficient Dasheng adaptation before fusion, and explicit injection that decomposes representations into residual components with auxiliary regularization to isolate non-redundant information. Ablations show that explicit injection yields stronger task-wise complementarity and more per-task best results, while implicit adaptation remains competitive and robust overall. We also identify a stable backbone based on token-wise softmax-gated residual fusion with a lightweight STFT residual branch. These results provide practical design insights for modular encoder fusion in the Large Audio Language Model (LALM) framework.

Index Terms: dual-encoder fusion, audio encoder evaluation, large audio language models, mixture-of-experts, residual injection

1. Introduction

LALMs have recently emerged as a unified framework for solving heterogeneous audio tasks through a standardized end-to-end pipeline [1, 2, 3]. In this paradigm, a pre-trained audio encoder transforms waveforms into frame-level representations that are subsequently consumed by a language-model backbone for both classification-style evaluation and higher-level audio understanding. As the encoder serves as the sole interface between raw signals and the language model, its representational bias directly constrains downstream capability [4, 5, 6, 7]. Previous challenges have shown that single encoder cannot always be uniformly optimal across diverse task families [8, 9], motivating the study of encoder complementarity under a unified evaluation setting.

Current pre-trained encoders are often optimized with different supervision objectives and inductive biases, leading to distinct representational strengths. For example, Whisper [5] is trained with large-scale weakly supervised speech transcription and excels at capturing speech-aligned linguistic content. In contrast, Dasheng [4] is trained with masked audio modeling at scale and is designed to encode broader acoustic semantics beyond speech. These differences suggest potential complementarity: speech-oriented semantic alignment from Whisper may benefit understanding tasks, while general acoustic representations from Dasheng may enhance acoustics-driven classification tasks. However, effectively integrating such complementary information requires mechanisms that balance synergy and redundancy.

Prior work has explored combining multiple pre-trained encoders in related domains. Feature-level aggregation of self-supervised speech models has shown measurable gains in downstream tasks [10]. More recently, Mixture-of-Experts (MoE) [11] routing has been introduced to adaptively combine multiple audio encoders for large-model pipelines [12], enabling input-dependent specialization [13, 14]. While these approaches demonstrate the potential of encoder fusion, less attention has been given to systematically analyzing how complementary information of different audio encoders is injected, preserved, and controlled within a unified LALM evaluation framework.

In this work, we conduct a systematic study of dual-encoder fusion under the unified LALM framework. We first compare lightweight fusion operators and identify a stable softmax-gated backbone augmented with an STFT residual branch. We then analyze two injection mechanisms—implicit adaptation via LoRA [15] and explicit residual decomposition with auxiliary regularization—to understand how complementary information is introduced and preserved. Through controlled ablations, we derive practical insights into translating encoder complementarity into consistent gains across various classification and audio understanding tasks. The model checkpoints and codes for both implicit[1] and explicit[2] fusion are open-sourced.

2. Methods

2.1. Problem Setup and Notation

AECC evaluates an audio encoder as the front-end of a LALM under a standardized pipeline. We study dual-encoder fusion of Whisper-Base and Dasheng-Base, aiming to preserve Whisper's speech-aligned semantics while injecting complementary acoustic knowledge from Dasheng, under a fixed interface.

Given a 16-kHz waveform x, Whisper and Dasheng produce frame-level token sequences

$$H_w \in \mathbb{R}^{T\times 512}, \quad H_d \in \mathbb{R}^{T\times 768}, \tag{1}$$

[**] indicates the corresponding author.

[1] https://huggingface.co/yucongzh/implicit_fusion

[2] https://huggingface.co/yucongzh/explicit_fusion

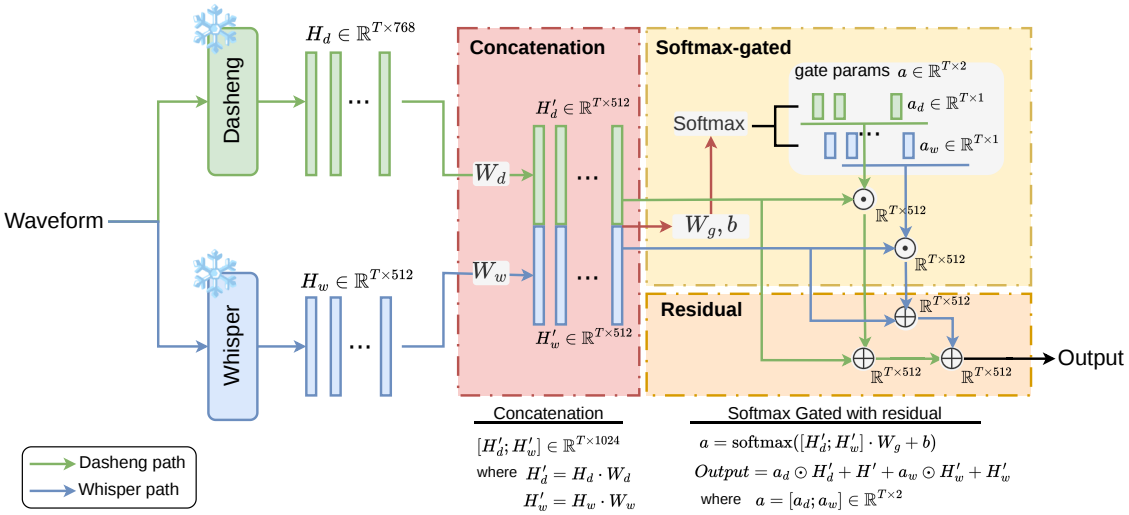

Figure 1: *System overview for the proposed softmax-gated residual fusion for dual encoders. W_d, W_w, W_g and b are learnable parameters. "$\odot$" and "$\oplus$" denotes element-wise multiplication and summation respectively.*

where both streams operate at 25 fps and are aligned to a common valid-frame mask. We map both streams into a fusion space of dimension d=512 via lightweight projections and output a fused sequence $Z \in \mathbb{R}^{T\times 512}$, which is directly consumed by the downstream pipeline.

2.2. Fusion Operators

We evaluate three front-end fusion operators that combine (H_w, H_d) into Z.

Concatenation (baseline)
We form $H_{\text{cat}} = [H_w; H_d] \in \mathbb{R}^{T\times 1280}$ and apply a linear projection to 512 dimensions. This baseline preserves both streams but does not explicitly control redundancy.

Mixture-of-Experts (MoE) routing
We implement a sparse Mixture-of-Experts (MoE) projector with E experts and top-k routing. The router operates on a token-wise fusion input x_t, whose form depends on the selected MoE input mode. We consider two variants:

- **MoE-concat:** the router input is the raw concatenation $x_t = [H_{w,t}; H_{d,t}] \in \mathbb{R}^{1280}$, preserving the full encoder dimensionality prior to expert routing.
- **MoE-proj:** both streams are first projected into a shared 512-dimensional fusion space, and the router input is $x_t = [d_t; w_t]$, where $d_t = W_d H_{d,t}$ and $w_t = W_w H_{w,t}$.

The router produces logits $r_t = g(x_t) \in \mathbb{R}^E$, probabilities $p_t = \text{softmax}(r_t)$, and selects the top-$k$ experts $\mathcal{K}_t$. The MoE output is computed as

$$y_t = \sum_{e\in\mathcal{K}_t} p_{t,e} f_e(x_t), \quad z_t = W_o y_t, \tag{2}$$

where each expert $f_e(\cdot)$ is a two-layer feed-forward network and W_o maps the aggregated expert output to the 512-dimensional fusion space. Each expert f_e is instantiated with input dimensionality matching the selected mode. We apply a standard load-balancing loss to encourage balanced expert utilization.

Softmax-gated residual fusion (backbone)
As a stable and lightweight alternative to sparse routing, we adopt a token-wise two-way softmax gate combined with residual aggregation. Let $H_{w,t} \in \mathbb{R}^{512}$ and $H_{d,t} \in \mathbb{R}^{768}$ denote Whisper and Dasheng features at frame t. We first project both streams into a shared 512-dimensional fusion space:

$$d_t = W_d H_{d,t}, \quad w_t = W_w H_{w,t}, \tag{3}$$

where $W_d \in \mathbb{R}^{512\times 768}$ and $W_w \in \mathbb{R}^{512\times 512}$ are learnable linear projections.

We then compute gating logits via

$$a_t = \text{softmax}(W_g[d_t; w_t] + b_g) \in \mathbb{R}^2, \tag{4}$$

where $W_g \in \mathbb{R}^{2\times 1024}$ and $b_g \in \mathbb{R}^2$. The gated mixture is defined as

$$m_t = a_{t,1} d_t + a_{t,2} w_t. \tag{5}$$

Finally, the fused representation is formed through residual aggregation followed by Layer Normalization:

$$z_t = \text{LN}\big(\text{LN}(m_t) + d_t + w_t\big), \tag{6}$$

where $\text{LN}(\cdot)$ denotes token-wise Layer Normalization. This design preserves both encoder streams through the residual path while enabling adaptive, frame-wise weighting via the softmax gate, thereby reducing the risk of suppressing complementary information.

2.3. STFT residual branch

In addition to representation-level fusion, we incorporate a lightweight signal-level residual branch based on short-time Fourier transform (STFT) features.

Given the input waveform x, we compute its log-magnitude STFT representation $S \in \mathbb{R}^{T\times F}$, where T denotes time frames aligned with the encoder outputs and F denotes frequency bins.

The spectral features are projected into the 512-dimensional fusion space:

$$s_t = W_s S_t, \tag{7}$$

where $W_s \in \mathbb{R}^{512\times F}$ is a learnable linear projection.

The projected spectral residual is then added to the fused representation with a learnable scaling factor:

$$z_t = z_t + \gamma s_t, \tag{8}$$

where γ is a scalar parameter initialized to a small value. This branch injects fine-grained frequency cues that may be underrepresented in pre-trained encoder embeddings, while keeping the overall fusion interface unchanged.

Table 1: *Performance comparison using different fusion operators (per-subtask and overall). Columns are grouped by fusion operator, with and without the STFT residual branch.*

	Single		Baseline	Softmax-gated		MoE-concat		MoE-proj	
Tasks	Whisper	Dasheng	Concat	w/o STFT	w/ STFT	w/o STFT	w/ STFT	w/o STFT	w/ STFT
asvspoof2015	0.943	0.937	0.937	**0.959**	**0.959**	0.947	0.954	0.950	0.946
cremad	0.516	**0.621**	**0.621**	0.574	0.598	0.531	0.576	0.594	0.582
esc-50	0.635	**0.755**	0.743	0.652	0.715	0.713	0.715	0.705	0.730
fluentspeechcommands [16]	0.817	**0.984**	0.961	0.905	0.947	0.822	0.795	0.842	0.858
freemusicarchive	0.579	0.429	0.539	0.570	0.583	**0.600**	0.569	0.569	0.556
fsd50k	0.092	0.063	0.092	0.093	0.103	0.102	0.109	0.110	**0.111**
fsdkaggle2018	0.552	0.415	0.531	0.539	0.539	**0.571**	0.561	0.527	0.569
gtzan	0.697	0.323	0.687	0.737	0.707	**0.798**	**0.798**	0.758	0.788
libricount	0.409	0.386	0.453	0.445	0.448	**0.459**	0.420	0.316	0.441
nsynth	0.638	0.675	0.648	0.631	0.685	**0.703**	0.698	0.693	0.678
speechcommandsv1	0.694	0.655	0.708	0.719	0.751	**0.825**	0.806	0.794	0.789
urbansound8k	0.737	**0.829**	0.785	0.781	0.803	0.783	0.797	0.798	0.812
vocalsound	0.867	0.855	0.891	0.879	0.887	0.896	0.898	**0.899**	0.898
voxceleb1	0.762	**0.974**	0.938	0.936	0.932	0.822	0.841	0.821	0.798
voxlingua33	0.835	0.311	0.836	0.835	0.855	0.879	**0.887**	0.883	0.872
Track A Overall	0.652	0.614	0.691	0.684	**0.701**	0.697	0.695	0.684	0.695
aishell-1	**0.361**	0.018	0.357	0.346	0.359	0.345	0.337	0.336	0.345
clotho	**0.358**	0.207	0.356	0.341	0.344	0.355	0.354	0.352	0.354
librispeech	0.385	0.103	0.385	0.381	**0.396**	0.367	0.375	0.354	0.377
mecat	0.624	0.600	0.631	0.614	0.626	**0.633**	0.632	0.625	0.626
songdescriber	0.448	0.410	0.500	0.489	0.486	0.472	0.477	0.495	**0.510**
Track B Overall	0.400	0.270	**0.446**	0.434	0.442	0.434	0.435	0.433	0.442

Table 2: *Training data ratios under the official "all" recipe.*

Dataset	Sampling Ratio (%)
ESC-50 [17]	0.453
SpeechCommandsV1 [18]	6.154
FluentSpeechCommands [16]	2.717
FSDKaggle2018 [19]	2.378
VoxCeleb1 [20]	4.529
NSynth [21]	3.692
AISHELL-1 [22]	6.154
CremaD [23]	0.679
SongDescriber [24]	0.226
ASVSpoof2015 [25]	2.038
FreeMusicArchive [26]	7.246
FSD50k [27]	1.019
UrbanSound8K [28]	1.019
GTZAN [29]	1.019
VocalSound [30]	2.378
Clotho [31]	2.462
LibriSpeech [32]	6.154
LibriCount [33]	0.453
VoxLingua33 [34]	24.615
MECAT [35]	24.615

2.4. Injection Mechanisms

With the fusion operator fixed to the softmax-gated residual backbone in Sec. 2.2, we study how complementary information is made available to the fusion module and retained during optimization. We consider two injection mechanisms.

2.4.1. Implicit injection via Dasheng adaptation.

In the implicit route, complementarity is introduced by adapting Dasheng to the AECC training mixture before fusion. Concretely, we perform parameter-efficient LoRA adaptation on Dasheng while keeping Whisper frozen, and then merge the LoRA weights into the Dasheng backbone. During subsequent fusion training, both backbones are frozen and only fusion-related parameters are updated. This two-stage protocol isolates the effect of injection from changes in the downstream pipeline and improves training stability by preventing co-adaptation of the two backbones.

2.4.2. Explicit injection via residual decomposition and regularization.

In the explicit route, we construct a complementary residual from Dasheng and inject it into Whisper. Specifically, we first predict the Dasheng component that is predictable from Whisper using a linear map

$$\hat{H}_d = H_w W_{w\to d}, \tag{9}$$

and define the residual

$$R_d = H_d - \hat{H}_d. \tag{10}$$

We then project R_d into the fusion space and fuse it with Whisper tokens using the same softmax-gated residual operator, producing the fused sequence Z. This design makes the injected information explicit and separates shared versus complementary components at the representation level.

To encourage non-redundant complementarity, we further apply auxiliary regularization on valid frames. First, we penalize residual energy to avoid unstable residual growth,

$$L_{\text{recon}} = \mathbb{E}_{t\in\Omega} \|R_{d,t}\|_2^2, \tag{11}$$

where Ω denotes the set of valid (non-padded) frames. Second, we enforce decorrelation between Whisper features and a projected residual via a cross-covariance penalty. Let $u_t = P_w(H_{w,t})$ and $v_t = P_r(R_{d,t})$ be projected features; we compute the cross-covariance over valid frames and minimize its Frobenius norm:

$$C = \frac{1}{|\Omega|}\sum_{t\in\Omega}(u_t-\bar{u})(v_t-\bar{v})^\top, \qquad L_{\text{decor}} = \|C\|_F^2. \tag{12}$$

The auxiliary term is added to the task loss as

$$L = L_{\text{task}} + \lambda_{\text{recon}} L_{\text{recon}} + \lambda_{\text{decor}} L_{\text{decor}}. \tag{13}$$

These regularizers are applied only during fusion training and introduce no overhead at inference.

3. Experiment

3.1. Experimental settings

Training details

All experiments are conducted under the official AECC evaluation pipeline based on XARES-LLM[3]. For the implicit injection, we adopt a two-stage training schedule and adjust the

[3]https://github.com/xiaomi-research/xares-llm

Table 3: *Performance comparison using different injection mechanisms under the same backbone with softmax-gated fusion operator and STFT residual. The baseline column is taken from softmax-gated fusion operator with STFT residual from Table 1. Mean±std columns are taken from performance results from three different servers.*

Tasks	Baseline	Explicit	Implicit
asvspoof2015	0.959	**0.960 ± 0.010**	0.959 ± 0.004
cremad	0.598	0.619 ± 0.001	**0.632 ± 0.004**
esc-50	0.715	0.721 ± 0.022	**0.763 ± 0.008**
fluentspeechcommands	**0.947**	0.895 ± 0.002	0.933 ± 0.007
freemusicarchive	**0.583**	0.566 ± 0.011	0.580 ± 0.012
fsd50k	0.103	**0.111 ± 0.001**	0.103 ± 0.001
fsdkaggle2018	0.539	**0.547 ± 0.005**	**0.547 ± 0.004**
gtzan	0.707	0.744 ± 0.024	**0.768 ± 0.029**
libricount	**0.448**	0.407 ± 0.011	0.430 ± 0.009
nsynth	0.685	**0.694 ± 0.006**	0.686 ± 0.009
speechcommandsv1	0.751	0.776 ± 0.005	**0.786 ± 0.002**
urbansound8k	**0.803**	0.797 ± 0.005	0.801 ± 0.004
vocalsound	0.887	**0.892 ± 0.004**	0.889 ± 0.001
voxceleb1	**0.932**	0.860 ± 0.004	0.872 ± 0.003
voxlingua33	0.855	**0.868 ± 0.007**	0.860 ± 0.003
Track A Overall	0.701	0.706 ± 0.001	**0.712 ± 0.001**
aishell-1	**0.359**	0.357 ± 0.002	0.354 ± 0.001
clotho	0.344	**0.361 ± 0.001**	0.349 ± 0.003
librispeech	**0.396**	0.390 ± 0.004	0.393 ± 0.002
mecat	0.626	**0.630 ± 0.002**	0.627 ± 0.001
songdescriber	0.486	0.484 ± 0.004	**0.497 ± 0.006**
Track B Overall	0.442	**0.446 ± 0.002**	0.445 ± 0.001

default parameter-freezing policy to enable encoder fusion. In Stage 1, we update only the LoRA adapters attached to Dasheng while keeping the Dasheng backbone and Whisper fully frozen. After Stage 1, LoRA weights are merged into the Dasheng backbone. In Stage 2, we freeze both encoders and optimize only fusion-related components.

For all the fusion model fine-tuning, we use AdamW with weight decay 0.01, batch size 8, and maximum gradient norm 1.0. For implicit injection, we LoRA-adapt Dasheng for 10k steps with learning rate 1×10^{-4} and 200 warmup steps. LoRA hyperparameters are $r = 16$, $\alpha = 32$, dropout 0.05, and target modules $\texttt{qkv}$. For others, we train only fusion-related parameters for 100k steps while freezing both encoders, using learning rate 5×10^{-5} with 10k warmup steps. STFT residual uses $n\_fft = win\_length = 1024$, $hop\_length = 640$, $n\_mels = 128$ (frame-rate matched to Whisper/Dasheng); γ is initialized to $1\mathrm{e}{-2}$ and learned;MoE uses 8 experts with $top - k = 2$ and load-balance weight $1\mathrm{e}{-2}$.

Training data recipe
We use the official "all" data recipe provided by XARES-LLM and do not change dataset composition or sampling ratios due to time constraints. The "all" recipe covers the training sets of all classification and understanding tasks. Table 2 summarizes the sampling ratios.

Evaluation protocol and benchmarking
We report results using the official AECC scoring scripts. Track A consists of multiple classification benchmarks, and Track B includes speech recognition and audio-language understanding benchmarks. For each track, we present per-subtask scores and the official overall score computed by the pipeline (see Table 1 and 3).

3.2. Fusion Operator Analysis

Table 1 compares several fusion operators under a unified interface. Overall, the softmax-gated fusion provides a more reliable accuracy–robustness trade-off than MoE routing, and further benefits from the STFT residual branch.

Softmax-gated fusion vs. MoE routing
Softmax-gated fusion achieves stronger and more consistent overall performance than MoE variants when considering both tracks. MoE can excel on specific subtasks, suggesting that conditional specialization is sometimes effective; however, these peaks do not consistently translate into higher overall scores. We attribute this to the heterogeneity of subtasks and the higher optimization sensitivity of sparse routing. In contrast, the softmax gate performs a token-wise weighted averaging of the two streams while preserving both encoder representations through the residual path, which stabilizes optimization and reduces the risk of suppressing either representation.

Effect of the STFT residual branch
Adding the STFT residual consistently improves the softmax-gated operator, yielding a clear gain on Track-A overall while keeping Track-B essentially unchanged. This suggests that the spectral branch contributes complementary cues not fully captured by encoder embeddings—particularly fine-grained frequency evidence that is useful for acoustics-driven classification. Since the STFT branch injects signal-level information through a lightweight projection and a learnable scaling, it can enhance such cues without forcing either pre-trained encoder to deviate from its representation space, which likely explains the stable gains observed in Table 1.

3.3. Injection Mechanism Analysis

Table 3 compares two injection mechanisms on top of the same backbone (softmax-gated residual fusion with an STFT residual)The results suggest a trade-off between overall robustness and task-specific peak gains.

Implicit injection achieves the best Track-A overall with consistently low variance, driven by broad improvements on several acoustics-centric classification tasks (CREMA-D, ESC-50, GTZAN, SpeechCommandsV1). However, its benefits are not uniform: on some subtasks the baseline remains competitive or better (FluentSpeechCommands, LibriCount, VoxCeleb1), indicating that encoder-side adaptation does not always align with the most task-relevant cues. Explicit injection yields a slightly lower Track-A overall but attains the best result on more Track-A subtasks, reflecting stronger task-specific complementarity. It also achieves the best Track-B overall (0.446±0.002), with gains on Clotho and MECAT, while staying close to the baseline in other tasks. Beyond empirical scores, explicit injection provides a controllable residual interface that can naturally extend to multi-encoder settings.

4. Conclusion

This paper analyzes dual-encoder fusion for AECC for the LALM framework and summarizes practical design choices for combining Whisper and Dasheng. We introduced two fusion strategies: implicit injection via parameter-efficient Dasheng adaptation and explicit injection via residual decomposition with auxiliary regularization. Empirically, explicit injection yields stronger task-specific complementarity and more per-task best results, suggesting explicit injection as a practical and extensible option, while implicit injection provides a competitive and robust alternative. In addition, our ablations indicate that a token-wise softmax-gated fusion operator with an STFT residual branch is a strong and stable backbone. Future work includes improving the data mixture recipe to better match the target evaluation distribution and further strengthening explicit injection with more expressive residual predictors and regularization.

5. Acknowledgments

This research is funded in part by the National Natural Science Foundation of China (62571223) and Yangtze River Delta Science and Technology Innovation Community Joint Research Project (2024CSJGG01100) and OPPO. Many thanks for the computational resource provided by the Advanced Computing East China Sub-Center.

6. Generative AI Use Disclosure

Generative AI tools were used for limited language editing purposes, including improving clarity and correcting grammatical issues. No substantive content, analysis, or conclusions were generated by AI. The authors remain fully responsible for the content of this manuscript.

7. References

[1] S. Ghosh, Z. Kong, S. Kumar, S. Sakshi, J. Kim, W. Ping, R. Valle, D. Manocha, and B. Catanzaro, "Audio flamingo 2: An audio-language model with long-audio understanding and expert reasoning abilities," in *Proc. ICML*, 2025.

[2] D. Ding, Z. Ju, Y. Leng, S. Liu, T. Liu, Z. Shang, K. Shen, W. Song, X. Tan, H. Tang *et al.*, "Kimi-audio technical report," *arXiv preprint arXiv:2504.18425*, 2025.

[3] Y. Chu, J. Xu, Q. Yang, H. Wei, X. Wei, Z. Guo, Y. Leng, Y. Lv, J. He, J. Lin *et al.*, "Qwen2-audio technical report," *arXiv preprint arXiv:2407.10759*, 2024.

[4] H. Dinkel, Z. Yan, Y. Wang, J. Zhang, Y. Wang, and B. Wang, "Scaling up masked audio encoder learning for general audio classification," in *Proc. Interspeech*, 2024, pp. 547–551.

[5] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. ICML*, 2023, pp. 28 492–28 518.

[6] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Proc. NeurIPS*, vol. 33, pp. 12 449–12 460, 2020.

[7] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, "Data2vec: A general framework for self-supervised learning in speech, vision and language," in *Proc. ICML*, 2022, pp. 1298–1312.

[8] J. Zhang, H. Dinkel, Y. Niu, C. Liu, S. Cheng, A. Zhao, and J. Luan, "X-ARES: A Comprehensive Framework for Assessing Audio Encoder Performance," in *Proc. Interspeech*, 2025, pp. 4868–4872.

[9] J. Turian, J. Shier, H. R. Khan, B. Raj, B. W. Schuller, C. J. Steinmetz, C. Malloy, G. Tzanetakis, G. Velarde, K. McNally *et al.*, "Hear: Holistic evaluation of audio representations," in *Proc. NeurIPS*, 2022, pp. 125–145.

[10] A. Arunkumar, V. Nileshkumar Sukhadia, and S. Umesh, "Investigation of Ensemble features of Self-Supervised Pretrained Models for Automatic Speech Recognition," in *Proc. Interspeech*, 2022, pp. 5145–5149.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[12] S. Mu and S. Lin, "A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications," *arXiv preprint arXiv:2503.07137*, 2025.

[13] W. Shan, Y. Li, Y. Zhang, Y. Luo, C. Xu, X. Zhao, L. Meng, Y. Lu, M. Zhang, H. Yang, T. Xiao, and J. Zhu, "Enhancing speech large language models with prompt-aware mixture of audio encoders," in *Proc. EMNLP*, 2025, pp. 19 305–19 320.

[14] X. Han, H. Nguyen, C. Harris, N. Ho, and S. Saria, "Fusemoe: Mixture-of-experts transformers for fleximodal fusion," in *Proc. NeurIPS*, vol. 37, 2024, pp. 67 850–67 900.

[15] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." in *Proc. ICLR*, 2022.

[16] L. Lugosch, M. Ravanelli, P. Ignoto, V. S. Tomar, and Y. Bengio, "Speech Model Pre-Training for End-to-End Spoken Language Understanding," in *Proc. Interspeech*, 2019, pp. 814–818.

[17] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proc. ACM MM*, 2015, pp. 1015–1018.

[18] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[19] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," in *Proc. DCASE*, 2018.

[20] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, p. 101027, 2020.

[21] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. ICML*, 2017, pp. 1068–1077.

[22] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *Proc. O-COCOSDA*, 2017, pp. 1–5.

[23] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma, "Crema-d: Crowd-sourced emotional multimodal actors dataset," *IEEE transactions on affective computing*, vol. 5, no. 4, pp. 377–390, 2014.

[24] I. Manco, B. Weck, S. Doh, M. Won, Y. Zhang, D. Bogdanov, Y. Wu, K. Chen, P. Tovstogan, E. Benetos, E. Quinton, G. Fazekas, and J. Nam, "The song describer dataset: a corpus of audio captions for music-and-language evaluation," in *Proc. NeurIPS Workshop*, 2023.

[25] Z. Wu, T. Kinnunen, N. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *Proc. Interspeech*, 2015, pp. 2037–2041.

[26] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," *arXiv preprint arXiv:1612.01840*, 2016.

[27] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2021.

[28] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. ACM MM*, 2014, pp. 1041–1044.

[29] B. L. Sturm, "The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use," *arXiv preprint arXiv:1306.1461*, 2013.

[30] Y. Gong, J. Yu, and J. Glass, "Vocalsound: A dataset for improving human vocal sounds recognition," in *Proc. ICASSP*, 2022, pp. 151–155.

[31] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: An audio captioning dataset," in *Proc. ICASSP*, 2020, pp. 736–740.

[32] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[33] F.-R. Stöter, S. Chakrabarty, E. Habets, and B. Edler, "Libricount, a dataset for speaker count estimation," 2018.

[34] J. Valk and T. Alumäe, "Voxlingua107: a dataset for spoken language recognition," in *Proc. SLT*, 2021, pp. 652–658.

[35] Y. Niu, T. Wang, H. Dinkel, X. Sun, J. Zhou, G. Li, J. Liu, X. Liu, J. Zhang, and J. Luan, "Mecat: A multi-experts constructed benchmark for fine-grained audio understanding tasks," *arXiv preprint arXiv:2507.23511*, 2025.