# MoLN: Mixture of Layer Normalization Experts for Speech Deepfake Detection

**Anonymous authors**
Paper under double-blind review

## Abstract

Speech deepfake detection (SDD) requires strong generalization ability due to the diverse and heterogeneous factors affecting speech generation, transmission, and recording conditions. Recent SDD systems increasingly leverage large pre-trained speech encoders and parameter-efficient fine-tuning (PEFT) to adapt transferable speech representations, with LoRA-based approaches becoming a common adaptation strategy. In this work, we investigate LayerNorm (LN) tuning as an alternative adaptation interface for generalizable SDD. Different from LoRA, LN tuning performs compact channel-wise feature recalibration, providing a favorable trade-off between parameter efficiency and generalization while better preserving pre-trained representations. We further show that a single shared LN is insufficient to model the heterogeneous and overlapping conditions in SDD. To address this limitation, we propose **Mixture of LayerNorm Experts (MoLN)**, a parameter-efficient framework that extends LN tuning from a single affine parameter to multiple lightweight LN experts. MoLN employs a frame-level soft router to dynamically combine expert-specific scale and shift parameters, enabling adaptive feature recalibration without requiring explicit domain labels or additional task-specific modules. Extensive experiments demonstrate that MoLN consistently improves cross-dataset generalization over state-of-the-art baselines with fewer trainable parameters.

## 1 Introduction

Speech deepfake detection (SDD) aims to distinguish bona fide speech from synthetic or manipulated audio (Tak et al., 2022b; Guo et al., 2024). Compared with many image- or text-based detection tasks, SDD is influenced by a broader range of factors, including the generation, vocoder, codec, speaker, and linguistic content. The joint variation of these factors gives rise to highly diverse and heterogeneous data distributions, making it difficult for a detector trained on limited data to cover all possible conditions. To address such distributional diversity, an effective SDD model must possess strong generalization ability (Zhang et al., 2021), meaning that it should not only perform well on the training distribution but also accurately recognize new samples that were never observed during training. Therefore, improving model generalization is of fundamental importance to robust SDD.

Recent SDD systems increasingly rely on large pre-trained speech encoders to improve generalization across diverse speech conditions. Large pre-trained speech encoders (Baevski et al., 2020; Babu et al., 2022; Chen et al., 2022) learn transferable acoustic and linguistic representations from large-scale and diverse speech or audio corpora, providing rich prior knowledge for downstream SDD. To adapt these models efficiently to downstream tasks, parameter-efficient fine-tuning (PEFT) updates only a small set of task-specific parameters while keeping most of the pre-trained backbone frozen. Existing PEFT studies for SDD have primarily explored low-rank adaptation (LoRA), including both conventional LoRA (Hu et al., 2022; Laakkonen et al., 2026) and mixtures of LoRA experts (Laakkonen et al., 2025; Pan et al., 2025), demonstrating the potential of lightweight adaptation for SDD.

In this paper, we investigate LayerNorm (LN) tuning as a more suitable adaptation interface than LoRA for generalizable SDD. Different from LoRA, LN tuning only adjusts channel-wise scale and shift parameters, providing a substantially more compact form of feature recalibration. Recent studies (Zhao et al., 2024; Chen et al., 2024) suggest that it can also offer favorable robustness under distribution shifts. We hypothesize that this constrained adaptation can better preserve the transferable feature structure learned during pre-training while retaining sufficient capacity for downstream SDD.



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

To examine this hypothesis, we compare LN tuning with LoRA at ranks $r \in \{1, 2, 4, 8, 16\}$ on trainable parameters, in-distribution (ID) and out-of-distribution (OOD) performance, and representation similarity measured by centered kernel alignment (CKA) and our defined relative feature basis preservation (rFBP). Our results show that LN tuning achieves a more favorable parameter efficiency and OOD generalization trade-off while better preserving prior transferable knowledge, supporting LN as an effective adaptation interface for generalizable SDD.

Meanwhile, a single shared affine parameter in LN may still lack sufficient flexibility to accommodate heterogeneous speech conditions. Conventional LN tuning applies the same affine adaptation to all inputs, although different spoofing conditions may favor different feature recalibrations. Our gradient analysis reveals that adaptation requirements across different conditions exhibit structured variation rather than arbitrary changes, suggesting that they can be captured by a small set of reusable affine parameters. This observation motivates extending LN tuning from a single shared parameter to multiple complementary parameters that can be adaptively combined according to the input.

To this end, we propose **Mixture of LayerNorm Experts (MoLN)**, a novel parameter-efficient adaptation method for generalizable SDD. MoLN introduces multiple LN experts, each parameterized by a residual scale and shift parameter, to capture diverse adaptation patterns under heterogeneous speech conditions. Specifically, a frame-level soft router predicts mixture weights over these experts according to the input representation, enabling each frame to adaptively combine complementary normalization parameters. To promote expert diversity and avoid imbalanced expert utilization, we further introduce orthogonality and load balancing regularization. By increasing adaptation flexibility within the normalization space, MoLN retains the compactness of LN tuning and its ability to preserve pre-trained representations, while better accommodating heterogeneous and overlapping spoofing conditions without requiring explicit domain labels or test-time updates. Overall, MoLN provides a flexible and lightweight adaptation mechanism for improving generalization of SDD.

Our main contributions can be summarized as follows:

- We investigate LN tuning for SDD and show that it achieves a favorable generalization and parameter efficiency trade-off while better preserving pre-trained representations than LoRA. To the best of our knowledge, it is the first systematic study of LN tuning for SDD.
- We propose **Mixture of LayerNorm Experts (MoLN)**, which improves generalization by dynamically combining multiple lightweight LN experts through soft routing for SDD.
- We conduct extensive experiments across multiple SDD datasets, demonstrating that MoLN consistently outperforms various state-of-the-art methods with fewer trainable parameters.

## 2 Related Works

### 2.1 Speech Deepfake Detection

Recent speech deepfake detection (SDD) utilizes large pretrained self-supervised speech foundation models as powerful frontend feature extractors to transfer broad prior knowledge. Such models as wav2vec 2.0 (Baevski et al., 2020), HuBERT (Hsu et al., 2021), XLS-R (Babu et al., 2022), and WavLM (Chen et al., 2022) have been retrained with SDD datasets (Wang & Yamagishi, 2022; Tak et al., 2022b; Guo et al., 2024; Lv et al., 2022; El Kheir et al., 2025) and achieved strong performances.

While full fine-tuning introduces substantial adaptation cost, motivating parameter-efficient fine-tuning (PEFT) for SDD. Recent PEFT-based SDD methods have explored LoRA-style low-rank adaptation (Laakkonen et al., 2026) implemented with meta-learning and the mixture of LoRA modules (Laakkonen et al., 2025; Pan et al., 2025). These methods typically introduce trainable transformations into selected projection layers while keeping most of the pre-trained encoder frozen.

Different from existing PEFT-based SDD methods that mainly rely on LoRA, our proposed MoLN performs efficient adaptation through LN tuning. By combining multiple LN experts with dynamic routing, MoLN improves generalization while maintaining a small number of trainable parameters.

Table 1: Comparison of LayerNorm adaptation methods.

| Method | Affine adaptation | Adaptation properties | | | | | |
|---|---|---|---|---|---|---|---|
| | | Expert bank | Feature-conditioned | No inference updates | Soft composition | Frame variation | Applied to SDD |
| LN tuning (Wang et al., 2022) | $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ | × | × | ✓ | × | × | × |
| Dynamic LN (Kim et al., 2017) | $g_\Theta(\bar{\mathbf{h}}(x))$ | × | ✓ | ✓ | × | × | × |
| MoETTA (Fan et al., 2026) | $\mathbf{v}_{\arg\max_k p_k(x)}$ | ✓ | ✓ | × | × | × | × |
| **MoLN (ours)** | $\sum_{k=1}^{K} p_{t,k}(x)\mathbf{v}_k$ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

### 2.2 LayerNorm Tuning

LayerNorm (LN) tuning adapts pre-trained models by updating the learnable scale and shift parameters in LN while keeping most of the backbone frozen, providing a lightweight adaptation strategy. As summarized in Table 1, LN tuning (Wang et al., 2022) learns a single set of affine parameters shared by all inputs, without conditioning the adaptation on input features. Dynamic LN (Kim et al., 2017) instead predicts the affine coefficients from the input representation, enabling the LN parameter to vary across samples, but does not maintain a reusable bank of parameters. MoETTA (Fan et al., 2026) introduces multiple LN experts and selects one expert according to the input features, extending LN adaptation with an expert bank and dynamic routing. However, its routing operates at the sample level through hard selection and further relies on parameter updates during inference. These methods demonstrate different ways of increasing the flexibility of LN adaptation beyond a single one.

Different from existing LN adaptation methods, MoLN maintains multiple reusable LN parameters for SDD and softly combines them at each frame based on the input representation. This enables complementary parameters to contribute jointly and allows the adaptation to vary over time without inference updates. MoLN thus provides flexible adaptation to heterogeneous speech conditions.

## 3 Problem Formulation and Analysis

### 3.1 Problem Formulation

Let $\mathcal{D}_s = \{(x_i, y_i)\}_{i=1}^{N}$ denote the source SDD dataset, where $x_i$ is a speech utterance and $y_i \in \{0, 1\}$ is its spoof label. Given a pre-trained speech encoder $E_{\Phi_0}$ and a detector $C_\psi$, the model is trained as

$$\mathcal{L}_{\text{SDD}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log p_i + (1 - y_i)\log(1 - p_i)\right], \tag{1}$$

where $p_i = \sigma(C_\psi(E_{\Phi_0}(x_i)))$ , is the probabilistic prediction of the model. Our goal is to adapt the pre-trained encoder on the source dataset while generalizing to unseen speech conditions.

A common parameter-efficient adaptation strategy for SDD is LoRA (Hu et al., 2022), which adapts the attention projection space while keeping the pre-trained backbone frozen. For a projection matrix $W_{\ell,m}^0$, LoRA parameterizes the update as

$$W_{\ell,m} = W_{\ell,m}^0 + \frac{\alpha}{r} B_{\ell,m} A_{\ell,m}, \tag{2}$$

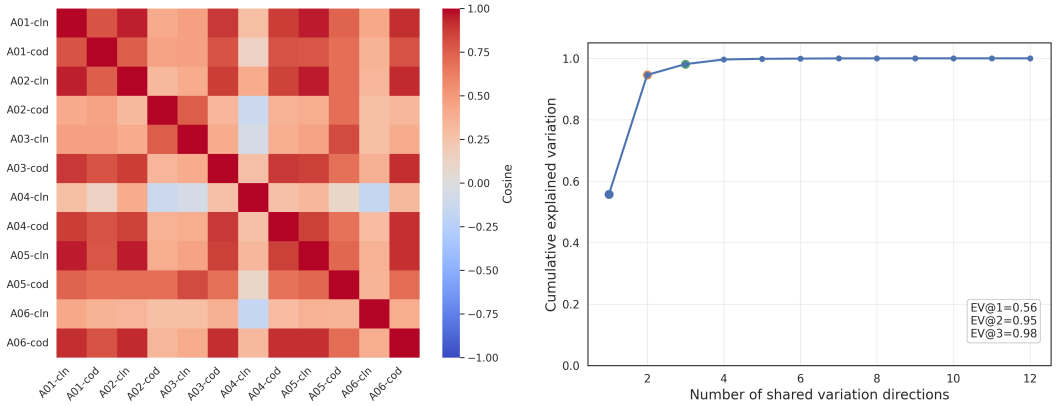
where $r$ is the rank and $A_{\ell,m}$ and $B_{\ell,m}$ are trainable low-rank matrices.

### 3.2 Analysis of LayerNorm Tuning

Different from LoRA, LayerNorm (LN) tuning updates only the affine scale and shift parameters while keeping the remaining encoder frozen. For the $\ell$-th LN module, we write

$$\boldsymbol{\gamma}_\ell = \boldsymbol{\gamma}_\ell^0 + \Delta\boldsymbol{\gamma}_\ell, \qquad \boldsymbol{\beta}_\ell = \boldsymbol{\beta}_\ell^0 + \Delta\boldsymbol{\beta}_\ell, \tag{3}$$

where $\boldsymbol{\gamma}_\ell^0$ and $\boldsymbol{\beta}_\ell^0$ are the pre-trained parameters, and $\Delta\boldsymbol{\gamma}_\ell$ and $\Delta\boldsymbol{\beta}_\ell$ are optimized for SDD. Thus, it performs channel-wise feature recalibration without modifying the learned projection matrices.




(a) Condition-gradient cosine similarity.

(b) Shared gradient variation.

Figure 1: Gradient diagnostic at a trained LN tuning checkpoint. Conditions combine six source-validation attack groups with clean speech (`cln`) or an acoustic condition shift (`cod`).

We examine whether this constrained adaptation provides a suitable interface for generalizable SDD by comparing LN tuning with LoRA across different ranks. In addition to ID and OOD EER, we evaluate representation similarity using centered kernel alignment (CKA) and relative feature basis preservation (rFBP), whose definitions are provided in Appendix A. As shown in Table 2, LN tuning uses only 0.55M trainable parameters, fewer than even rank-1 LoRA, while achieving the highest CKA (0.1406) and an rFBP close to the pre-trained self-reference (1.0469). Across the LoRA ranks, both CKA and rFBP are consistently lower, indicating larger changes to the pre-trained representation structure. Meanwhile, LN tuning achieves an OOD EER of 3.75%, outperforming several LoRA configurations but remaining behind rank-4 LoRA (2.82%).

Table 2: Comparison of LN tuning and LoRA. Parameters are in millions (M) and EER in percent.

| Method | Trainable params. (M) | CKA ↑ | rFBP ↑ | ID EER ↓ | OOD EER ↓ |
|---|---|---|---|---|---|
| LN tuning | 0.55 | 0.1406 | 1.0469 | 0.038 | 3.75 |
| LoRA ($r = 1$) | 0.64 | 0.0594 | 0.8633 | 0.079 | 5.74 |
| LoRA ($r = 2$) | 0.84 | 0.0503 | 0.8637 | 0.000 | 4.23 |
| LoRA ($r = 4$) | 1.23 | 0.0894 | 0.8576 | 0.038 | 2.82 |
| LoRA ($r = 8$) | 2.02 | 0.0866 | 0.9350 | 0.033 | 5.02 |
| LoRA ($r = 16$) | 3.59 | 0.0863 | 0.9230 | 0.035 | 4.52 |

These results indicate that LN tuning offers a favorable trade-off between parameter efficiency, representation preservation, and generalization. However, its remaining OOD gap suggests that a single shared LN response may be too restrictive for the heterogeneous conditions in SDD.

### 3.3 Analysis of Shared LayerNorm Tuning

We further examine whether a single shared LN response is sufficient for heterogeneous speech conditions. For the $\ell$-th LN module, the shared parameter is written as $\mathbf{v}_\ell = [\Delta\boldsymbol{\gamma}_\ell; \Delta\boldsymbol{\beta}_\ell] \in \mathbb{R}^{2D}$, which is applied to all inputs after training. If different speech conditions favor different feature recalibrations, their preferred updates in the LN adaptation space should also differ.

To examine whether different conditions favor different LN parameters, we construct 12 controlled conditions from six ASVspoof 2019 LA source validation attack groups, each using original audio or audio with a bandwidth restriction. At a fixed LN tuning checkpoint, let $\boldsymbol{\theta}_{\mathrm{LN}}$ denote the trainable LN affine parameters and $\mathcal{L}_c$ denote the detection loss for condition $c$. We compute $\mathbf{g}_c = \nabla_{\boldsymbol{\theta}_{\mathrm{LN}}} \mathcal{L}_c$, where $\mathbf{g}_c$ characterizes the local adaptation preference of condition $c$. These gradients are computed without model updates or target evaluation data, and attack IDs are used only for diagnostic grouping. Details are provided in Appendix B.

As shown in Figure 1a, the mean gradient cosine similarity across conditions is lower than within minibatches of the same condition, indicating distinct adaptation preferences across conditions. Meanwhile, in Figure 1b, the first, second, and third leading directions explain 55.7%, 94.6%, and 98.1% of the centered gradient variation, respectively. The variation is therefore substantial but concentrated in a small number of dominant directions.

**Summary of Analysis.** These results suggest that LN tuning provides a compact adaptation space, but a single shared response is insufficient for heterogeneous speech conditions. The variation across conditions indicates that a small set of reusable responses can capture diverse adaptation requirements. This motivates extending LN tuning with multiple responses for more flexible adaptation.

## 4 METHODOLOGY

Motivated by the analysis results, we propose a novel SDD method called **Mixture of Layer Normalization Experts (MoLN)**. The MoLN architecture extends conventional LN tuning with multiple affine experts. To better exploit multiple experts, the **Training Objective** further encourages diverse expert responses and balanced expert usage.

### 4.1 MIXTURE OF LAYER NORMALIZATION EXPERTS

As illustrated in Figure 2, MoLN replaces the single affine response in conventional LN tuning with a bank of $K$ learnable affine experts. The encoder remains frozen, and each expert learns a residual scale and shift parameter update that the model adaptively combines based on the input.

For the $\ell$-th LN module, let $\widehat{\mathbf{h}}_{\ell,t}(x) \in \mathbb{R}^D$ denote the normalized representation of frame $t$ in utterance $x$. The frozen LN output is

$$\mathbf{y}^0_{\ell,t}(x) = \widehat{\mathbf{h}}_{\ell,t}(x) \odot \boldsymbol{\gamma}^0_\ell + \boldsymbol{\beta}^0_\ell, \tag{4}$$

where $\boldsymbol{\gamma}^0_\ell$ and $\boldsymbol{\beta}^0_\ell$ are the pre-trained affine parameters. Each LN module contains $K$ residual affine experts, where expert $k$ is parameterized by $\boldsymbol{\gamma}_{\ell,k}, \boldsymbol{\beta}_{\ell,k} \in \mathbb{R}^D$. A linear router $\mathbf{W}_{r,\ell} \in \mathbb{R}^{K\times D}$ maps the normalized representation to $K$ routing weights as

$$p_{\ell,t,k}(x) = \mathrm{softmax}\left(\mathbf{W}_{r,\ell}\widehat{\mathbf{h}}_{\ell,t}(x)\right)_k. \tag{5}$$

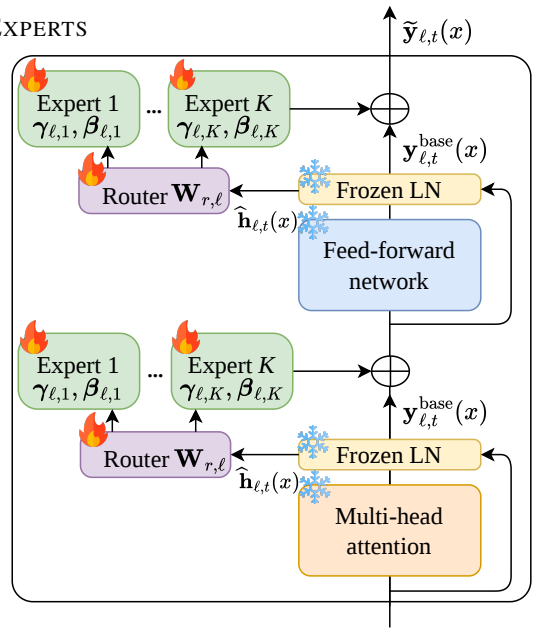



Figure 2: Illustration of Mixture of Layer Normalization Experts. Snowflakes and flames denote frozen and trainable components, respectively.

The residual scale and shift for each frame are obtained by softly combining the expert responses

$$\Delta\boldsymbol{\gamma}_{\ell,t}(x) = \sum_{k=1}^{K} p_{\ell,t,k}(x)\boldsymbol{\gamma}_{\ell,k}, \qquad \Delta\boldsymbol{\beta}_{\ell,t}(x) = \sum_{k=1}^{K} p_{\ell,t,k}(x)\boldsymbol{\beta}_{\ell,k}. \tag{6}$$

These mixed parameters produce the residual MoLN output

$$\mathbf{o}^{\text{MoLN}}_{\ell,t}(x) = \widehat{\mathbf{h}}_{\ell,t}(x) \odot \Delta\boldsymbol{\gamma}_{\ell,t}(x) + \Delta\boldsymbol{\beta}_{\ell,t}(x). \tag{7}$$

The residual output is added to the frozen LN output by

$$\begin{aligned}\mathbf{y}_{\ell,t}(x) &= \mathbf{y}^0_{\ell,t}(x) + \mathbf{o}^{\text{MoLN}}_{\ell,t}(x) \\ &= \widehat{\mathbf{h}}_{\ell,t}(x) \odot \left(\boldsymbol{\gamma}^0_\ell + \Delta\boldsymbol{\gamma}_{\ell,t}(x)\right) + \boldsymbol{\beta}^0_\ell + \Delta\boldsymbol{\beta}_{\ell,t}(x).\end{aligned} \tag{8}$$

Therefore, each frame can adaptively combine the shared expert bank according to its representation, allowing the affine response to vary across different speech conditions without assigning predefined conditions to individual experts. This preserves the compact adaptation space of LN tuning while providing greater flexibility than a single shared response. Only the affine experts, routers, and detector are optimized, while the pretrained encoder remains frozen. All learned parameters are fixed during inference, requiring no additional updates or condition labels.

### 4.2 TRAINING OBJECTIVE

The effectiveness of multiple affine experts depends on both learning complementary responses and avoiding excessive preference for a small subset of experts. We therefore augment the SDD objective with two regularization terms for expert diversity and balanced expert usage.

**Expert diversity.** Similar affine experts provide redundant modulation and reduce the benefit of the expert bank. We therefore encourage diversity among the residual scaling vectors. For the $\ell$-th MoLN module, let $\mathbf{\Gamma}_\ell \in \mathbb{R}^{K\times D}$ stack $\{\boldsymbol{\gamma}_{\ell,k}\}_{k=1}^{K}$ and define $\mathbf{P}_\ell = \mathbf{\Gamma}_\ell \mathbf{\Gamma}_\ell^\top$. The diversity loss is

$$\mathcal{L}_{orth}^{(\ell)} = \sum_{j=1}^{K}\sum_{k\neq j} \frac{\langle \boldsymbol{\gamma}_{\ell,j}, \boldsymbol{\gamma}_{\ell,k}\rangle^2}{\|\boldsymbol{\gamma}_{\ell,k}\|_2^2 + \epsilon} = \sum_{j=1}^{K}\sum_{k\neq j} \frac{P_{\ell,jk}^2}{P_{\ell,kk} + \epsilon}. \tag{9}$$

For nonzero responses and negligible $\epsilon$, each term satisfies

$$\frac{\langle \boldsymbol{\gamma}_{\ell,j}, \boldsymbol{\gamma}_{\ell,k}\rangle^2}{\|\boldsymbol{\gamma}_{\ell,k}\|_2^2} = \|\boldsymbol{\gamma}_{\ell,j}\|_2^2 \cos^2\left(\boldsymbol{\gamma}_{\ell,j}, \boldsymbol{\gamma}_{\ell,k}\right). \tag{10}$$

Thus, the loss penalizes both response alignment and magnitude rather than imposing a purely angular constraint. We apply this regularization only to $\boldsymbol{\gamma}$, which controls channel scaling, while leaving $\boldsymbol{\beta}$ unconstrained to allow shared offset corrections. Appendix C provides the derivation.

**Load balancing.** Diverse experts are useful only when the router makes effective use of the expert bank. For the $\ell$-th MoLN module, let $p_{b,t,k} = p_{\ell,t,k}(x_b)$ denote the routing probability of expert $k$, and let $m_{b,t} \in \{0,1\}$ indicate whether frame $t$ of utterance $b$ is valid. The mean routing probability is

$$\bar{p}_{\ell,k} = \frac{1}{N_{\text{valid}}} \sum_{b,t} m_{b,t} p_{b,t,k}, \qquad N_{\text{valid}} = \sum_{b,t} m_{b,t}. \tag{11}$$

Following MoE load balancing objectives (Fedus et al., 2022), we define

$$\mathcal{L}_{lb}^{(\ell)} = K \sum_{k=1}^{K} \bar{p}_{\ell,k}^2. \tag{12}$$

This loss reaches its minimum value of 1 under uniform mean expert usage and increases as the routing distribution becomes more imbalanced.

**Final objective.** We jointly optimize detection, expert diversity, and load balancing:

$$\mathcal{L}_{MoLN} = \mathcal{L}_{SDD} + \lambda_{orth} \sum_{\ell=1}^{L} \mathcal{L}_{orth}^{(\ell)} + \frac{\lambda_{lb}}{L} \sum_{\ell=1}^{L} \mathcal{L}_{lb}^{(\ell)}, \tag{13}$$

where $L$ is the number of MoLN modules, and $\lambda_{orth}$ and $\lambda_{lb}$ weight expert diversity and load balancing, respectively. We sum the diversity loss over all MoLN modules and average the load-balancing loss, keeping the scale of the load-balancing term independent of the number of modules.

## 5 EXPERIMENTS

### 5.1 EXPERIMENT SETUP

**Training protocol.** We train all models on ASVspoof 2019 LA (Nautsch et al., 2021) using only binary spoof labels. RawBoost-4 (Tak et al., 2022a) is applied during training to increase acoustic diversity. For MoLN, the default expert count $K$ and the regularization scale $\lambda_{\text{orth}} = 0.1$ and $\lambda_{\text{lb}} = 0.1$ are selected on the ASVspoof 2019 LA development set and fixed before evaluation.

Table 3: Main comparison on XLSR-AASIST over two random seeds. EER (%) is reported for each evaluation set. Avg. is the unweighted mean over the four sets. The best and second-best results among our runs are shown in bold and underlined, respectively. † denotes the five-seed mean reported by Laakkonen et al. (2026), with the full mean ± standard deviation given in Appendix E.3. ‡ denotes the reported LoRA-MoE result with three experts and rank 4 by Laakkonen et al. (2025). Our reproduced LoRA-MoE follows the implementation of Laakkonen et al. (2025), except for the learned noisy gating mechanism.

| | | | Cross-dataset EER (%)↓ | | | | |
|---|---|---|---|---|---|---|---|
| Method | Configuration | Trainable params. (M) | 21DF | ITW | FoR | ASV5 | Avg. |
| ***Full and shared adaptation baselines*** | | | | | | | |
| Full fine-tuning | All parameters | 317.84 | $3.25 \pm 0.85$ | $9.73 \pm 2.98$ | $11.17 \pm 4.62$ | $19.81 \pm 2.43$ | $10.99 \pm 2.72$ |
| Backend-only tuning | Detector backend | 0.45 | $6.62 \pm 0.21$ | $20.49 \pm 1.15$ | $13.34 \pm 0.18$ | $\mathbf{15.54 \pm 0.00}$ | $14.00 \pm 0.28$ |
| BitFit | Encoder biases + backend | 0.73 | $3.61 \pm 0.40$ | $12.14 \pm 0.53$ | $16.83 \pm 6.80$ | $18.65 \pm 0.04$ | $12.81 \pm 1.48$ |
| LN tuning | $\gamma, \beta$ + backend | 0.55 | $4.06 \pm 0.27$ | $8.87 \pm 0.51$ | $16.37 \pm 0.23$ | $20.30 \pm 0.88$ | $12.40 \pm 0.08$ |
| Scale-only LN adapter | Residual $\gamma$ + backend | 0.50 | $5.41 \pm 0.64$ | $13.21 \pm 1.17$ | $16.05 \pm 0.09$ | $15.92 \pm 1.56$ | $12.65 \pm 0.86$ |
| Houlsby adapter | 2/block; $d_a = 4$ + backend | 0.89 | $2.67 \pm 0.24$ | $\underline{8.05 \pm 1.41}$ | $5.29 \pm 0.76$ | $20.30 \pm 3.46$ | $9.08 \pm 0.76$ |
| ***Projection-space PEFT*** | | | | | | | |
| LoRA† | Q,K,V,out; $r = 16$ | 3.59 | 4.35 | 11.05 | – | 20.05 | – |
| LoRA (reproduced) | Q,K,V,out; $r = 16$ | 3.59 | $3.09 \pm 0.54$ | $9.26 \pm 0.06$ | $\underline{5.09 \pm 2.82}$ | $\underline{15.75 \pm 1.06}$ | $\underline{8.29 \pm 1.09}$ |
| LoRA-MoE‡ | Q,K,V,out; $r = 4$; $K = 3$ | 3.40 | 6.78 | 10.60 | – | 22.49 | – |
| LoRA-MoE (reproduced) | Q,K,V,out; $r = 4$; $K = 3$ | 3.10 | $2.82 \pm 0.35$ | $9.07 \pm 0.13$ | $6.12 \pm 0.59$ | $16.47 \pm 2.52$ | $8.62 \pm 0.66$ |
| ***MoLN-based adaptation (ours)*** | | | | | | | |
| **LoRA+MoLN** | Q,K,V,out; $r = 4$; $K = 3$ | 1.69 | $\underline{2.50 \pm 0.29}$ | $8.23 \pm 0.94$ | $6.29 \pm 1.60$ | $19.63 \pm 0.45$ | $9.16 \pm 0.67$ |
| **MoLN** | $K = 3$; soft routing | 0.90 | $\mathbf{2.38 \pm 0.08}$ | $\mathbf{6.69 \pm 0.03}$ | $\mathbf{4.30 \pm 0.90}$ | $16.84 \pm 0.49$ | $\mathbf{7.55 \pm 0.32}$ |

**Baselines.** We compare MoLN with full fine-tuning, backend-only tuning, LN tuning, scale-only LN adaptation, BitFit, Houlsby adapters, LoRA, and a dense soft LoRA-MoE variant. These baselines cover full model adaptation, lightweight affine tuning, bias tuning, bottleneck adapters, and projection based parameter efficient adaptation. All methods train the corresponding detector backend. Method specific optimization settings and implementation details, including deviations from published baselines, are provided in Appendix D.

**Evaluation benchmarks.** We evaluate cross-dataset generalization on ASVspoof 2021 DF (**21DF**) (Liu et al., 2023), ASVspoof 5 (**ASV5**) (Wang et al., 2024), In-the-Wild (**ITW**) (Müller et al., 2022), and Fake-or-Real (**FoR**) (Reimao & Tzerpos, 2019). For WavLM-AASIST and XLSR-Conformer-TCM, we additionally evaluate on ADD 2023 R1 and R2 (**ADDR1** and **ADDR2**) (Yi et al., 2023). These benchmarks cover different spoofing attacks, recording conditions, and data collection settings, providing a broad evaluation of generalization to unseen speech conditions.

**Evaluation protocol.** We report equal error rate (EER, %), where lower values indicate better performance. Main results are averaged over two random seeds. For each table, *Avg.* denotes the unweighted mean EER across the evaluation benchmarks included in that table.

### 5.2 Overall Performance

**Comparison with existing methods.** MoLN achieves the lowest average EER in Table 3, reaching 7.55% with only 0.90M trainable parameters. Among the non-routed PEFT baselines, the parameter-matched Houlsby adapter performs strongest, with an average EER of 9.08% using 0.89M parameters. MoLN reduces the average EER by 16.9% relative to this closely matched baseline, with particularly clear gains on ITW and FoR. This consistent advantage over a comparably sized adapter suggests that dynamically conditioning the LN response on a frame level provides benefits beyond simply increasing adaptation capacity. The two-seed mean of MoLN also improves over full fine-tuning on every dataset despite updating less than 1M parameters. The improvement is less pronounced on ASV5, where backend-only tuning and reproduced LoRA achieve lower EERs. One possible explanation is that the self-supervised knowledge retained in the pre-trained frontend is already particularly beneficial for the ASV5 distribution, making more conservative adaptation preferable on this domain. Nevertheless, MoLN remains substantially more consistent across the other cross-dataset conditions, resulting in the best overall average performance.

**Compared with LoRA.** The reproduced rank-16 LoRA remains competitive on ASV5 but is weaker than MoLN on 21DF, ITW, and FoR, while the reproduced LoRA-MoE does not close this gap. LoRA+MoLN performs strongly on 21DF, suggesting that MoLN can complement projection-space adaptation under certain conditions, but its performance is less consistent across datasets. These

Table 4: Backbone generality. Avg. is the unweighted mean EER (%) over the six evaluation sets. Bold and underlined values indicate the best and second-best results within each model block. Complete expert-count sweeps are in appendix E.4.

| Method | Configuration | Trainable params. (M) | Cross-dataset EER (%)↓ | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | 21DF | ITW | FoR | ASV5 | ADDR1 | ADDR2 | Avg. |
| ***WavLM-AASIST*** | | | | | | | | | |
| Full fine-tuning | All parameters | 315.90 | <u>6.07</u> | 15.88 | 23.98 | 18.80 | 39.90 | 31.79 | 22.74 |
| LN tuning | $\gamma, \beta$ | 0.56 | **5.66** | 13.55 | <u>16.03</u> | <u>14.36</u> | 34.64 | 31.43 | 19.28 |
| MoLN | $K = 2$; soft routing | 0.75 | 6.60 | **12.80** | **11.65** | 16.43 | **23.66** | **22.21** | **15.56** |
| MoLN | $K = 3$; soft routing | 0.90 | 6.58 | <u>13.18</u> | 16.53 | **13.66** | <u>25.81</u> | <u>24.30</u> | <u>16.68</u> |
| ***XLSR-Conformer-TCM*** | | | | | | | | | |
| Full fine-tuning | All parameters | 319.00 | **2.25** | <u>7.19</u> | 7.94 | <u>16.74</u> | 29.89 | 27.50 | 15.25 |
| LN tuning | $\gamma, \beta$ | 2.38 | 9.41 | 31.45 | 22.18 | 25.05 | <u>25.93</u> | <u>21.29</u> | 22.55 |
| MoLN | $K = 2$; soft routing | 2.69 | <u>2.81</u> | 8.29 | <u>7.08</u> | **15.38** | 28.51 | 28.41 | <u>15.08</u> |
| MoLN | $K = 3$; soft routing | 2.84 | 2.90 | **7.08** | **6.24** | 18.09 | **25.22** | **20.92** | **13.41** |

Table 5: MoLN ablations on XLSR-AASIST. Avg. is the unweighted mean EER (%) over the four evaluation sets. Bold and underlined values indicate the best and second-best results, respectively.

| Method | Configuration | Trainable params. (M) | Cross-dataset EER (%)↓ | | | | |
|---|---|---|---|---|---|---|---|
| | | | 21DF | ITW | FoR | ASV5 | Avg. |
| ***Main configuration*** | | | | | | | |
| **MoLN** | $K = 3$; soft routing; $\lambda_{\text{orth}} = \lambda_{\text{lb}} = 0.1$ | 0.90 | <u>2.44</u> | **6.71** | **3.67** | **16.49** | **7.33** |
| ***Expert count and routing*** | | | | | | | |
| MoLN | $K = 1$; soft routing | 0.60 | 3.68 | 10.32 | 14.22 | 23.82 | 13.01 |
| MoLN | $K = 2$; soft routing | 0.75 | 2.68 | 8.65 | 6.67 | 18.78 | <u>9.20</u> |
| MoLN | $K = 3$; top-1 routing | 0.90 | 4.53 | 13.13 | 16.38 | 19.99 | 13.51 |
| MoLN | $K = 5$; soft routing | 1.06 | 3.01 | 8.70 | <u>5.57</u> | 22.56 | 9.96 |
| MoLN | $K = 10$; soft routing | 1.97 | 3.00 | <u>8.37</u> | 6.15 | 20.14 | 9.42 |
| ***Auxiliary losses*** | | | | | | | |
| MoLN | $K = 3$; orth. on $\gamma, \beta$ | 0.90 | 3.36 | 13.13 | 9.99 | 21.32 | 11.95 |
| MoLN | $K = 3$; no load balancing | 0.90 | 3.40 | 11.49 | 6.58 | <u>18.31</u> | 9.95 |
| MoLN | $K = 3$; no orth. loss | 0.90 | 3.63 | 8.62 | 21.34 | 18.70 | 13.07 |
| MoLN | $K = 3$; no auxiliary losses | 0.90 | **2.33** | 9.88 | 5.87 | 21.27 | 9.84 |
| ***Auxiliary-loss weights*** | | | | | | | |
| MoLN | $K = 3$; $\lambda_{\text{orth}} = \lambda_{\text{lb}} = 0.01$ | 0.90 | 3.03 | 8.38 | 8.78 | 21.83 | 10.51 |
| MoLN | $K = 3$; $\lambda_{\text{orth}} = \lambda_{\text{lb}} = 0.2$ | 0.90 | 3.61 | 9.28 | 10.21 | 18.87 | 10.49 |

results suggest that introducing conditional capacity alone is insufficient. Where the adaptation is applied and how the conditional affine updates are parameterized and composed are also important for cross-dataset generalization.

**Generality across architectures.** Table 4 combines XLS-R or WavLM with AASIST or Conformer-TCM (Babu et al., 2022; Chen et al., 2022; Jung et al., 2022; Truong et al., 2024). The average EER is 15.56% on WavLM-AASIST and 13.41% on XLSR-Conformer-TCM, with the source-selected default $K = 2$ and $K = 3$, respectively, compared with 19.28% and 22.55% for LN tuning. The sensitivity sweep further shows that WavLM-AASIST reaches 15.56% with $K = 2$, while XLSR-Conformer-TCM remains stronger with $K = 3$. Expert count therefore affects the two configurations differently, but no target-specific value is selected for the main claim.

### 5.3 Further Study

**Design ablations.** The $K = 1$ control in table 5 keeps the router code path and insertion sites. Its softmax has one logit, so $p_{\ell,t,1}(x) = 1$ and the affine expert is shared across inputs and frames. Increasing $K$ from 1 to 3 lowers average EER from 13.01% to 7.33%. Top-1 routing raises it to 13.51%, and regularizing both $\gamma$ and $\beta$ raises it to 11.95%. Removing load balancing, orthogonality loss, or both auxiliary losses gives 9.95%, 13.07%, and 9.84%, respectively. The loss-free model performs slightly better on 21DF but worse on the other targets. For the auxiliary-loss weights, the default setting $\lambda_{\text{orth}} = \lambda_{\text{lb}} = 0.1$ performs best among the tested values. Both weaker (0.01) and stronger (0.2) regularization increase the average EER to 10.51% and 10.49%, respectively, indicating that the effect is not monotonic with regularization strength. All ablations are single runs.

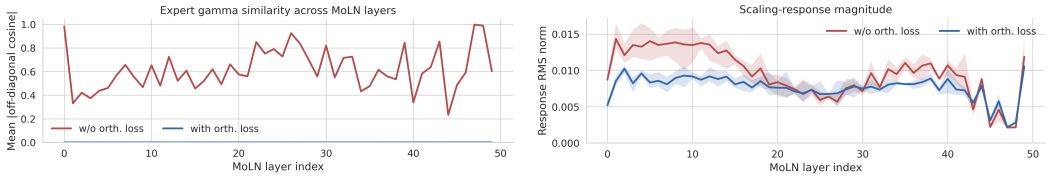



Figure 3: Expert diversity analysis of MoLN on XLSR-AASIST, with and without the orthogonality loss. Left: mean absolute off-diagonal cosine similarity between the residual scaling vectors at each of the 50 MoLN layers. Right: RMS norms of the residual scaling vectors; the solid line denotes the mean over the three experts, and the shaded region spans their minimum and maximum. Together, these results show that the orthogonality loss learns distinct expert directions without norm collapse.

**Mechanism analysis.** Without $\mathcal{L}_{orth}$, the mean absolute off-diagonal cosine similarity between the scaling parameters is 0.6172 across 50 layers. The loss reduces it to $1.07\times10^{-4}$ (Figure 3, left). Across the same 150 expert scaling vectors, the mean RMS norm decreases only from 0.00957 to 0.00767 and remains within the same order of magnitude. Figure 3 (right) shows that the expert norms remain nonzero throughout the network, supporting directional separation without norm collapse. Thus, $\mathcal{L}_{orth}$ learns distinct, non-collapsed parameter update directions. Appendix G.2 provides additional details. Figure 4 illustrates this shift in relative expert usage across the two input groups. Together, these results support the two main components of MoLN, a bank of distinct affine experts and representation-conditioned soft routing over that bank.

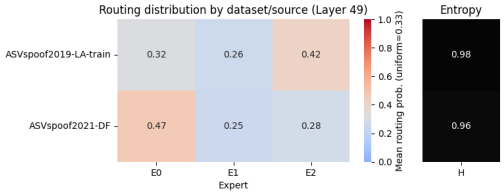



Figure 4: Average routing distributions at the last MoLN layer on ASVspoof 2019 LA training data and ASVspoof 2021 DF. The input groups exhibit different expert mixtures, while multiple experts remain active at the aggregate level.

**Representation analysis.** In the shared PCA projection shown in Figure 5, the adaptation methods induce distinct changes to the pre-trained representation distribution. Full fine-tuning causes both a substantial centroid shift and a strong contraction of the projected representations, whereas LN tuning and LoRA-r4 produce different shifts and reorientations while retaining broader variation. MoLN also changes the centroid and covariance structure, but preserves substantial overlap and spread relative to the pre-trained representation. This indicates that MoLN does not simply keep the pre-trained features unchanged. Instead, it introduces structured adaptation while retaining much of their representation structure. MoLN provides a balance between preserving transferable representation structure and introducing task-specific adaptation.

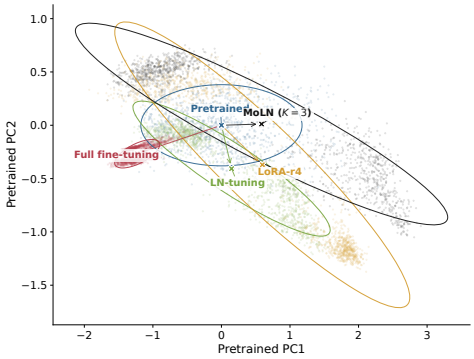



Figure 5: Final-layer XLS-R representations in a shared PCA projection fitted to the pre-trained encoder.

## 6 Conclusion

In this work, we investigate LayerNorm tuning as a compact adaptation interface for generalizable SDD. Our analysis shows that while LN tuning provides favorable parameter efficiency and representation preservation, a single set of shared affine parameters remains limited under heterogeneous speech conditions. We therefore propose MoLN, which extends LN tuning with multiple lightweight experts and frame-level soft routing. To better exploit the expert structure, we further introduce expert diversity and load balancing objectives to encourage complementary responses and balanced expert usage. Extensive cross-dataset experiments across different encoder and detector configurations demonstrate strong generalization with few trainable parameters, without requiring explicit domain labels, test-time updates, or the access of target data.

### AI USE STATEMENT

In this work, we did not use generative AI tools to generate synthetic data sets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean or reformat datasets, support qualitative or thematic data analysis, or interpret results. The remaining tasks requiring disclosure were not applicable to this work.

Additionally, we used generative AI tools to draft and revise parts of the manuscript, improve readability and organization, refine the presentation and wording of mathematical formulations, refine figure and table descriptions, and check reference formatting. We reviewed all AI-assisted work. In particular, all AI-assisted text and edits to the presentation of mathematical expressions were reviewed by the authors to ensure that they did not alter the underlying mathematical claims or scientific meaning. Any references suggested or reformatted with the assistance of generative AI were checked against the original publications, and all reported results were verified against the authors' experimental records. Generative AI was not used to make methodological decisions or independently draw scientific conclusions.

The authors made all final methodological and editorial decisions and take responsibility for the final content of this work, including any text or other artifacts produced with the aid of generative AI.

## REFERENCES

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. XLS-R: Self-supervised cross-lingual speech representation learning at scale. In *Proc. Interspeech*, pp. 2278–2282, 2022. doi:10.21437/Interspeech.2022-143.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 12449–12460, 2020.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language models. In *Proc. 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, 2022. doi:10.18653/v1/2022.acl-short.1.

Jiawei Chen, Dingkang Yang, Yue Jiang, Mingcheng Li, Jinjie Wei, Xiaolu Hou, and Lihua Zhang. Efficiency in focus: LayerNorm as a catalyst for fine-tuning medical visual language models. In *Proc. ACM International Conference on Multimedia*, pp. 3122–3130, 2024.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6): 1505–1518, 2022. doi:10.1109/JSTSP.2022.3188113.

Yassine El Kheir, Younes Samih, Suraj Maharjan, Tim Polzehl, and Sebastian Möller. Comprehensive layer-wise analysis of SSL models for audio deepfake detection. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 4070–4082, 2025. doi:10.18653/v1/2025.findings-naacl.227.

Xiao Fan, Jingyan Jiang, Zhaoru Chen, Fanding Huang, Xiao Chen, Qinting Jiang, Bowen Zhang, Xing Tang, and Zhi Wang. MoETTA: test-time adaptation under mixed distribution shifts with MoE-LayerNorm. In *Proc. AAAI Conference on Artificial Intelligence*, volume 40, pp. 21011–21019, 2026. doi:10.1609/aaai.v40i25.39243.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Yinlin Guo, Haofan Huang, Xi Chen, He Zhao, and Yuehai Wang. Audio deepfake detection with self-supervised WavLM and multi-fusion attentive classifier. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12702–12706, 2024. doi:10.1109/ICASSP48485.2024.10447923.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proc. International Conference on Machine Learning (ICML)*, volume 97, pp. 2790–2799, 2019.

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. doi:10.1109/TASLP.2021.3122291.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proc. International Conference on Learning Representations (ICLR)*, 2022.

Jee-Weon Jung, Hee-Soo Heo, Hemlata Tak, Hye-Jin Shim, Joon Son Chung, Bong-Jin Lee, Ha-Jin Yu, and Nicholas Evans. AASIST: Audio anti-spoofing using integrated spectro-temporal graph attention networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6367–6371, 2022. doi:10.1109/ICASSP43922.2022.9747766.

Taesup Kim, Inchul Song, and Yoshua Bengio. Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition. In *Proc. Interspeech*, pp. 2411–2415, 2017. doi:10.21437/Interspeech.2017-556.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. International Conference on Learning Representations (ICLR)*, 2015.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proc. International Conference on Machine Learning (ICML)*, volume 97, pp. 3519–3529, 2019.

Janne Laakkonen, Ivan Kukanov, and Ville Hautamäki. Mixture of low-rank adapter experts in generalizable audio deepfake detection. In *Proc. Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 2211–2216, 2025.

Janne Laakkonen, Ivan Kukanov, and Ville Hautamäki. Generalizable speech deepfake detection via meta-learned LoRA. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19092–19096, 2026. doi:10.1109/ICASSP55912.2026.11463170.

Xuechen Liu, Xin Wang, Md Sahidullah, Jose Patino, Héctor Delgado, Tomi Kinnunen, Massimiliano Todisco, Junichi Yamagishi, Nicholas Evans, Andreas Nautsch, et al. ASVspoof 2021: Towards spoofed and deepfake speech detection in the wild. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2507–2522, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. International Conference on Learning Representations (ICLR)*, 2019.

Zhiqiang Lv, Shanshan Zhang, Kai Tang, and Pengfei Hu. Fake audio detection based on unsupervised pretraining models. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 9231–9235, 2022.

Nicolas M Müller, Pavel Czempin, Franziska Diekmann, Adam Froghyar, and Konstantin Böttinger. Does audio deepfake detection generalize? In *Proc. Interspeech*, pp. 2783–2787, 2022. doi:10.21437/Interspeech.2022-108.

Andreas Nautsch, Xin Wang, Nicholas Evans, Tomi H. Kinnunen, Ville Vestman, Massimiliano Todisco, Héctor Delgado, Md Sahidullah, Junichi Yamagishi, and Kong Aik Lee. ASVspoof 2019: Spoofing countermeasures for the detection of synthesized, converted and replayed speech. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 3(2):252–265, 2021. doi:10.1109/TBIOM.2021.3059479.

Zihan Pan, Sailor Hardik Bhupendra, and Jinyang Wu. MoLEx: Mixture of LoRA experts in speech self-supervised models for audio deepfake detection. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8, 2025. doi:10.1109/ASRU65441.2025.11434670.

Ricardo Reimao and Vassilios Tzerpos. FoR: A dataset for synthetic speech detection. In *Proc. Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pp. 1–10, 2019.

Hemlata Tak, Madhu Kamble, Jose Patino, Massimiliano Todisco, and Nicholas Evans. RawBoost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6382–6386, 2022a. doi:10.1109/ICASSP43922.2022.9746213.

Hemlata Tak, Massimiliano Todisco, Xin Wang, Jee-weon Jung, Junichi Yamagishi, and Nicholas Evans. Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*, pp. 112–119, 2022b. doi:10.21437/Odyssey.2022-16.

Duc-Tuan Truong, Ruijie Tao, Tuan Nguyen, Hieu-Thi Luong, Kong Aik Lee, and Eng Siong Chng. Temporal-channel modeling in multi-head self-attention for synthetic speech detection. In *Proc. Interspeech*, pp. 537–541, 2024. doi:10.21437/Interspeech.2024-659.

Qi Wang, Yu-Ping Ruan, Yuan Zuo, and Taihao Li. Parameter-efficient tuning on layer normalization for pre-trained language models. *arXiv preprint arXiv:2211.08682*, 2022. URL `https://arxiv.org/abs/2211.08682`.

Xin Wang and Junichi Yamagishi. Investigating self-supervised front ends for speech spoofing countermeasures. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*, pp. 100–106, 2022. doi:10.21437/Odyssey.2022-14.

Xin Wang, Héctor Delgado, Hemlata Tak, Jee-weon Jung, Hye-jin Shim, Massimiliano Todisco, Ivan Kukanov, Xuechen Liu, Md Sahidullah, Tomi Kinnunen, et al. ASVspoof 5: Crowdsourced speech data, deepfakes, and adversarial attacks at scale. In *Proc. The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)*, pp. 1–8, 2024. doi:10.21437/ASVspoof.2024-1.

Jiangyan Yi, Jianhua Tao, Ruibo Fu, Xinrui Yan, Chenglong Wang, Tao Wang, Chu Yuan Zhang, Xiaohui Zhang, Yan Zhao, Yong Ren, et al. ADD 2023: the second audio deepfake detection challenge. In *Proc. IJCAI Workshop on Deepfake Audio Detection and Analysis (DADA)*, pp. 125–130, 2023.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. doi:10.1145/3446776.

Bingchen Zhao, Haoqin Tu, Chen Wei, Jieru Mei, and Cihang Xie. Tuning LayerNorm in attention: Towards efficient multi-modal LLM finetuning. In *Proc. International Conference on Learning Representations (ICLR)*, 2024. URL `https://openreview.net/forum?id=YR3ETaElNK`.

# A REPRESENTATION-SPACE DIAGNOSTICS

We compare adapted XLS-R encoders with their pre-trained initialization using centered linear kernel alignment (CKA) and feature-basis preservation (FBP). CKA measures similarity between utterance-level representation geometries. FBP is a diagnostic defined for this analysis that measures the concentration of frame-level cross-channel correlations on matching channel indices.

## A.1 DATA AND FIXED-CHECKPOINT PROTOCOL

All representation diagnostics use the ASVspoof 2019 LA development partition, without target evaluation data. We retain all 2,548 bona fide utterances and sample 2,548 spoof utterances, stratified by attack identifier: 425 from each of A01–A04 and 424 from each of A05–A06. Sampling uses seed 1111. The resulting $N = 5{,}096$ utterances and their order are fixed across checkpoints.

Audio is loaded as mono at 16 kHz and repeated or truncated to 64,600 samples. RawBoost and stochastic masking are disabled. All encoders are evaluated at fixed checkpoints in evaluation mode, without parameter updates. The analysis excludes the AASIST backend and classifier.

For utterance $x_n$, the final XLS-R Transformer layer produces

$$\mathbf{H}_n^{(m)} = f_{\mathrm{SSL}}^{(m)}(x_n) \in \mathbb{R}^{T_n \times D}, \qquad D = 1024, \tag{14}$$

where $f_{\mathrm{SSL}}^{(m)}$ is the encoder, $m = 0$ denotes the pre-trained checkpoint, and $m = a$ denotes an adapted checkpoint. The fixed input length gives $T_n = 201$ frames for every utterance. Corresponding rows use the same waveform and frame position across encoders.

Table 6: Observation units for representation diagnostics. PCA is used only for visualization and does not enter either metric.

| Analysis | Observation unit | Count | Dimension |
|---|---|---|---|
| Linear CKA | Mean-pooled utterance | 5,096 | 1,024 |
| PCA | Mean-pooled utterance | 5,096 | $1{,}024 \to 2$ |
| FBP | Aligned frame | 1,024,296 | 1,024 |

## A.2 CENTERED LINEAR CKA

We follow centered linear CKA (Kornblith et al., 2019). First, temporal mean pooling gives one column vector per utterance:

$$\mathbf{h}_n^{(m)} = \frac{1}{T_n} \sum_{t=1}^{T_n} \big(\mathbf{H}_{n,t,:}^{(m)}\big)^\top \in \mathbb{R}^D. \tag{15}$$

We stack their transposes as rows of $\mathbf{X}^{(m)} \in \mathbb{R}^{N \times D}$ and center each feature across utterances:

$$\widetilde{\mathbf{X}}^{(m)} = \mathbf{X}^{(m)} - \mathbf{1}_N \boldsymbol{\mu}^{(m)\top}, \qquad \boldsymbol{\mu}^{(m)} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{h}_n^{(m)}, \tag{16}$$

where $\mathbf{1}_N$ is the length-$N$ vector of ones. Let $\mathbf{K}^{(m)} = \widetilde{\mathbf{X}}^{(m)} \widetilde{\mathbf{X}}^{(m)\top}$ be the centered linear Gram matrix. Its normalized Frobenius inner product gives

$$\begin{aligned} \mathrm{CKA}(0, a) &= \frac{\langle \mathbf{K}^{(0)}, \mathbf{K}^{(a)} \rangle_F}{\|\mathbf{K}^{(0)}\|_F \|\mathbf{K}^{(a)}\|_F} \\ &= \frac{\|\widetilde{\mathbf{X}}^{(0)\top} \widetilde{\mathbf{X}}^{(a)}\|_F^2}{\|\widetilde{\mathbf{X}}^{(0)\top} \widetilde{\mathbf{X}}^{(0)}\|_F \|\widetilde{\mathbf{X}}^{(a)\top} \widetilde{\mathbf{X}}^{(a)}\|_F}. \end{aligned} \tag{17}$$

where $\langle \cdot, \cdot \rangle_F$ and $\| \cdot \|_F$ denote the Frobenius inner product and norm. The second expression follows from the trace identity for matrix products and avoids constructing the $N \times N$ Gram matrices.

For nonzero centered representations, CKA lies in $[0, 1]$. It is invariant to orthogonal feature transformations and nonzero isotropic rescaling, so it need not preserve channel identities. We compute it from the full 1,024-dimensional pooled vectors, without dimensionality reduction.

### A.3 PCA VISUALIZATION

We fit PCA once to the pre-trained pooled representations $\mathbf{X}^{(0)}$. Let $\boldsymbol{\mu}_{\mathrm{PCA}} \in \mathbb{R}^D$ be the fitted mean and $\mathbf{W}_{\mathrm{PCA}} \in \mathbb{R}^{D\times 2}$ contain the first two principal directions. All checkpoints use the same projection:

$$\mathbf{P}^{(m)} = \big(\mathbf{X}^{(m)} - \mathbf{1}_N \boldsymbol{\mu}_{\mathrm{PCA}}^\top\big)\mathbf{W}_{\mathrm{PCA}} \in \mathbb{R}^{N\times 2}. \tag{18}$$

The scatter plot displays a fixed class-balanced subset of 1,200 utterances. PCA fitting, projected centroids, covariance ellipses, and CKA use all 5,096 utterances. The common projection makes centroid shifts, spread, and covariance changes directly comparable in the same two-dimensional coordinates; CKA and rFBP quantify complementary properties in the original feature space.

### A.4 FEATURE-BASIS PRESERVATION

**Frame-level channel correspondence.** CKA does not distinguish representations related by an orthogonal change of basis. To examine correspondence at the original channel indices, we define FBP from aligned final-layer frames. Let $\mathbf{f}_q^{(m)} \in \mathbb{R}^D$ denote frame $q$ after concatenating utterances in protocol order. Stacking their transposes gives

$$\mathbf{F}^{(m)} = \begin{bmatrix} \mathbf{f}_1^{(m)\top} \\ \vdots \\ \mathbf{f}_M^{(m)\top} \end{bmatrix} \in \mathbb{R}^{M\times D}, \qquad M = \sum_{n=1}^{N} T_n = 1{,}024{,}296. \tag{19}$$

Here $q$ indexes the same utterance and frame position for both encoders. Each channel $d \in \{1,\dots,D\}$ is standardized across frames:

$$Z_{q,d}^{(m)} = \frac{F_{q,d}^{(m)} - \nu_d^{(m)}}{\sigma_d^{(m)} + \epsilon}, \qquad \epsilon = 10^{-12}, \tag{20}$$

where $\nu_d^{(m)}$ and $\sigma_d^{(m)}$ are its sample mean and sample standard deviation. The cross-channel correlation matrix is

$$\mathbf{C}_{0a} = \frac{1}{M-1}\mathbf{Z}^{(0)\top}\mathbf{Z}^{(a)} \in \mathbb{R}^{D\times D}. \tag{21}$$

Its entry $C_{0a,ij}$ is the sample correlation between pre-trained channel $i$ and adapted channel $j$, up to the stabilizer $\epsilon$.

**Diagonal concentration and self-reference calibration.** We separate squared correlations into matching-index and remaining pairs:

$$E_{\mathrm{diag}} = \sum_{d=1}^{D} C_{0a,dd}^2, \qquad E_{\mathrm{off}} = \sum_{i\neq j} C_{0a,ij}^2. \tag{22}$$

The raw FBP is the fraction of correlation energy on the diagonal:

$$R_{0a} = \mathrm{FBP}_{\mathrm{raw}}(0,a) = \frac{E_{\mathrm{diag}}}{E_{\mathrm{diag}} + E_{\mathrm{off}}} = \frac{\|\operatorname{diag}(\mathbf{C}_{0a})\|_2^2}{\|\mathbf{C}_{0a}\|_F^2}, \tag{23}$$

Here $\operatorname{diag}(\cdot)$ extracts the diagonal as a vector and $\|\cdot\|_2$ is the Euclidean norm. For nonzero $\mathbf{C}_{0a}$, this ratio lies in $[0,1]$. Squaring makes it insensitive to correlation signs.

Even identical representations need not give raw FBP equal to one, because the pre-trained channels can be mutually correlated. We therefore compute $\mathbf{C}_{00} = \mathbf{Z}^{(0)\top}\mathbf{Z}^{(0)}/(M-1)$ on the same frames and define

$$\mathrm{rFBP}(0,a) = \frac{R_{0a}}{R_{00}}, \qquad R_{00} = \frac{\|\operatorname{diag}(\mathbf{C}_{00})\|_2^2}{\|\mathbf{C}_{00}\|_F^2}. \tag{24}$$

For a nondegenerate reference, the self-comparison has rFBP equal to one. A value below or above one indicates lower or higher diagonal concentration relative to that reference. Thus, rFBP is a self-calibrated measure of feature-basis preservation, with one denoting the pre-trained reference level.

Table 7: Fixed-checkpoint representation diagnostics relative to pre-trained XLS-R. CKA uses mean-pooled utterances; FBP uses aligned frames. CKA and rFBP are shown to four decimal places, matching the main-text reporting precision; raw FBP retains six decimal places.

| Encoder | Linear CKA | Raw FBP | Relative FBP |
|---|---|---|---|
| Pre-trained self | 1.0000 | 0.005632 | 1.0000 |
| LN tuning | 0.1406 | 0.005896 | 1.0469 |
| LoRA ($r = 4$) | 0.0894 | 0.004830 | 0.8576 |
| MoLN ($K = 3$) | 0.1148 | 0.005729 | 1.0172 |
| Full fine-tuning | 0.0494 | 0.000930 | 0.1652 |

**Streaming computation.** We avoid storing the full frame matrices by accumulating the channel sums $\mathbf{s}^{(m)}$, squared sums $\mathbf{u}^{(m)}$, and matrix products $\mathbf{Q}_{0a}$ and $\mathbf{Q}_{00}$:

$$\begin{aligned} \mathbf{s}^{(m)} &= \sum_{q=1}^{M} \mathbf{f}_q^{(m)}, & \mathbf{u}^{(m)} &= \sum_{q=1}^{M} \mathbf{f}_q^{(m)} \odot \mathbf{f}_q^{(m)}, \\ \mathbf{Q}_{0a} &= \sum_{q=1}^{M} \mathbf{f}_q^{(0)} \mathbf{f}_q^{(a)\top}, & \mathbf{Q}_{00} &= \sum_{q=1}^{M} \mathbf{f}_q^{(0)} \mathbf{f}_q^{(0)\top}, \end{aligned} \tag{25}$$

where $\odot$ denotes element-wise multiplication. The sample statistics are recovered as

$$\nu_d^{(m)} = \frac{s_d^{(m)}}{M}, \qquad \left(\sigma_d^{(m)}\right)^2 = \frac{u_d^{(m)} - \left(s_d^{(m)}\right)^2 / M}{M - 1}. \tag{26}$$

Writing $\mathbf{V}^{(m)} = \mathrm{diag}(\sigma_1^{(m)} + \epsilon, \ldots, \sigma_D^{(m)} + \epsilon)$, where diag constructs a diagonal matrix from its arguments, gives

$$\mathbf{C}_{0a} = \left(\mathbf{V}^{(0)}\right)^{-1} \frac{\mathbf{Q}_{0a} - \mathbf{s}^{(0)} \mathbf{s}^{(a)\top} / M}{M - 1} \left(\mathbf{V}^{(a)}\right)^{-1}. \tag{27}$$

The self-reference follows by setting $a = 0$. This calculation is algebraically equivalent to explicit centering and standardization. Matrix products use FP32 with tensor-float-32 disabled, and batch-level statistics are accumulated in FP64.

### A.5 Results and interpretation

**LN tuning preserves the pre-trained representation structure more strongly than rank-4 LoRA and full fine-tuning, while MoLN retains near-reference feature-basis preservation with conditional adaptation.** Table 7 reports fixed-checkpoint results. LN tuning achieves CKA of 0.1406 and rFBP of 1.0469, compared with 0.0894 and 0.8576 for rank-4 LoRA. MoLN retains an rFBP of 1.0172 and a CKA of 0.1148, both above rank-4 LoRA and full fine-tuning. Full fine-tuning produces the largest change under both diagnostics, with CKA of 0.0494 and rFBP of 0.1652. These findings support the main-text conclusion that LN-based feature recalibration preserves pre-trained representation structure while enabling task-specific adaptation.

The shared PCA visualization provides a complementary view: full fine-tuning shifts and contracts the projected distribution, whereas MoLN retains substantial overlap and spread relative to the pre-trained encoder while changing its centroid and covariance. Together, the quantitative diagnostics and visualization support MoLN's balance between representation preservation and adaptive feature recalibration.

**Complementary diagnostics.** CKA measures utterance-level geometric similarity and is invariant to orthogonal feature transformations; rFBP measures matching-channel correlation concentration over aligned frames after channel-wise standardization. The two metrics therefore characterize complementary aspects of representation preservation. Near-reference rFBP together with lower CKA captures the key pattern of LN-based adaptation: preserving feature-basis correspondence while adapting representation geometry. These source-development diagnostics support the representation analysis alongside the cross-dataset evaluation; rFBP is a relative concentration measure rather than a percentage of retained information.

# B CONDITION-GRADIENT ANALYSIS OF SHARED LN TUNING

## B.1 CHECKPOINT AND DATA

The diagnostic in Figure 1a and 1b uses a trained XLSR-AASIST LN tuning checkpoint. LN tuning updates one shared set of LN gain and bias parameters. Its affine response is shared across inputs and frames. We collect gradients only for the LN affine parameters and do not take an optimizer step.

All samples come from the ASVspoof 2019 LA development partition. We use the six available spoofing-system identifiers, A01–A06, to form reproducible metadata groups. Each group contains 3,716 spoof utterances, and the shared bona fide pool contains 2,548 utterances. The diagnostic uses no target evaluation data, including ASVspoof 2021 DF.

## B.2 CONTROLLED CONDITIONS

We pair each metadata group with original audio (`cln`) and a deterministic bandwidth restriction (`cod`), giving 12 conditions. For `cod`, we downsample each 16 kHz waveform to 8 kHz and resample it to 16 kHz. We then match its length to the original input. The figure label `cod` denotes this bandwidth-restricted acoustic condition.

For each condition, we draw eight balanced mini-batches of size 16. Each batch contains equal numbers of spoof samples from the current group and bona fide samples from the fixed pool. We apply the same acoustic treatment to both classes. This design limits variation caused by class proportions.

## B.3 GRADIENT AND VARIATION STATISTICS

Let $g_{c,b}$ denote the cross-entropy gradient for batch $b$ in condition $c$. We concatenate the 100 LN affine tensors returned by the model's gradient selector and average the eight batch gradients:

$$g_c = \frac{1}{8} \sum_{b=1}^{8} g_{c,b}. \tag{28}$$

Panel (a) reports $g_i^\top g_j / (\|g_i\| \|g_j\|)$ for every condition pair. We compute the mean between-condition cosine and negative-pair ratio from the off-diagonal entries. The within-condition statistic is the mean pairwise cosine among batch gradients from the same condition. As reported in the main text, the mean between-condition cosine is lower than the within-condition statistic, indicating distinct local adaptation preferences across conditions.

For panel (b), we stack the condition gradients as rows of $G$ and center them across conditions, $\widetilde{G} = G - \bar{g}$. Let $\sigma_k$ denote the singular values of $\widetilde{G}$. The cumulative explained variation is

$$\mathrm{EV}(K) = \frac{\sum_{k=1}^{K} \sigma_k^2}{\sum_j \sigma_j^2}. \tag{29}$$

The first one, two, and three directions cumulatively explain 55.7%, 94.6%, and 98.1% of the centered gradient variation. Thus, condition-dependent adaptation preferences are concentrated in a small number of dominant directions.

## B.4 FINDINGS AND MOTIVATION FOR MOLN

**Heterogeneous speech conditions favor distinct but structured LN updates.** The lower between-condition gradient similarity shows why a single shared affine response limits adaptation flexibility. At the same time, the concentration of 98.1% of the centered variation in three directions motivates a compact bank of reusable affine experts. MoLN implements this design with frame-level soft routing, allowing complementary responses to be combined for heterogeneous and overlapping conditions. The diagnostic characterizes local adaptation preferences at the fixed source-trained checkpoint; the expert-count and routing ablations evaluate the resulting architecture.

# C Derivation motivating the expert-diversity regularizer

For one MoLN module, consider residual expert responses $j$ and $k$ applied to the same normalized frame $\widehat{\mathbf{h}}$:

$$\mathbf{y}_j = \boldsymbol{\gamma}_j \odot \widehat{\mathbf{h}} + \boldsymbol{\beta}_j, \qquad \mathbf{y}_k = \boldsymbol{\gamma}_k \odot \widehat{\mathbf{h}} + \boldsymbol{\beta}_k. \tag{30}$$

LN normalizes each frame across feature dimensions. For the following moment-based motivation, we use the approximations $\mathbb{E}[\widehat{h}_i] \approx 0$ and $\mathbb{E}[\widehat{h}_i^2] \approx 1$ over the data distribution; per-frame normalization does not itself impose these channel-wise moments. Expanding the expected inner product gives

$$\begin{aligned}\mathbb{E}[\mathbf{y}_j^\top \mathbf{y}_k] = \mathbb{E}\Big[(\boldsymbol{\gamma}_j \odot \widehat{\mathbf{h}})^\top (\boldsymbol{\gamma}_k \odot \widehat{\mathbf{h}})\Big] \\ + \mathbb{E}\Big[(\boldsymbol{\gamma}_j \odot \widehat{\mathbf{h}})^\top \boldsymbol{\beta}_k\Big] + \mathbb{E}\Big[\boldsymbol{\beta}_j^\top (\boldsymbol{\gamma}_k \odot \widehat{\mathbf{h}})\Big] + \boldsymbol{\beta}_j^\top \boldsymbol{\beta}_k.\end{aligned} \tag{31}$$

The two cross terms are small when $\mathbb{E}[\widehat{h}_i] \approx 0$, leaving

$$\mathbb{E}[\mathbf{y}_j^\top \mathbf{y}_k] \approx \mathbb{E}\left[\sum_{i=1}^{D} \gamma_{j,i}\gamma_{k,i}\widehat{h}_i^2\right] + \boldsymbol{\beta}_j^\top \boldsymbol{\beta}_k. \tag{32}$$

Using $\mathbb{E}[\widehat{h}_i^2] \approx 1$ then gives

$$\mathbb{E}[\mathbf{y}_j^\top \mathbf{y}_k] \approx \langle \boldsymbol{\gamma}_j, \boldsymbol{\gamma}_k \rangle + \langle \boldsymbol{\beta}_j, \boldsymbol{\beta}_k \rangle. \tag{33}$$

Because $\boldsymbol{\gamma}_k$ controls channel-wise scaling, reducing alignment among the scaling vectors encourages the routed responses to provide different modulation directions. We leave $\boldsymbol{\beta}_k$ unconstrained so that the responses can share offset corrections. This relation motivates the projection loss in Equation 9. For nonzero scaling vectors and negligible numerical stabilizer, each term is

$$\frac{\langle \boldsymbol{\gamma}_j, \boldsymbol{\gamma}_k \rangle^2}{\|\boldsymbol{\gamma}_k\|_2^2} = \|\boldsymbol{\gamma}_j\|_2^2 \cos^2(\boldsymbol{\gamma}_j, \boldsymbol{\gamma}_k). \tag{34}$$

The loss therefore penalizes both alignment and magnitude, as stated in the main text. The cosine and norm analyses in Appendix G show that it learns distinct, non-collapsed expert directions.

# D Implementation and optimization settings

## D.1 Training protocol and baseline implementation

**Training protocol.** All methods train the detector backend together with their designated encoder adaptation parameters. We use AASIST (Jung et al., 2022) and Conformer-TCM (Truong et al., 2024) as backends and follow the corresponding backbone–detector training protocols (Tak et al., 2022b; Guo et al., 2024; Truong et al., 2024). For XLSR-AASIST, full fine-tuning uses batch size 20, while the remaining methods use batch size 32. The optimizer, learning rate, schedule, and weight decay for each method are listed in Table 8.

**Data augmentation.** We apply RawBoost-4 (Tak et al., 2022a)[1] during training to increase acoustic diversity, following the main-text training protocol.

**BitFit and bottleneck adapters.** BitFit (Ben Zaken et al., 2022) updates the 283,264 bias parameters in XLS-R and the AASIST backend, while freezing the remaining encoder parameters. The Houlsby baseline (Houlsby et al., 2019) freezes the pre-trained encoder and inserts two serial residual bottleneck adapters into each of its 24 Transformer blocks, one after self-attention and one after the feed-forward sublayer. Each adapter uses bottleneck dimension $d_a = 4$, giving a trainable parameter budget close to that of MoLN.

**LoRA and LoRA-MoE.** Our LoRA implementation follows the baseline protocol in (Laakkonen et al., 2026). For routed projection-space adaptation, we implement a dense soft LoRA-MoE variant based on Laakkonen et al. (2025). It replaces the learned noisy gating mechanism with a deterministic dense soft router, matching the implementation distinction stated in the main comparison. The LoRA ranks, target projections, and expert counts are specified in the corresponding result tables.

[1] https://github.com/TakHemlata/RawBoost-antispoofing

**Model selection.** The default MoLN expert count is $K = 3$ for XLSR-AASIST and XLSR-Conformer-TCM, and $K = 2$ for WavLM-AASIST. These counts and the auxiliary-loss weights, $\lambda_{\text{orth}} = \lambda_{\text{lb}} = 0.1$, are selected using the ASVspoof 2019 LA development set and fixed before external evaluation. MoLN training uses only binary bona fide/spoof labels; attack identifiers are used for diagnostic grouping, not as routing supervision.

### D.2 Optimization protocols

We follow the optimization protocols of the corresponding backbone–detector baselines (Tak et al., 2022b; Guo et al., 2024; Truong et al., 2024) for full fine-tuning, backend-only tuning, BitFit, LN tuning, the scale-only adapter, the Houlsby adapter, and MoLN. These methods use Adam (Kingma & Ba, 2015). The LoRA and LoRA-MoE settings use AdamW (Loshchilov & Hutter, 2019) and follow the cyclical learning-rate protocol in (Laakkonen et al., 2026); Table 8 lists the settings for these method families. The main XLSR-AASIST comparison reports mean $\pm$ standard deviation over two random seeds; the design and auxiliary-loss ablations report single runs.

Table 8: Optimization settings used for each method family. All configurations use early stopping with patience 10.

| Method family | Optimizer | Learning rate / schedule | Weight decay | Early stop |
|---|---|---|---|---|
| Full fine-tuning | Adam | $10^{-6}$ | $10^{-4}$ | 10 |
| Backend-only tuning | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| BitFit | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| LN tuning ($\gamma, \beta$) | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| Scale-only LN adapter | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| Houlsby adapter ($d_a = 4$) | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| MoLN | Adam | $10^{-4}$ | $10^{-4}$ | 10 |
| LoRA / LoRA-MoE | AdamW | CyclicLR: $10^{-6} \rightarrow 10^{-4}$ | $10^{-4}$ | 10 |

## E Additional experiments and hyperparameter sweeps

### E.1 MoLN ablations on XLSR-AASIST

**A compact expert bank with soft composition gives the strongest ablation result.** Table 9 lists the single-run expert-count and routing variants in the main text. Increasing $K$ from 1 to 3 lowers average EER from 13.01% to 7.33%; $K = 5$ and $K = 10$ give 9.96% and 9.42%, respectively. Thus, adding experts beyond the compact $K = 3$ bank does not improve average transfer. The $K = 1$ control retains the router code path and insertion sites, but its sole mixture probability is one, so its affine response is shared across inputs and frames. Top-1 routing raises average EER to 13.51%, demonstrating the benefit of soft composition. Regularizing both $\gamma$ and $\beta$ raises it to 11.95%, supporting scaling-only diversity regularization with unconstrained offset corrections. Appendix F reports the auxiliary-loss sweep.

Table 9: MoLN expert-count and routing sweep on XLSR-AASIST. All EER values are percentages from single runs, matching the main-text ablations.

| Method | Configuration | Trainable params. (M) | 21DF | ITW | FoR | ASV5 |
|---|---|---|---|---|---|---|
| MoLN | $K = 10$; soft routing | 1.97 | 3.00 | 8.37 | 6.15 | 20.14 |
| MoLN | $K = 5$; soft routing | 1.06 | 3.01 | 8.70 | 5.57 | 22.56 |
| MoLN | $K = 3$; $\lambda_{orth}$=$\lambda_{lb}$=0.1 | 0.90 | 2.44 | 6.71 | 3.67 | 16.49 |
| MoLN | $K = 3$; top-1 routing | 0.90 | 4.53 | 13.13 | 16.38 | 19.99 |
| MoLN | $K = 3$; orth. on $\gamma, \beta$ | 0.90 | 3.36 | 13.13 | 9.99 | 21.32 |
| MoLN | $K = 2$; soft routing | 0.75 | 2.68 | 8.65 | 6.67 | 18.78 |
| MoLN | $K = 1$; soft routing | 0.60 | 3.68 | 10.32 | 14.22 | 23.82 |

### E.2 Additional non-routed PEFT baselines

**MoLN improves on a closely parameter-matched bottleneck adapter.** Table 10 gives the two-seed results used in the main comparison. BitFit updates all 283,264 XLS-R bias parameters and the detector backend, for 730,506 trainable parameters in total. The Houlsby model trains 48 bottleneck adapters with $d_a = 4$ and the backend, for 889,802 trainable parameters, closely matching MoLN's 903,584-parameter budget.

Houlsby achieves 9.08% average EER, while MoLN achieves 7.55%, a 16.9% relative reduction at a comparable parameter budget. MoLN improves on Houlsby on all four benchmarks, with particularly clear gains on ITW and FoR. This comparison supports the benefit of frame-level conditional LN adaptation beyond parameter count alone. BitFit achieves 12.81% average EER, with improvements over backend-only tuning on 21DF and ITW but higher EER on FoR and ASV5.

Table 10: BitFit, Houlsby, and MoLN on XLSR-AASIST. EER (%) is reported as mean ± standard deviation over two random seeds, matching the main comparison. Avg. is the unweighted mean over the four benchmarks.

| Method | Trainable params. (M) | 21DF | ITW | FoR | ASV5 | Avg. |
|---|---|---|---|---|---|---|
| BitFit | 0.73 | $3.61 \pm 0.40$ | $12.14 \pm 0.53$ | $16.83 \pm 6.80$ | $18.65 \pm 0.04$ | $12.81 \pm 1.48$ |
| Houlsby ($d_a = 4$) | 0.89 | $2.67 \pm 0.24$ | $8.05 \pm 1.41$ | $5.29 \pm 0.76$ | $20.30 \pm 3.46$ | $9.08 \pm 0.76$ |
| MoLN ($K = 3$) | 0.90 | $2.38 \pm 0.08$ | $6.69 \pm 0.03$ | $4.30 \pm 0.90$ | $16.84 \pm 0.49$ | $7.55 \pm 0.32$ |

### E.3 LoRA and LoRA-MoE sweeps

**MoLN provides the strongest average transfer among the compared adaptation methods.** Table 11 separates published references, the two-seed comparisons in the main text, and additional single-run configurations. Reproduced rank-16 LoRA achieves 8.29% average EER and remains competitive on ASV5; MoLN improves on 21DF, ITW, and FoR and achieves 7.55% average EER. The dense soft LoRA-MoE variant gives 8.62%, and LoRA+MoLN gives 9.16%. These results support the importance of both the adaptation interface and the composition of conditional updates. The additional configurations retain their single-run status and use the method-specific optimization protocols in Appendix D. The rank-8 supplementary run is reported separately from the rank-analysis result in the main text (OOD EER 5.02%); these results are not pooled or used interchangeably.

Table 11: LoRA, LoRA-MoE, and LoRA+MoLN results on XLSR-AASIST. † denotes the five-seed mean ± standard deviation reported by Laakkonen et al. (2026); ‡ denotes point estimates from Laakkonen et al. (2025). Both cited works evaluate FakeAVCeleb rather than FoR, so FoR is unavailable for these rows. Our main-comparison rows report two-seed mean ± standard deviation; additional configurations are single runs. Our LoRA-MoE uses a deterministic dense soft router in place of learned noisy gating. All EER values are percentages.

| Method | Configuration | Trainable params. (M) | 21DF | ITW | FoR | ASV5 |
|---|---|---|---|---|---|---|
| | | ***Reported references*** | | | | |
| LoRA‡ | Q,K,V,out; $r = 4$ | 1.23 | 4.37 | 10.69 | – | 25.97 |
| LoRA‡ | Q,K,V,out; $r = 8$ | 2.02 | 5.02 | 13.15 | – | 21.05 |
| LoRA† | Q,K,V,out; $r = 16$ | 3.59 | 4.35±1.37 | 11.05±6.27 | – | 20.05±4.36 |
| LoRA-MoE‡ | Q,K,V,out; $K = 3$; $r = 4$ | 3.40 | 6.78 | 10.60 | – | 22.49 |
| | | ***Main comparison: two random seeds*** | | | | |
| LoRA | Q,K,V,out; $r = 16$ | 3.59 | $3.09 \pm 0.54$ | $9.26 \pm 0.06$ | $5.09 \pm 2.82$ | $15.75 \pm 1.06$ |
| LoRA-MoE | Q,K,V,out; $r = 4$; $K = 3$ | 3.10 | $2.82 \pm 0.35$ | $9.07 \pm 0.13$ | $6.12 \pm 0.59$ | $16.47 \pm 2.52$ |
| LoRA+MoLN | Q,K,V,out; $r = 4$; $K = 3$ | 1.69 | $2.50 \pm 0.29$ | $8.23 \pm 0.94$ | $6.29 \pm 1.60$ | $19.63 \pm 0.45$ |
| | | ***Additional single-run configurations*** | | | | |
| LoRA | Q,K,V,out; $r = 4$ | 1.23 | 2.82 | 10.20 | 8.65 | 17.20 |
| LoRA | Q,K,V,out; $r = 8$ | 2.02 | 6.66 | 20.95 | 12.71 | 15.78 |
| LoRA+MoLN | Q,K,V,out; $r = 4$; $K = 3$; $\lambda = 0.01$ | 1.69 | 3.20 | 10.21 | 12.99 | 18.38 |
| LoRA+MoLN | Q,K,V,out; $r = 8$; $K = 3$; $\lambda = 0.01$ | 2.48 | 3.11 | 7.82 | 14.01 | 18.87 |

### E.4 Backbone expert-count sweeps

**Compact conditional LN adaptation generalizes across encoder–detector architectures.** Table 12 reports the expert-count sweep for WavLM-AASIST and XLSR-Conformer-TCM. With source-

selected defaults of $K = 2$ and $K = 3$, respectively, MoLN achieves average EERs of 15.56% and 13.41%, compared with 19.28% and 22.55% for LN tuning. WavLM-AASIST favors $K = 2$ on ITW, FoR, ADDR1, and ADDR2, whereas XLSR-Conformer-TCM favors $K = 3$ on those benchmarks among the tested MoLN counts. The gains support a small expert bank, with the default count determined on source-development data for each architecture.

Table 12: Full WavLM-AASIST and XLSR-Conformer-TCM MoLN expert sweeps. EER values are percentages.

| Model | Configuration | Trainable params. (M) | 21DF | ITW | FoR | ASV5 | ADDR1 | ADDR2 |
|---|---|---|---|---|---|---|---|---|
| WavLM-AASIST | backend-only | 0.45 | 6.85 | 16.36 | 12.32 | 17.69 | 37.61 | 33.39 |
| WavLM-AASIST | Full fine-tuning | 315.90 | 6.07 | 15.88 | 23.98 | 18.80 | 39.90 | 31.79 |
| WavLM-AASIST | LN tuning ($\gamma, \beta$) | 0.56 | 5.66 | 13.55 | 16.03 | 14.36 | 34.64 | 31.43 |
| WavLM-AASIST | MoLN; $K = 4$ | 1.06 | 8.17 | 17.24 | 14.80 | 16.11 | 33.89 | 32.15 |
| WavLM-AASIST | MoLN; $K = 3$ | 0.90 | 6.58 | 13.18 | 16.53 | 13.66 | 25.81 | 24.30 |
| WavLM-AASIST | MoLN; $K = 2$ | 0.75 | 6.60 | 12.80 | 11.65 | 16.43 | 23.66 | 22.21 |
| WavLM-AASIST | MoLN; $K = 1$ | 0.60 | 7.10 | 16.17 | 14.44 | 15.46 | 40.86 | 38.01 |
| XLSR-Conformer-TCM Truong et al. (2024) | - | 319.00 | 2.15 | 7.79 | 10.69 | – | – | – |
| XLSR-Conformer-TCM | Full fine-tuning | 319.00 | 2.25 | 7.19 | 7.94 | 16.74 | 29.89 | 27.50 |
| XLSR-Conformer-TCM | LN tuning ($\gamma, \beta$) | 2.38 | 9.41 | 31.45 | 22.18 | 25.05 | 25.93 | 21.29 |
| XLSR-Conformer-TCM | MoLN; $K = 4$ | 2.99 | 3.04 | 10.35 | 10.47 | 16.44 | 28.15 | 25.77 |
| XLSR-Conformer-TCM | MoLN; $K = 3$ | 2.84 | 2.90 | 7.08 | 6.24 | 18.09 | 25.22 | 20.92 |
| XLSR-Conformer-TCM | MoLN; $K = 2$ | 2.69 | 2.81 | 8.29 | 7.08 | 15.38 | 28.51 | 28.41 |
| XLSR-Conformer-TCM | MoLN; $K = 1$ | 2.54 | 4.90 | 9.19 | 10.42 | 17.04 | 31.03 | 26.55 |

# F AUXILIARY-LOSS WEIGHT ANALYSIS

MoLN weights the orthogonality and load-balancing losses by $\lambda_{orth}$ and $\lambda_{lb}$. The main setting, $\lambda_{orth} = \lambda_{lb} = 0.1$, is selected on the ASVspoof 2019 LA development set before external evaluation. Table 13 evaluates matched values of $0.01$, $0.1$, and $0.2$ as a post-selection sensitivity analysis. It also includes models without either loss, without load balancing ($\lambda_{orth} = 0.1$, $\lambda_{lb} = 0$), and without orthogonality regularization ($\lambda_{orth} = 0$, $\lambda_{lb} = 0.1$). All configurations use $K = 3$, soft routing, and regularization on $\gamma$ only. The source-selected setting remains fixed across external benchmarks.

Table 13: Auxiliary-loss weight analysis on XLSR-AASIST. EER (%) is reported for each evaluation set; Avg. is the unweighted mean over the four sets. All results are from single runs.

| $\lambda_{orth}$ | $\lambda_{lb}$ | 21DF | ITW | FoR | ASV5 | Avg. |
|---|---|---|---|---|---|---|
| 0 | 0 | **2.33** | 9.88 | 5.87 | 21.27 | 9.84 |
| 0.01 | 0.01 | 3.03 | 8.38 | 8.78 | 21.83 | 10.51 |
| 0.1 | 0 | 3.40 | 11.49 | 6.58 | 18.31 | 9.95 |
| 0 | 0.1 | 3.63 | 8.62 | 21.34 | 18.70 | 13.07 |
| 0.1 | 0.1 | 2.44 | **6.71** | **3.67** | **16.49** | **7.33** |
| 0.2 | 0.2 | 3.61 | 9.28 | 10.21 | 18.87 | 10.49 |

**Expert diversity and balanced usage jointly support cross-dataset generalization.** The source-selected value of 0.1 gives the lowest average external EER (7.33%) and the best results on ITW, FoR, and ASV5 among the tested settings. Weaker (0.01) and stronger (0.2) weights give average EERs of 10.51% and 10.49%, respectively. Removing load balancing raises average EER to 9.95%; removing orthogonality regularization raises it to 13.07%; removing both gives 9.84%. The loss-free model improves slightly on 21DF (2.33% versus 2.44%) but is worse on the other three benchmarks. These ablations support the joint use of complementary expert directions and balanced routing. Figure 3 and Appendix G connect this performance pattern to the learned expert geometry and routing distributions.

# G ADDITIONAL MECHANISM VISUALIZATIONS

## G.1 EFFECT OF ORTHOGONALITY ON SCALING RESPONSES

Figure 6 shows the pairwise cosine similarities of $\gamma$ responses at every MoLN layer. Without the orthogonality loss, several layers contain highly aligned scaling vectors. Some pairs in later layers are nearly collinear. The average absolute off-diagonal cosine across 50 layers is 0.6172.

With $\mathcal{L}_{orth}$, the average drops to $1.07\times10^{-4}$ and remains near zero across the network. Together with the non-collapsed norms below, this demonstrates that the loss learns distinct residual scaling directions throughout the expert bank.

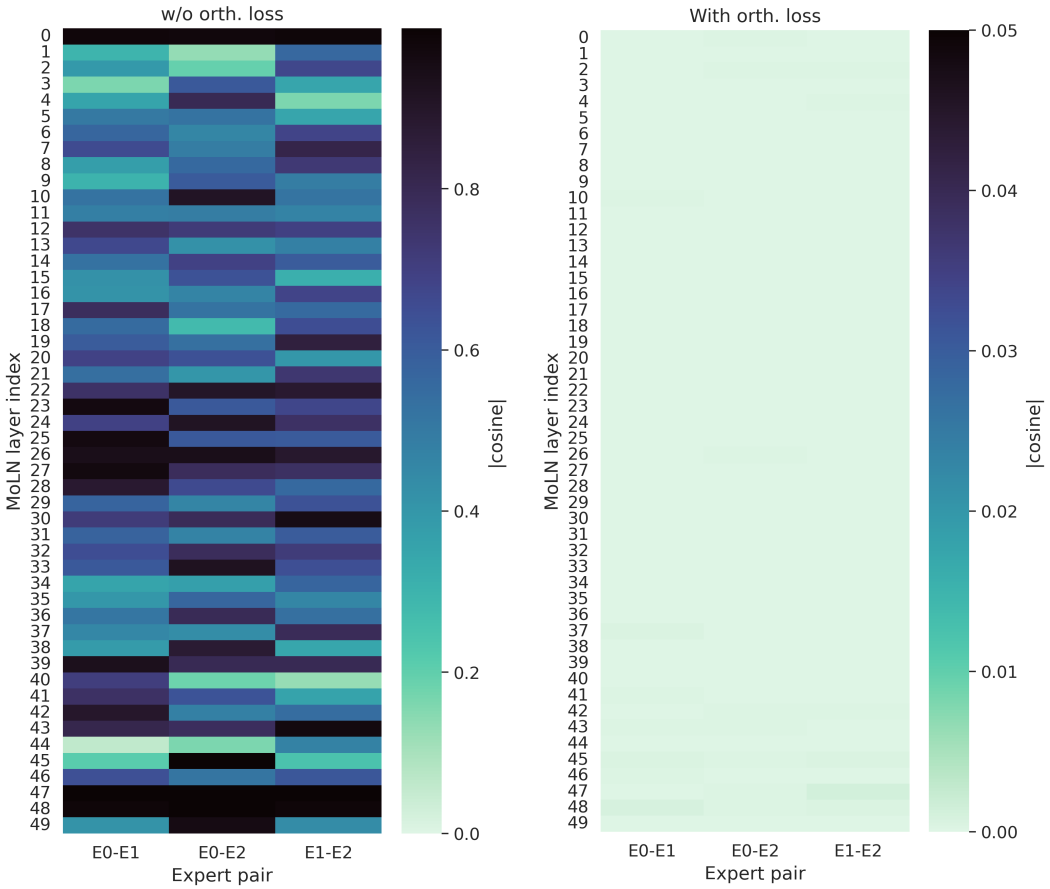



Figure 6: Pairwise $\gamma$-expert cosine similarities across MoLN layers, without orthogonality loss (left) and with orthogonality loss (right).

### G.2 Scaling-response norms

The projection-based loss is not scale invariant, whereas the cosine-similarity analysis above removes response magnitude. We therefore compare the response root-mean-square norm

$$r_{\ell,k} = \frac{\|\boldsymbol{\gamma}_{\ell,k}\|_2}{\sqrt{D}} \tag{35}$$

with and without $\mathcal{L}_{orth}$ at every MoLN layer. Figure 7 summarizes all $50 \times 3 = 150$ responses. With the loss, the mean and median RMS norms are 0.00767 and 0.00789, compared with 0.00957 and 0.00939 without it. The mean response magnitude is therefore about 20% lower, but remains nonzero and within the same order of magnitude throughout the network. This moderate reduction cannot account for the change in mean absolute cosine similarity from 0.6172 to $1.07\times10^{-4}$. The observed directional separation is therefore not a trivial consequence of collapsing the response norms.

For each layer, the solid curve is the arithmetic mean of $r_{\ell,k}$ over the three responses. The shaded region spans the response-wise range $[\min_k r_{\ell,k}, \max_k r_{\ell,k}]$ at that layer. The shading summarizes within-layer variation among the three experts at the analyzed checkpoint.

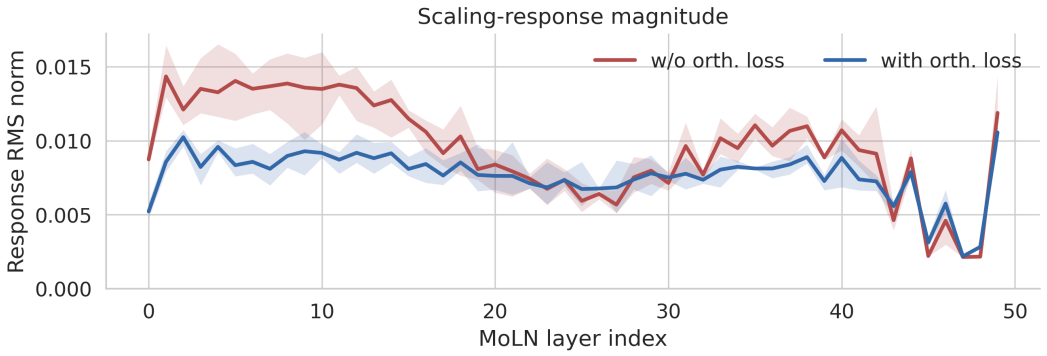



Figure 7: Layer-wise RMS norms $\|\gamma_{\ell,k}\|_2/\sqrt{D}$ of the residual scaling responses, comparing MoLN trained with and without $\mathcal{L}_{orth}$. The solid line is the mean over the three responses at each layer, and the shading spans their minimum and maximum.

### G.3 Effective scaling after routing

The individual response norms do not fully describe the modulation applied to an input because the router combines the responses at each frame. We therefore measure the RMS norm of the routed scaling,

$$r_\ell^{\text{eff}} = \mathbb{E}_{x,t}\left[\frac{\left\|\sum_{k=1}^{K} p_{\ell,t,k}(x)\gamma_{\ell,k}\right\|_2}{\sqrt{D}}\right], \tag{36}$$

where the expectation is taken over the analyzed inputs and fixed-length frame positions. Figure 8 shows that the effective scaling remains nonzero at every layer, although it is generally lower with $\mathcal{L}_{orth}$. This pattern is expected when less aligned responses are combined by a soft router. For equally weighted responses with equal norm $r$, the mixture norm is $r$ when the responses are aligned and $r/\sqrt{K}$ when they are mutually orthogonal. The lower routed norm therefore reflects less constructive addition among responses, together with the moderate reduction in their individual norms.

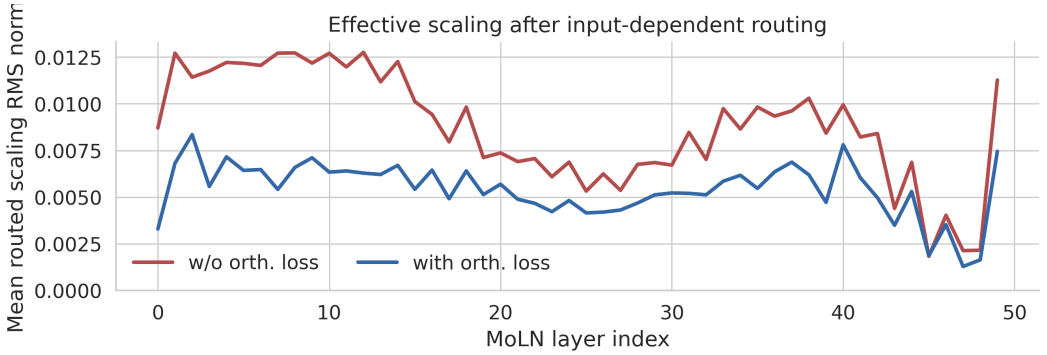



Figure 8: Mean RMS norm of the effective scaling after input-dependent routing across the 50 MoLN layers, comparing models trained with and without $\mathcal{L}_{orth}$. Each layer-level value averages 804,000 frame positions from 2,000 ASVspoof 2019 LA training utterances and 2,000 ASVspoof 2021 DF utterances.

### G.4 Routing shifts across layers between evaluation sources

**Data and aggregation.** We evaluate the $K = 3$ XLSR-AASIST checkpoints used in Sections G.1 and G.2 in evaluation mode, without optimizer updates. The analysis uses the first 2,000 entries, in protocol order, from the ASVspoof 2019 LA training protocol and the first 2,000 entries from

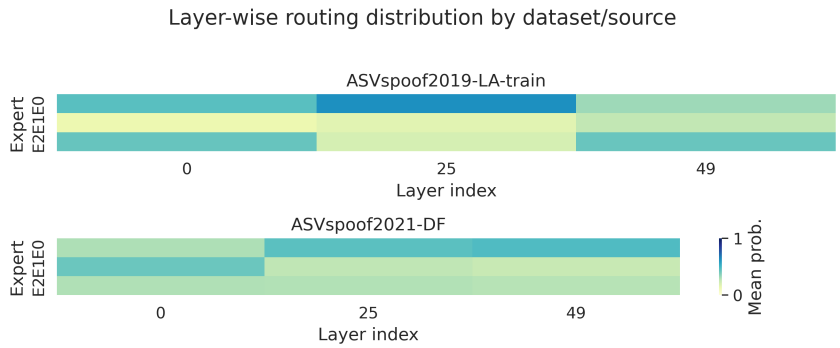



Figure 9: Group-mean routing distributions for 2,000 ASVspoof 2019 LA training utterances and 2,000 ASVspoof 2021 DF utterances at MoLN layers 0, 25, and 49. The averages describe expert usage on the two protocol-ordered subsets; their label compositions are specified below.

the ASVspoof 2021 DF trial metadata. RawBoost is disabled. Each waveform is loaded at 16 kHz and repeated or truncated to 64,600 samples, following the evaluation pipeline. This produces 201 XLS-R frame positions per utterance. Consequently, each dataset contributes 402,000 routed frame positions per layer, and each point on the combined curve averages 804,000 positions. The same 4,000 utterances are used for the checkpoints trained with and without $\mathcal{L}_{orth}$.

Combined with Figure 7, this result shows that the orthogonality loss changes the geometry of the routed modulation without eliminating it.

Figure 9 compares routing distributions at selected early, middle, and final MoLN layers for ASVspoof 2019 LA training data and ASVspoof 2021 DF. Their average mixtures differ at several depths. At the final layer, the mean mixture changes from $(0.32, 0.26, 0.42)$ on ASVspoof 2019 LA to $(0.47, 0.25, 0.28)$ on ASVspoof 2021 DF, with normalized entropies of 0.98 and 0.96, respectively, as reported in the main text. Multiple experts remain active at the aggregate level, and their relative contributions vary across the analyzed inputs rather than reducing to one fixed global mixture.

**Data, routing average, and entropy.** This analysis uses the checkpoint trained with $\mathcal{L}_{orth}$ and the same two protocol-ordered subsets of 2,000 utterances described in Section G.3. At layer $\ell$, we first average the frame-level routing probabilities within utterance $i$, and then average over the $N_g$ utterances in group $g$:

$$\bar{p}_{\ell,g,k} = \frac{1}{N_g} \sum_{i \in g} \left( \frac{1}{T_i} \sum_{t=1}^{T_i} p_{\ell,t,k}(x_i) \right). \tag{37}$$

The entropy reported in the routing visualizations is the normalized entropy of this group-mean distribution,

$$H_{\ell,g} = -\frac{1}{\log K} \sum_{k=1}^{K} \bar{p}_{\ell,g,k} \log\big(\max(\bar{p}_{\ell,g,k}, 10^{-12})\big), \tag{38}$$

with $K = 3$. Thus, $H = 0$ indicates aggregate use of one response and $H = 1$ indicates uniform aggregate use. This entropy measures aggregate expert-usage balance; per-frame routing softness is specified by the softmax architecture and evaluated through the top-1 ablation. Figure 9 reports layers 0, 25, and 49, with $N_g = 2{,}000$ for each dataset at every layer.

Routing changes with depth as well as across the analyzed input groups. Together with the expert-direction analysis and the top-1 ablation, these results support MoLN's combination of a diverse affine expert bank and representation-conditioned soft composition. The 2019 subset contains 2,000 bona fide utterances with attack field –, while the 2021 subset contains 74 bona fide and 1,926 spoof utterances. The comparison therefore characterizes routing on these observed input distributions, which differ in both dataset and label composition.

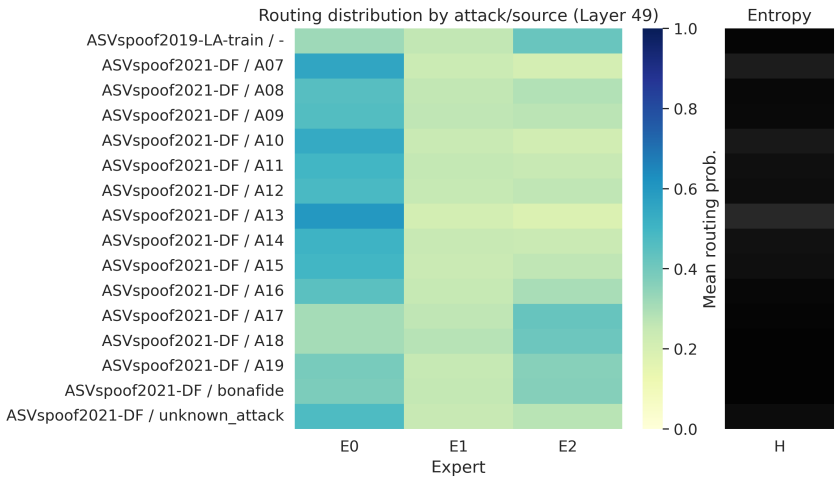



Figure 10: Routing distributions by attack/source at the final MoLN layer.

### G.5 Routing grouped by attack metadata

Figure 10 groups final-layer routing on ASVspoof 2021 DF by the available attack/source metadata. A07, A10, and A13 place relatively more weight on response 0, while A17 and A18 place more weight on response 2. This within-dataset variation further supports input-dependent use of the expert bank under heterogeneous speech conditions. Attack/source metadata are used only to summarize routing after training; MoLN learns its routing from binary detection supervision without explicit condition labels.