

REIMU: Efficient Heterogeneous Hierarchical Reasoning for SSL-Based Speech Deepfake Detection

Kwok-Ho Ng¹, Tingting Song^{1*}, Bingwen Feng¹, Peiya Li¹

¹ College of Cyber Security, Jinan University, Guangzhou, China
kwokhong@stu2024.jnu.edu.cn, tingtingsong@jnu.edu.cn

Abstract

The increasing realism of speech generated by text-to-speech and voice conversion systems poses growing challenges to media integrity and voice authentication. Self-supervised learning (SSL) has substantially advanced speech deepfake detection, where downstream backbones conventionally process SSL representations through a single forward pass. This work investigates the practical effectiveness of recurrent hierarchical reasoning for this task. We term this controlled study REIMU and systematically compare conventional single-pass backbones, weight-shared recurrence, homogeneous HRM, and heterogeneous HRM across four Base-scale SSL frontends. We further examine heterogeneous high- and low-level modules that combine self-attention with linear attention. Experiments on the ASVspoof 2019 and 2021 evaluation sets show that recurrence and hierarchical decomposition do not inherently improve detection, whereas heterogeneous operator assignment provides a more competitive configuration. Notably, the heterogeneous design remains competitive while using 10.8% fewer downstream parameters than the matched baseline, demonstrating its potential for parameter-efficient speech deepfake detection.

Code — <https://github.com/saki-ciallo/REIMU-SDD>

Introduction

Recent advances in text-to-speech (TTS) synthesis and voice conversion (VC) have enabled high-quality synthetic speech to be widely adopted in content creation and assistive applications (Hu et al. 2026). However, the malicious deployment of these technologies facilitates voice cloning, identity impersonation, misinformation, and presentation attacks against voice authentication systems, posing concrete threats to digital media integrity and information security (Li, Ahmadiadli, and Zhang 2025). To counter these risks, benchmarks such as the ASVspoof series (Wang et al. 2020; Yamagishi et al. 2021) have continuously driven research on speech anti-spoofing and deepfake detection. The AT-ADD challenge (Xie et al. 2026a) further broadens the scope to realistic conditions across diverse audio types, including speech, environmental sounds, singing, and music, while multilingual datasets like MLAAD (Müller et al. 2024) evaluate generalization to unseen languages and synthesis engines. Consequently, exploring and constructing reliable speech deepfake

detection (SDD) systems has become a critical research priority in audio security.

Early SDD architectures focused on learning discriminative spoofing cues directly from raw waveforms. For instance, RawNet2 established end-to-end neural network paradigms to learn artifact patterns without relying on handcrafted acoustic features (Tak et al. 2021b). Building upon raw waveform modeling, RawGAT-ST (Tak et al. 2021b) introduced joint temporal and spectral graph attention to capture spoofing artifacts across time and frequency regions effectively. Subsequently, AASIST (Jung et al. 2022) incorporated heterogeneous spectro-temporal graph attention and graph pooling strategies, becoming a widely adopted end-to-end baseline architecture in speech anti-spoofing research.

With the rapid progress of self-supervised speech learning, encoders pretrained on massive unlabelled corpora can be effectively transferred to SDD. Combining an SSL frontend with a downstream classifier has thus emerged as a dominant paradigm. For instance, XLSR-AASIST (Tak et al. 2022) utilized wav2vec 2.0 XLSR (300M) as a feature extractor. It achieved substantial performance gains on the ASVspoof 2021 evaluation sets. Similar efforts introduced Conformer backbones combining convolution and self-attention (Rosello et al. 2023). Additionally, XLSR-Mamba (Xiao and Das 2025) leveraged a dual-column bidirectional Mamba backbone to model local and long-range dependencies with linear complexity. Beyond backbone enhancements, other research explores extracting richer representations from SSL encoders (El Kheir et al. 2025). For example, XLSR-SLS (Zhang, Wen, and Hu 2024) treated all intermediate layers of the pretrained Transformer as hierarchical features and learned sensitivity weights for aggregation. This approach leveraged the rich spoofing artifacts preserved across different SSL hidden layers. Similarly, parameter-efficient adaptation techniques demonstrate effective representation alignment with minimal trainable parameters. A prominent example learns wavelet prompts inside a frozen SSL encoder paired with an AASIST backbone (Xie et al. 2026b).

Overall, existing studies primarily enhance modeling capacity by adapting SSL frontends, aggregating layer-wise features, or designing complex downstream backbones. Nevertheless, most backbones still transform input features through a single forward pass of fixed depth. It remains in-

*Corresponding author.

sufficiently explored whether a compact backbone can iteratively refine latent representations through parameter reuse rather than continuously adding new parameters. Specifically, when the SSL frontend already yields frame-level features, the impact of weight-shared recurrence on artifact extraction is not fully understood. Furthermore, it is worth examining whether modules with different update frequencies should employ complementary sequence-modeling operators.

To achieve deep computation within constrained parameter budgets, the Hierarchical Reasoning Model (HRM) (Wang et al. 2025) was recently proposed. It utilizes two interacting recurrent modules operating at distinct update frequencies. The high-level module updates an abstract latent state at a lower frequency, while the low-level module performs granular computations more frequently. These intermediate operations are conducted entirely in continuous latent space. The subsequent HRM-Text framework (Wang et al. 2026) extended this architecture to language modeling. It achieved competitive performance against larger open models while using substantially less pretraining data and compute.

However, the underlying mechanisms of the dual-module H/L architecture remain actively discussed in the literature. Subsequent studies reported that simple, approximately equivalent Transformer (Vaswani et al. 2017) baselines can achieve comparable performance under equivalent settings (Ge, Liao, and Poggio 2025). These observations suggest that performance gains from recurrent computation should not be automatically attributed to hierarchical design alone. Therefore, when transferring HRM to speech deepfake detection, it is essential to reframe it as a mechanism for recurrent latent feature refinement. Furthermore, it is necessary to systematically isolate the distinct effects of standard single forward pass depth, simple weight-shared recurrence, and true hierarchical recurrence with separately parameterized modules.

Motivated by these considerations, this paper investigates the applicability of recurrent hierarchical architectures for speech deepfake detection using self-supervised representations. We refer to this controlled investigation of recurrent and heterogeneous hierarchical processing for SSL-based SDD as REIMU. Unlike symbolic reasoning, SDD requires identifying subtle local acoustic artifacts from continuous frame-level features and integrating evidence across an utterance. We hypothesize that the H and L modules do not necessarily benefit from identical sequence operators. Specifically, the low-level module can employ linear attention (e.g., Gated DeltaNet-2 (Hatamizadeh, Choi, and Kautz 2026) or Raven (Afzal et al. 2026)) for low-cost recurrent refinements on local features. Meanwhile, the high-level module can use multi-head self-attention (MHSA) for global temporal modeling over refined states.

To validate this hypothesis, we establish a controlled benchmarking framework using four 95M-parameter (approximately) SSL frontends: wav2vec 2.0 Base (Baevski et al. 2020), HuBERT Base (Hsu et al. 2021), WavLM Base (Chen et al. 2022), and WavLM Base+. Under frozen-encoder settings, we systematically evaluate standard single forward pass baselines, simple weight-shared Looped

models (Yang et al. 2024), and both homogeneous and heterogeneous HRM configurations across multiple execution schedules (H_2L_1, H_2L_2, H_2L_3). In Looped models, gradients participate only in the final module call. Based on overall performance, Gated DeltaNet-2 (GDN2) is selected as the core sequence operator for further ablation studies. We unfreeze top SSL layers and apply a unified data augmentation setup to assess the performance gains of selective adaptation.

The primary contributions of this work are summarized as follows:

- To the best of our knowledge, this work innovatively presents a controlled investigation of hierarchical recurrent architectures for SDD.
- We conduct a unified comparison across four 95M-parameter SSL frontends under frozen-encoder conditions, assessing downstream architectural performance and generalization across diverse speech representations.
- We explicitly disentangle the effects of standard single forward pass computation, single-module weight-shared Looped refinement, homogeneous HRM, and heterogeneous HRM under truncated-gradient constraints.
- We investigate heterogeneous operator assignment across update frequencies by combining a high-level MHSA module with low-level linear attention operators (GDN2 and Raven). Experimental results demonstrate that under matched settings, introducing operator heterogeneity (e.g., $H_{\text{MHSA}} + L_{\text{GDN2}}$) reduces downstream backbone parameters by 10.8% while achieving competitive equal error rates relative to standard baselines on the ASVspoof 2019 and 2021 evaluation sets.

Methodology

Framework Definition

Figure 1 illustrates the pipeline of the proposed experimental method. Given a raw speech waveform $x \in \mathbb{R}^T$ of length T , the goal of SDD is to predict a continuous spoofing score $\hat{y} \in [0, 1]$, where values closer to 0 and 1 represent bonafide and spoofed speech, respectively.

$$\hat{y} = C(P(B(\text{Linear}(F(x)))) \quad (1)$$

where $F(\cdot)$ denotes the frozen or partially fine-tuned SSL frontend, $\text{Linear}(\cdot)$ maps the SSL representations to the hidden dimension of the backbone, $B(\cdot)$ represents the sequence modeling backbone, $P(\cdot)$ denotes the pooling operator, and $C(\cdot)$ is the final binary classifier.

SSL Frontend

To evaluate the impact of frontend representations on downstream backbones under controlled conditions, we select four SSL encoders with comparable parameter scale ($\approx 95\text{M}$) and identical output embedding dimensions. Given an input waveform x , the feature extraction process is expressed as:

$$z_{\text{ssl}} = F(x) \in \mathbb{R}^{S \times D}, \quad (2)$$

where S denotes the subsampled frame sequence length and D represents the SSL hidden dimension.

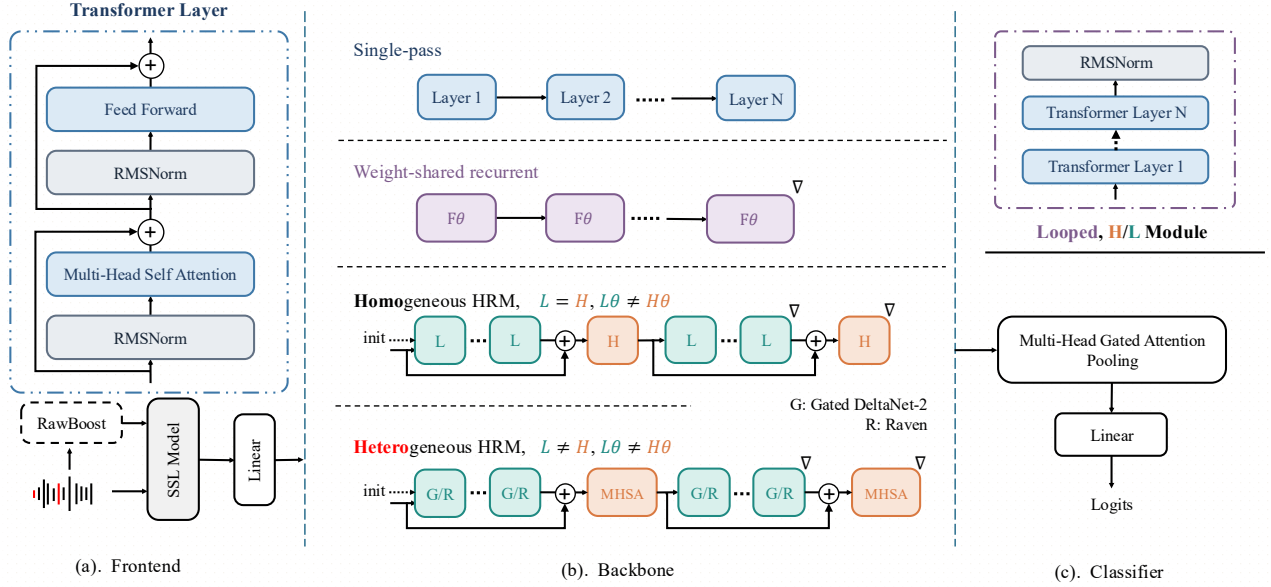


Figure 1: Overview of the structure of the designed experiment.

To bridge the dimension gap between various SSL frontends and downstream backbones, a linear projection layer projects z_{ssl} to the backbone model dimension d :

$$z_0 = \text{Linear}(z_{\text{ssl}}) \in \mathbb{R}^{S \times d}. \quad (3)$$

In main architectural comparisons, all parameters of $F(\cdot)$ remain strictly frozen to isolate the downstream architectural effects. In subsequent fine-tuning experiments, we unfreeze only the last two Transformer blocks of the SSL encoder for selective adaptation.

Unified Backbone and Sequence Mixers

To ensure fair comparisons across different sequence operators, we construct a unified Transformer-like backbone. The backbone consists of N stacked blocks, where each block adopts a pre-normalization (RMSNorm (Zhang and Senrich 2019)) structure with a sequence mixing module and a SwiGLU (Shazeer 2020) Feed-Forward Network (FFN):

$$z^{(l)'} = z^{(l-1)} + M(\text{RMSNorm}(z^{(l-1)})), \quad (4)$$

$$z^{(l)} = z^{(l)'} + \text{SwiGLU}(\text{RMSNorm}(z^{(l)'})), \quad (5)$$

where $l \in \{1, \dots, N\}$ indicates the block index, and $M(\cdot)$ represents the candidate sequence mixing operator. All candidate operators share identical hyperparameters, including the number of attention/state heads and hidden projections.

We evaluate three sequence mixing operators with distinct properties:

- **MHSA** captures global temporal dependencies through pairwise content interactions. It is used in the Transformer baselines and as the high-level operator in heterogeneous HRM.

- **GDN2** compresses temporal history into a fixed-size matrix state with gated linear-complexity updates, making it suitable for frequent low-level refinement.
- **Raven** maintains routed memory slots that are selectively updated to capture localized and transient acoustic artifacts.

Standard Single Forward Pass The standard single forward pass baseline serves as the conventional feed-forward reference. It consists of N sequentially stacked backbone blocks with independent parameters:

$$z^{(l)} = B_l(z^{(l-1)}), \quad l \in \{1, 2, \dots, N\}, \quad (6)$$

where $z^{(0)} = z_0$ is the linearly projected SSL representation from Eq. (3), and $B_l(\cdot)$ denotes the l -th distinct backbone block parameterized by an independent parameter set Θ_l . The final output $z^{(N)}$ is passed to the pooling layer.

To establish fair control baselines, we fix the network depth N across all standard architectures and strictly alter only the internal sequence mixing operator $M(\cdot)$ inside B_l . By substituting this operator, we construct three feed-forward baselines: **MHSA-FFN**, **GDN2-FFN**, and **Raven-FFN**. These baselines represent conventional Transformer-style architectures where feature transformations are executed through a single pass over independently parameterized layers.

Weight-Shared Looped Refinement To isolate the effects of iterative computation from hierarchical state decomposition, we construct a parameter-shared recurrent baseline. Rather than stacking N unique blocks, a single composite backbone module B_{shared} with parameter set Θ_{shared} is applied repeatedly over R recurrent steps in continuous latent space.

To maintain numerical stability during multi-turn recurrent updates and use a similar design in HRM, the recurrent module is structured with an outer post-normalization layer:

$$B_{\text{shared}}(x) = \text{RMSNorm}\left(\text{Transformer}(x)\right), \quad (7)$$

where $\text{Transformer}(\cdot)$ adopts the unified pre-norm block. The composite block B_{shared} is then treated as a unified atomic unit for recurrent execution:

$$z^{(r)} = B_{\text{shared}}\left(z^{(r-1)}\right), \quad r \in \{1, 2, \dots, R\}, \quad (8)$$

where $z^{(0)} = z_0$, and R specifies the total number of recurrent passes. All recurrent steps share the exact same parameters Θ_{shared} .

To ensure memory efficiency during training and prevent gradient explosion, backpropagation is applied exclusively to the final recurrent iteration. Formally, let $\text{sg}(\cdot)$ denote the stop-gradient operator. The complete execution flow is expressed as:

$$\begin{aligned} \tilde{z}^{(0)} &= z_0, \\ \tilde{z}^{(r)} &= \text{sg}\left(B_{\text{shared}}\left(\tilde{z}^{(r-1)}\right)\right), \quad \text{for } r = 1, \dots, R-1, \end{aligned} \quad (9)$$

$$z^{(R)} = B_{\text{shared}}\left(\tilde{z}^{(R-1)}\right). \quad (10)$$

Under this formulation, gradients flow strictly through the final iteration step R , while the previous $R-1$ iterations serve solely to refine the latent representation in a forward-only manner. This baseline enables us to examine whether simple parameter-shared recurrence over a single composite module can match the refinement efficacy of hierarchical H/L architectures.

Homogeneous HRM To achieve deeper effective computation without expanding parameter budgets, the Hierarchical Reasoning Model (HRM) decomposes the backbone into two interacting modules: a High-level module $H(\cdot; \Theta_H)$ and a Low-level module $L(\cdot; \Theta_L)$. To strictly control total parameter capacity, both H and L modules are constructed with $N/2$ Transformer-style blocks (half the depth of the N -block standard baseline). Thus, the combined parameter count of H and L matches a single standard N -block baseline ($\Theta_H + \Theta_L \approx \Theta_{\text{standard}}$). In Homogeneous HRM, H and L employ identical sequence mixing operator types (e.g., both using MHSA or GDN2).

The execution follows a hierarchical schedule denoted as H_2L_k ($k \in \{1, 2, 3\}$), where 2 specifies the total number of outer High-level cycles, and k dictates the number of inner Low-level recurrent refinements preceding each High-level step.

Let z_0 denote the projected SSL representation from Eq. (3), and $h_{\text{init}} \in \mathbb{R}^{S \times d}$ denote a learnable initial high-level state. The recurrent state initialization is formulated as:

$$h^{(0)} = h_{\text{init}}, \quad z_{\text{in}}^{(1,1)} = z_0 + h^{(0)}, \quad (11)$$

where $z_{\text{in}}^{(1,1)}$ serves as the input to the very first Low-level step.

Formally, for the m -th High-level cycle ($m \in \{1, 2\}$) and its n -th Low-level sub-step ($n \in \{1, \dots, k\}$), the state transitions and residual fusion mechanisms are defined recursively as:

$$\text{Low Input: } z_{\text{in}}^{(m,n)} = \begin{cases} z_0 + h^{(m-1)}, & \text{if } n = 1, \\ z_{\text{out}}^{(m,n-1)}, & \text{if } n > 1, \end{cases} \quad (12)$$

$$\text{Low-Level: } z_{\text{out}}^{(m,n)} = L\left(z_{\text{in}}^{(m,n)}; \Theta_L\right), \quad (13)$$

$$\text{High Input: } h_{\text{in}}^{(m)} = h^{(m-1)} + z_{\text{out}}^{(m,k)}, \quad (14)$$

$$\text{High-Level: } h^{(m)} = H\left(h_{\text{in}}^{(m)}; \Theta_H\right), \quad (15)$$

where Eq. (14) explicitly fuses the high-level latent state from the previous cycle $h^{(m-1)}$ with the output of the current final Low-level step $z_{\text{out}}^{(m,k)}$ ($H_1_{\text{output}} + L_2_{\text{output}}$ for $m = 2$) before feeding into the High-level module H .

Under the H_2L_k schedule, the explicit execution sequences are unrolled as follows:

- H_2L_1 ($k = 1$): $L \rightarrow H \rightarrow L \rightarrow H$ (2 outer cycles, each preceded by 1 L -pass).
- H_2L_2 ($k = 2$): $L \rightarrow L \rightarrow H \rightarrow L \rightarrow L \rightarrow H$ (2 outer cycles, each preceded by 2 L -passes).
- H_2L_3 ($k = 3$): $L \rightarrow L \rightarrow L \rightarrow H \rightarrow L \rightarrow L \rightarrow L \rightarrow H$ (2 outer cycles, each preceded by 3 L -passes).

Similar to the Looped baseline, gradient propagation is truncated during intermediate recurrent steps. Backpropagation is executed strictly through the final module calls of the H_2L_k sequence using the stop-gradient operator $\text{sg}(\cdot)$, preventing gradient instability across unrolled computation graphs.

Heterogeneous HRM While Homogeneous HRM enforces H and L to share the same sequence mixing operator, we hypothesize that modules operating at different update frequencies require distinct sequence-modeling inductive biases. To investigate this structural hypothesis, we propose the Heterogeneous Hierarchical Architecture (Hetero-HRM).

Hetero-HRM retains the $N/2$ -block H and $N/2$ -block L setup, the residual fusion mechanism, and the H_2L_k schedule, but decouples the choice of sequence operators across temporal frequencies:

- **High-Level Module (H_{MHSA}):** Operating at a lower execution frequency (2 cycles total), the high-level module consistently employs MHSA. It performs global pairwise temporal modeling over the composite features $h_{\text{in}}^{(m)}$ to aggregate utterance-level spoofing evidence.
- **Low-Level Module (L_{Linear}):** Operating at a high execution frequency (k steps per cycle), the low-level module is instantiated with efficient linear attention operators—specifically testing GDN2 and Raven. These operators feature $O(S)$ sequence complexity and dynamic state/slot update mechanisms, enabling rapid, low-cost recurrent refinements of local acoustic artifact boundaries.

By systematically evaluating different linear sequence mixers within the low-level module (L_{GDN2} vs. L_{Raven}) alongside a high-level MHSA module, Hetero-HRM investigates

Dataset	Partition	Bonafide	Spoof	Total
19LA	Train	2,580	22,800	25,380
19LA	Dev.	2,548	22,296	24,844
19LA	Eval.	7,355	63,882	71,237
21LA	Eval.	14,816	133,360	148,176
21DF	Eval.	14,869	519,059	533,928

Table 1: Number of bonafide and spoofed utterances in the datasets used for ASVspoof 2019/2021 training, development, and evaluation.

whether operator heterogeneity across update frequencies serves as a general and effective strategy for recurrent speech deepfake detection.

Pooling and Classifier

To aggregate frame-level features $Z \in \mathbb{R}^{S \times d_{in}}$ into a fixed-dimensional utterance representation z_{pooled} , we employ Multi-Head Gated Attention Pooling (MHGAP). MHGAP combines frame-wise channel gating with multi-head temporal attention to adaptively aggregate spoofing evidence.

Given the input sequence Z , a linear projection produces the value and gate features:

$$[\mathbf{V}, \mathbf{G}] = \text{Linear}(Z) \in \mathbb{R}^{S \times (2d_{pooled})}. \quad (16)$$

The projected features are divided into K heads ($d_k = d_{pooled}/K$), yielding $\mathbf{V}_{s,k}, \mathbf{G}_{s,k} \in \mathbb{R}^{d_k}$.

For frame $s \in \{1, \dots, S\}$ and head $k \in \{1, \dots, K\}$, the gated evidence vector is computed as:

$$\mathbf{E}_{s,k} = \mathbf{V}_{s,k} \odot \text{SiLU}(\mathbf{G}_{s,k}). \quad (17)$$

In parallel, each head is associated with a learnable scoring vector $\mathbf{w}_k \in \mathbb{R}^{d_k}$. The normalized temporal attention weight $a_{s,k}$ is computed via a temperature-scaled Softmax:

$$a_{s,k} = \frac{\exp(\mathbf{V}_{s,k}^\top \mathbf{w}_k / \tau)}{\sum_{s'=1}^S \exp(\mathbf{V}_{s',k}^\top \mathbf{w}_k / \tau)}, \quad (18)$$

where $\tau > 0$ denotes the score temperature.

The representation for head k is obtained by aggregating the gated evidence over the temporal dimension:

$$\mathbf{P}_k = \sum_{s=1}^S a_{s,k} \mathbf{E}_{s,k}. \quad (19)$$

The head representations are concatenated and regularized by dropout:

$$z_{pooled} = \text{Dropout}(\text{Concat}(\mathbf{P}_1, \dots, \mathbf{P}_K)). \quad (20)$$

Finally, a linear classifier maps the pooled representation to two class logits $\ell = W_{cls} z_{pooled} \in \mathbb{R}^2$. The predicted class is obtained via:

$$\hat{y} = \arg \max_{c \in \{0,1\}} \ell_c. \quad (21)$$

SSL frontend	Operator	19LA	21LA	21DF
HuBERT Base	MHSA	6.45	8.78	18.27
	GDN2	5.52	9.73	15.15
	Raven	7.31	12.53	15.65
wav2vec 2.0 Base	MHSA	4.48	12.08	19.20
	GDN2	4.44	13.84	18.49
	Raven	4.33	9.77	17.72
WavLM Base	MHSA	6.67	11.47	15.58
	GDN2	8.83	13.00	17.35
	Raven	11.24	13.86	16.70
WavLM Base+	MHSA	7.17	14.71	17.25
	GDN2	6.14	14.88	16.07
	Raven	11.27	15.94	14.28

Table 2: EER (%) results of 6-layer baseline backbones using frozen SSL frontends. Lower is better.

Experiment Setup

Datasets and Metrics

All models are trained exclusively on the ASVspoof 2019 Logical Access (19LA) training set, with model selection performed on its development set. We evaluate all systems across three evaluation benchmarks: 19LA, ASVspoof 2021 Logical Access (21LA), and ASVspoof 2021 Deepfake (21DF). The 19LA evaluation set contains previously unseen spoofing attack configurations based on TTS and VC systems. The 21LA evaluation set assesses robustness to telephony and Voice over Internet Protocol (VoIP) transmission and codec distortions, whereas 21DF evaluates cross-domain generalization under diverse spoofing systems, source corpora, and lossy compression conditions. Dataset statistics are summarized in Table 1. Following the official ASVspoof evaluation protocols, we report the equal error rate (EER) as the performance metric.

Implementation Details

Audio signals are resampled to 16 kHz and fixed to 4 seconds (64,000 samples) via sequential repetition or cropping (random during training, center during evaluation). The downstream backbone hidden dimension is $d = 128$, and all sequence operators (MHSA, GDN2, Raven) as well as the MHGAP layer use $K = 4$ heads.

Models are trained for up to 20 epochs with a batch size of 32 and early stopping after 7 epochs. We optimize using AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$, weight decay 0.1) (Loshchilov and Hutter 2017) and Focal Loss (Lin et al. 2017) with class weights $\alpha = [0.8, 0.2]$ (prioritizing class 0, bonafide), with an initial learning rate of 1×10^{-4} governed by a cosine annealing schedule with 5% linear warmup. Training employs BF16/FP32 mixed precision. RawBoost data augmentation (Algo 4) (Tak et al. 2021a) is applied only in specific ablation setups. All models are trained with fixed random seeds on NVIDIA RTX 4090 and RTX 4080 Super GPUs.

Weight-Shared Recurrent Latent Refinement													
Passes	Operator	HuBERT Base			wav2vec 2.0 Base			WavLM Base			WavLM Base+		
		19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF
2	MHSA	9.52	13.90	19.06	4.56	10.14	20.04	9.25	13.81	16.85	7.05	15.81	29.81
	GDN2	7.02	11.60	17.61	5.38	11.27	21.96	12.12	15.44	20.49	9.57	15.10	20.24
	Raven	9.61	12.95	29.67	12.65	16.46	25.05	17.89	19.51	22.79	12.93	16.26	26.01
3	MHSA	8.01	11.01	15.97	7.47	12.27	20.62	6.86	11.84	17.80	8.52	17.05	23.58
	GDN2	7.29	12.53	19.41	4.46	9.62	21.19	10.27	13.74	20.16	9.13	13.29	21.71
	Raven	15.17	16.16	28.76	13.32	23.27	30.01	18.98	20.96	24.74	14.01	17.29	25.34

Table 3: Table 3: EER (%) of weight-shared recurrent backbones with two and three recurrent passes. All SSL frontends are frozen. The best result for each SSL frontend, recurrent setting, and evaluation set is highlighted in bold.

Homogeneous HRM													
Schedule	Operator	HuBERT Base			wav2vec 2.0 Base			WavLM Base			WavLM Base+		
		19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF
H_2L_1	MHSA	11.58	13.82	25.31	14.30	14.25	27.42	14.83	18.97	21.74	15.55	16.46	24.36
	GDN2	10.36	11.27	23.27	9.57	12.80	26.57	11.97	15.35	24.49	14.60	17.13	23.88
	Raven	14.04	14.79	25.21	13.27	15.02	24.72	13.99	17.01	24.81	14.39	16.17	27.00
H_2L_2	MHSA	13.02	15.18	22.85	8.66	12.09	24.40	13.17	17.00	29.04	17.85	17.20	36.71
	GDN2	10.24	12.51	20.28	11.03	13.45	24.70	13.47	17.04	27.72	12.68	15.09	19.92
	Raven	15.70	14.84	25.19	13.70	18.33	26.11	16.50	17.64	26.46	21.50	16.87	44.70
H_2L_3	MHSA	9.23	11.46	24.24	11.48	13.53	22.08	8.93	13.46	24.09	15.67	16.78	23.25
	GDN2	9.24	10.88	21.16	8.47	10.40	24.28	12.10	14.54	24.64	12.46	15.04	16.97
	Raven	17.93	17.66	25.21	15.53	15.91	25.01	17.49	17.64	27.31	14.86	16.36	22.01

Table 4: EER (%) of homogeneous hierarchical backbones under different recurrent schedules. All SSL frontends are frozen. The best result for each SSL frontend and evaluation set is highlighted in bold.

Experiments

Single-Pass Backbones

Table 2 presents single-pass baseline results under frozen SSL frontends. Paired with W2V2B, all backbones achieved low 19LA EERs, with Raven yielding the best 19LA EER of 4.33% (and 9.77%/17.72% on 21LA/21DF). However, Raven’s performance degraded when combined with WavLMB or WavLMB+ compared to HuB or W2V2B. These results demonstrate a clear interaction between SSL features and sequence operators: W2V2B excels in-domain (19LA), while the WavLM family shows advantages in cross-domain scenarios.

Weight-Shared Recurrent

To test whether gains stem merely from extra computation, we repeatedly executed the same module for 2 or 3 iterations without adding parameters (Table 3). For HuB, increasing MHSA loops from 2 to 3 improved performance across all benchmarks, reducing 21DF EER from 19.06% to 15.97% (a 16.21% relative drop). On W2V2B, 3-loop GDN2 achieved 4.46% and 9.62% EERs on 19LA and 21LA, outperforming its 2-loop setting. Conversely, Raven degraded with more loops. These findings indicate that increasing effective depth does not necessarily refine features, as naive repetition can introduce redundancy or error accumulation.

Homogeneous HRM

Table 4 compares homogeneous HRM configurations. On HuB, H_2L_3 MHSA and GDN2 achieved 9.23% and 10.88% EERs on 19LA and 21LA, respectively, failing to surpass single-pass baselines. On W2V2B, H_2L_3 -GDN2 achieved the best homogeneous results, yet remained weaker than certain single-pass or Looped baselines. WavLM variants were generally uncompetitive. Compared to standard models, homogeneous HRM showed more performance fluctuations; increasing low-level updates brought no consistent gains, and optimal schedules varied across frontends and test conditions.

Heterogeneous HRM

Homogeneous experiments indicate that decoupling update frequencies alone is insufficient to consistently improve detection performance. Based on this observation, we further assign MHSA to the high-level module while adopting GDN2 and Raven in the low-level module, forming two heterogeneous configurations: HALG and HALR. Table 5 presents the corresponding results. The heterogeneous design markedly outperformed homogeneous HRM under several frontend and schedule settings: for example, W2V2B- H_2L_2 -HALG achieved EERs of 7.32% and 10.89% on 19LA and 21LA, outperforming most homogeneous counterparts under the same schedule; WavLMB- H_2L_1 -HALG yielded

Heterogeneous HRM													
Schedule	System	HuBERT Base			wav2vec 2.0 Base			WavLM Base			WavLM Base+		
		19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF	19LA	21LA	21DF
H_2L_1	HALG	12.27	15.99	21.88	9.66	14.19	25.76	9.93	12.25	21.04	13.78	15.28	21.73
	HALR	13.84	15.81	22.20	12.51	17.10	25.19	14.29	16.75	25.36	16.61	17.42	26.27
H_2L_2	HALG	11.40	12.63	20.75	7.32	10.89	22.46	13.99	16.56	24.20	14.15	16.08	22.08
	HALR	13.21	15.30	29.77	9.67	12.60	23.10	11.99	15.21	23.42	16.74	17.16	23.79
H_2L_3	HALG	15.04	16.74	24.30	10.86	12.74	22.32	14.33	15.76	24.19	15.13	16.43	21.27
	HALR	20.63	20.62	31.38	9.35	16.01	19.17	12.76	14.50	22.22	15.75	26.59	17.31

Table 5: EER (%) of heterogeneous HRM configurations under different recurrent schedules.

Architecture	Schedule	19LA		21LA		21DF	
		w/o DA	w/ DA	w/o DA	w/ DA	w/o DA	w/ DA
Baseline	6-layer	1.50	4.47	7.26	5.14	12.07	9.54
Looped	2 passes	6.09	10.94	10.39	10.12	21.42	14.49
	3 passes	6.30	12.13	10.93	9.39	19.62	14.23
Homo-HRM	H_2L_1	1.74	4.80	8.15	5.31	13.53	11.04
	H_2L_2	1.54	4.55	6.78	4.89	13.34	9.94
	H_2L_3	1.56	4.65	7.89	5.04	13.84	9.59
Hetero-HRM	H_2L_1	1.65	4.43	5.85	4.70	14.78	10.29
	H_2L_2	1.36	4.52	6.72	4.93	11.64	10.69
	H_2L_3	1.87	6.21	6.72	6.20	12.70	9.69

Table 6: Ablation results under selective SSL fine-tuning and different data augmentation settings (reported in EER (%)).

relatively balanced EERs of 9.93%, 12.25%, and 21.04% across the three evaluation sets. In most setups, HALG performed better than HALR. These results demonstrate that while complementary sequence operators can alleviate redundancy from homogeneous H/L updates, their gains may depend on whether frontend representations can adapt to downstream hierarchical updates. Guided by this insight, we unfreeze the top two Transformer layers of the SSL encoder in the final ablation study.

Ablation Study of GDN2

Based on the overall performance in the frozen-frontend experiments, we selected W2V2B and GDN2 for the final ablation study, unfreezing the top two Transformer layers of the SSL encoder. Table 6 presents the ablation results across different configurations. Without data augmentation, selective fine-tuning significantly reduced the EERs of all major models: Hetero- H_2L_2 achieved EERs of 1.36%, 6.72%, and 11.64% on 19LA, 21LA, and 21DF, consistently outperforming the GDN2 baseline; Hetero- H_2L_1 achieved the lowest 21LA EER of 5.85%. This indicates that when high-level SSL representations participate in task adaptation, heterogeneous high/low-level configurations can obtain performance advantages over matched baselines across diverse evaluation conditions. In contrast, the overall performance of weight-shared recurrent models remained inferior, suggesting that such structures are less suitable as detection backbones under current settings. Data augmentation significantly improved

cross-domain performance, though at the cost of reduced accuracy on 19LA: for instance, the 19LA EER of Hetero- H_2L_1 increased from 1.65% to 4.43%, whereas its 21LA and 21DF EERs dropped from 5.85% and 14.78% to 4.70% and 10.29% (relative reductions of 19.66% and 30.38%), respectively. These results demonstrate that data augmentation enhances model robustness and cross-domain generalizability. Notably, the downstream backbone parameter counts for the GDN2 baseline and Hetero-HRM are 1.405 M and 1.252 M, respectively. Hetero-HRM reduces backbone parameters by 10.89% while achieving competitive performance under several evaluation conditions, highlighting its strong competitiveness for the SDD task.

Conclusion

This paper presents REIMU, a controlled investigation of recurrent hierarchical architectures for SSL-based speech deepfake detection. By comparing single-pass backbones, weight-shared recurrence, homogeneous HRM, and heterogeneous HRM, we disentangle the effects of recurrent computation, parameter sharing, and operator assignment. The results show that recurrence or hierarchical decomposition alone does not guarantee improved detection, whereas complementary high- and low-level operators provide a more competitive design. Moreover, the heterogeneous configurations remain competitive with fewer downstream parameters, demonstrating the potential of parameter-efficient hierarchical reasoning for SDD.

References

- Afzal, A.; Bick, A.; Xing, E. P.; Cevher, V.; and Gu, A. 2026. Raven: High-Recall Sequence Modeling with Sparse Memory Routing.
- Baevski, A.; Zhou, H.; Mohamed, A.; and Auli, M. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477*.
- Chen, S.; Wang, C.; Chen, Z.; Wu, Y.; Liu, S.; Chen, Z.; Li, J.; Kanda, N.; Yoshioka, T.; Xiao, X.; et al. 2022. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6): 1505–1518.
- El Kheir, Y.; Samih, Y.; Maharjan, S.; Polzehl, T.; and Möller, S. 2025. Comprehensive layer-wise analysis of ssl models for audio deepfake detection. In *Findings of the Association for Computational Linguistics: NAACL 2025*, 4070–4082.
- Ge, R.; Liao, Q.; and Poggio, T. 2025. Hierarchical reasoning models: Perspectives and misconceptions. *arXiv preprint arXiv:2510.00355*.
- Hatamizadeh, A.; Choi, Y.; and Kautz, J. 2026. Gated DeltaNet-2: Decoupling erase and write in linear attention. *arXiv preprint arXiv:2605.22791*.
- Hsu, W.-N.; Bolte, B.; Tsai, Y.-H. H.; Lakhotia, K.; Salakhutdinov, R.; and Mohamed, A. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29: 3451–3460.
- Hu, H.; Zhu, X.; He, T.; Guo, D.; Zhang, B.; Wang, X.; Guo, Z.; Jiang, Z.; Hao, H.; Guo, Z.; et al. 2026. Qwen3-TTS Technical Report. *arXiv preprint arXiv:2601.15621*.
- Jung, J.-w.; Heo, H.-S.; Tak, H.; Shim, H.-j.; Chung, J. S.; Lee, B.-J.; Yu, H.-J.; and Evans, N. 2022. Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks. In *ICASSP 2022-2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 6367–6371. IEEE.
- Li, M.; Ahmadiadli, Y.; and Zhang, X.-P. 2025. A survey on speech deepfake detection. *ACM Computing Surveys*, 57(7): 1–38.
- Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.
- Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Müller, N. M.; Kawa, P.; Choong, W. H.; Casanova, E.; Gölge, E.; Müller, T.; Syga, P.; Sperl, P.; and Böttinger, K. 2024. Mlaad: The multi-language audio anti-spoofing dataset. In *2024 International Joint Conference on Neural Networks (IJCNN)*, 1–7. IEEE.
- Rosello, E.; Alanís, A. G.; Gomez, A. M.; Peinado, A. M.; Harte, N.; Carson-Berndsen, J.; and Jones, G. 2023. A conformer-based classifier for variable-length utterance processing in anti-spoofing. In *Interspeech*, volume 2023, 5281–5285.
- Shazeer, N. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.
- Tak, H.; Kamble, M.; Patino, J.; Todisco, M.; and Evans, N. 2021a. Rawboost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing. *arXiv preprint arXiv:2111.04433*.
- Tak, H.; Patino, J.; Todisco, M.; Nautsch, A.; Evans, N.; and Larcher, A. 2021b. End-to-end anti-spoofing with rawnet2. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6369–6373. IEEE.
- Tak, H.; Todisco, M.; Wang, X.; Jung, J.-w.; Yamagishi, J.; and Evans, N. 2022. Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation. *arXiv preprint arXiv:2202.12233*.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, G.; Li, J.; Sun, Y.; Chen, X.; Liu, C.; Wu, Y.; Lu, M.; Song, S.; and Yadkori, Y. A. 2025. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*.
- Wang, G.; Liu, C.; Wang, C.; Zhou, C.; Sun, Y.; Wu, Y.; Zhen, S.; Scimeca, L.; and Yadkori, Y. A. 2026. HRM-Text: Efficient Pretraining Beyond Scaling. *arXiv preprint arXiv:2605.20613*.
- Wang, X.; Yamagishi, J.; Todisco, M.; Delgado, H.; Nautsch, A.; Evans, N.; Sahidullah, M.; Vestman, V.; Kinnunen, T.; Lee, K. A.; et al. 2020. ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech. *Computer Speech & Language*, 64: 101114.
- Xiao, Y.; and Das, R. K. 2025. XLSR-Mamba: A dual-column bidirectional state space model for spoofing attack detection. *IEEE Signal Processing Letters*.
- Xie, Y.; Cheng, H.; Zhou, J.; Guo, X.; Wang, T.; Liu, J.; Wang, W.; Fu, R.; Wang, X.; Huang, H.; et al. 2026a. At-add: All-type audio deepfake detection challenge evaluation plan. *arXiv preprint arXiv:2604.08184*.
- Xie, Y.; Fu, R.; Wang, X.; Wang, Z.; Cao, S.; Ma, L.; Cheng, H.; and Ye, L. 2026b. Detect all-type deepfake audio: Wavelet prompt tuning for enhanced auditory perception. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 35922–35930.
- Yamagishi, J.; Wang, X.; Todisco, M.; Sahidullah, M.; Patino, J.; Nautsch, A.; Liu, X.; Lee, K. A.; Kinnunen, T.; Evans, N.; et al. 2021. ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection. *arXiv preprint arXiv:2109.00537*.
- Yang, L.; Lee, K.; Nowak, R.; and Papailiopoulos, D. 2024. Looped transformers are better at learning learning algorithms. In *International conference on learning representations*, volume 2024, 42195–42214.
- Zhang, B.; and Sennrich, R. 2019. Root mean square layer normalization. *Advances in neural information processing systems*, 32.

Zhang, Q.; Wen, S.; and Hu, T. 2024. Audio deepfake detection with self-supervised xls-r and sls classifier. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 6765–6773.