

CORELOOP: PARAMETER-EFFICIENT CONTROLLED RECURRENT REFINEMENT FOR AUDIO DEEPPAKE DETECTION

Kunyu Feng¹, Yuxiang Wang¹, Li Wang¹, Wan Lin¹, Zhizheng Wu^{1,2}

¹The Chinese University of Hong Kong, Shenzhen ²Amphion Technology Co., Ltd.

ABSTRACT

Generalizing to unseen attacks remains challenging for audio deepfake detectors, and collecting training data covering all potential attacks is impractical. We explore recurrent refinement in an already-trained SSL-based detector without additional data or changes to its original parameters. However, directly recycling encoder outputs as inputs degrades detection in our diagnostic. We propose CoReLoop, which makes this reuse effective by adapting recurrent inputs to the frozen encoder, controlling state updates, and aligning refined outputs with the frozen classifier. By training only lightweight refinement modules and loop-specific low-rank adapters on the original data, CoReLoop enables additional refinement while preserving the detector’s original first-pass prediction. On 14 cross-domain test sets, the 24-layer model reduces pooled equal error rate (EER) from 4.85% to 3.74% with two passes, with approximately 10M trainable parameters out of 598M. To selectively apply this refinement, an optional halting head chooses the depth for each utterance, achieving 3.73% pooled EER with an average of 1.18 passes.

Index Terms— audio deepfake detection, recurrent refinement, parameter-efficient adaptation, adaptive computation

1. INTRODUCTION

Audio deepfake detection aims to distinguish genuine speech from speech generated by synthesis or voice conversion. A common detector combines a self-supervised learning (SSL) speech encoder with a classifier that predicts authenticity from the encoded representation [1]. These detectors are expected to generalize to unseen attacks whose acoustic distributions may differ from the training data [2]. As synthesis and voice conversion methods continue to evolve, collecting training data that cover all potential attacks is impractical [3]. This motivates seeking better generalization without relying on broader attack coverage during training. We investigate whether an already-trained detector can improve upon its single-pass performance on unseen attacks through additional refinement, without expanding the training data or updating its original parameters.

To enable additional refinement, we consider recurrence,

where the representation produced by one encoder pass guides the next. This feedback allows subsequent processing to build on previously extracted features, while parameter sharing increases effective depth without duplicating the encoder. Recurrent-depth language models demonstrate that this strategy can improve reasoning performance when their shared layers are trained for iterative processing [4, 5]. In our setting, however, the encoder has already been optimized for single-pass inference, and its original parameters remain frozen. Introducing recurrence therefore requires converting this existing feed-forward computation into an effective iterative refinement process.

This conversion requires reconciling the representations produced and consumed by the encoder. Its first layer was trained to receive acoustic frontend features, whereas direct recurrence supplies the final layer’s output in their place. In our diagnostic, simply feeding this output back without adaptation raises pooled EER from 4.85% at the first pass to 10.52% and 64.68% at the next two. These results show that additional encoder evaluations alone do not guarantee better detection. With the original detector frozen, effective recurrence therefore calls for mechanisms that construct suitable recurrent inputs, control state updates, and maintain compatibility with the classifier.

We propose CoReLoop (**C**ontrolled **R**efinement **L**oop), which enables controlled recurrence in a frozen detector through loop-specific LoRA [6] and lightweight state control. Only the added components are trained on the original data, while the first-pass prediction remains unchanged. An optional halting head adaptively selects the refinement depth for each utterance during inference.

While REIMU [7] studies recurrent downstream networks operating on SSL features, CoReLoop enables recurrence within the existing SSL encoder while preserving the original detector parameters and first-pass prediction.

We evaluate whether this additional computation improves detection on 14 cross-domain test sets. Two-pass refinement improves pooled EER across 3-, 6-, 12-, and 24-layer encoders. With approximately 10M trainable parameters in the 598M-parameter model, CoReLoop reduces pooled EER from 4.85% to 3.74%. Adaptive inference achieves 3.73% pooled EER at an average of 1.18 passes, using 41% fewer encoder-core evaluations than fixed two-pass inference.

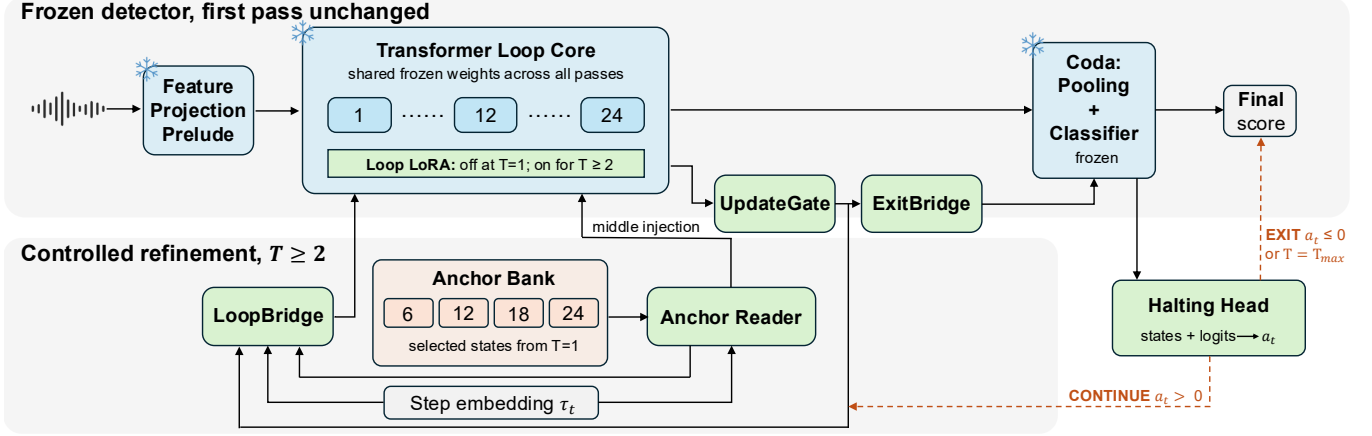


Fig. 1. Overview of CoReLoop. At $T = 1$, loop-specific modules are bypassed, yielding the frozen baseline path. For $T \geq 2$, lightweight adaptation and state-control modules refine the recurrent state, while the halting head determines whether each sample should exit or continue.

2. METHOD

2.1. Frozen detector and recurrent inputs

A trained detector comprises an acoustic frontend P_{θ_p} , an encoder core F_{θ_f} containing all retained layers, and a pooling-classification module C_{θ_c} . All original parameters remain frozen. The first pass produces frontend features $\mathbf{h}_0$ and encoder output $\mathbf{s}_1$, which the frozen classifier processes directly. All refinement modules are bypassed during the first pass, exactly preserving the original detector’s prediction.

For $t \geq 2$, a LoopBridge maps the previous endpoint $\mathbf{s}_{t-1}$ back into the core input space, using $\mathbf{h}_0$ as a fixed Prelude anchor. Conditioned on first-pass anchor context $\mathbf{e}_t$ and a step embedding τ_t , it constructs the recurrent input as

$$\begin{aligned} \mathbf{h}_t = & \mathbf{h}_0 + B_{\Delta}(\text{Norm}(\mathbf{s}_{t-1} - \mathbf{h}_0)) \\ & + B_e(\mathbf{e}_t) + B_{\tau}(\tau_t), \end{aligned} \quad (1)$$

where Norm denotes normalization, and B_{Δ} , B_e , and B_{τ} are lightweight trainable projections into the core input space. This construction conditions the recurrent input on the previous core state’s deviation from the Prelude anchor.

To provide context $\mathbf{e}_t$, we construct an anchor bank $\mathcal{A}$ from the first-pass hidden states using layer-specific LayerNorm and linear projections from 1024 to 256 dimensions. The 24-layer model uses layers $\{6, 12, 18, 24\}$. The bank is bypassed at $t = 1$ and remains fixed throughout refinement. For $t \geq 2$, an AnchorReader conditioned on $\mathbf{s}_{t-1}$ and τ_t reads across anchor layers and frames to obtain $\mathbf{e}_t$. This context is also injected after layer 12 in the 24-layer model.

2.2. Controlled recurrent refinement

Later passes activate loop-specific LoRA updates $\Delta\theta_t$ on attention projections (q/k/v/o) and FFN linear layers. Each

adapted frozen linear map receives a rank- r update with trainable factors A and B shared across passes, scaled by α/r and a learned step-specific multiplier γ_t . The shared encoder produces a candidate state, which is then blended with the previous endpoint through a learned update gate:

$$\begin{aligned} \tilde{\mathbf{s}}_t &= F_{\theta_f, \Delta\theta_t}(\mathbf{h}_t; \mathbf{e}_t), \\ \mathbf{s}_t &= \mathbf{s}_{t-1} + \mathbf{g}_t \odot (\tilde{\mathbf{s}}_t - \mathbf{s}_{t-1}). \end{aligned} \quad (2)$$

The UpdateGate predicts an utterance-level sigmoid gate $\mathbf{g}_t$ from the previous and candidate states, anchor context, and step embedding; the gate is broadcast over frames and features. For $T > 1$, an ExitBridge applies a lightweight residual projection to $\mathbf{s}_T$, conditioned on $\mathbf{s}_1$ and τ_T , to align the refined state with the frozen classifier.

2.3. Training and adaptive halting

Only LoRA and the new modules are trained, using the original data. Each batch uniformly samples $T \in \{2, 3\}$ and minimizes BCE on the terminal prediction.

We then freeze the refinement model and train a halting head H_{ψ} . Evaluating each sample at all three depths without gradients gives its per-depth BCE losses ℓ_t , computed from the classification logits. At $t \in \{1, 2\}$, H_{ψ} predicts a continue logit a_t from the current and previous endpoints and logits; at $t = 1$, $\mathbf{h}_0$ and a zero logit provide the previous inputs. Continuation is supervised when a later pass reduces the loss by more than $\delta = 0.01$:

$$c_t = \mathbb{I} \left[\min_{t < k \leq 3} \ell_k + \delta < \ell_t \right], \quad t \in \{1, 2\}. \quad (3)$$

With $p_t = \sigma(a_t)$, the soft exit probabilities are $q_1 = 1 -$

Table 1. Cross-domain EER (%) and RTF on 24-layer W2V-BERT (seed 42; 598M parameters, 10M trainable). Pooled EER (primary metric) uses a pooled threshold; individual EERs use test-set-specific thresholds. Macro averages EER over 14 test sets. RTF is inference time divided by audio duration. Lower is better; best non-oracle results are bolded. Random, Halting, and Oracle are CoReLoop policies; Oracle denotes a non-deployable ideal halting policy.

Model	RTF 10^{-2}	Pooled	Macro	ITW	ASVspoof				FoR	CF	ADD				DFADD	LSV	SONAR
					19	21LA	21DF	24			22T1	22T3	23R1	23R2			
W2V-BERT ($T = 1$)	1.71	4.85	2.86	1.88	0.34	2.73	2.69	10.43	2.15	0.22	12.45	0.90	1.63	4.51	0.00	0.13	0.00
LoRA ($T = 1$)	2.44	4.24	2.81	2.20	0.87	3.27	2.15	9.28	2.15	0.45	10.88	0.89	1.31	3.99	0.00	0.63	1.20
CoReLoop ($T = 2$)	3.63	3.74	2.47	1.66	0.63	2.26	2.18	9.53	0.86	0.21	10.55	0.81	1.36	4.20	0.00	0.38	0.00
CoReLoop ($T = 3$)	5.47	3.69	2.49	1.24	0.62	2.39	2.18	9.56	1.28	0.22	10.60	0.92	1.25	4.20	0.00	0.44	0.00
Random ($\bar{T} = 1.75$)	3.31	4.03	2.75	2.05	0.77	2.72	2.04	9.72	1.73	0.35	11.53	1.34	1.69	4.14	0.00	0.44	0.00
Halting ($\bar{T} = 1.18$)	2.02	3.73	2.55	1.02	0.60	2.48	2.15	9.86	1.28	0.58	10.90	0.92	1.25	4.20	0.00	0.44	0.00
Oracle ($\bar{T} = 1.08$)	–	3.39	2.01	1.02	0.34	1.74	1.78	8.77	0.42	0.16	8.28	0.63	1.18	3.81	0.00	0.00	0.00

$p_1, q_2 = p_1(1 - p_2)$, and $q_3 = p_1p_2$. We minimize

$$\mathcal{L}_{\text{halt}} = \sum_{t=1}^2 \text{BCE}_{w_t^+}(a_t, c_t) + 0.5 \mathbb{E} \left[\sum_{t=1}^3 q_t \ell_t \right]. \quad (4)$$

Here, w_t^+ is an inverse-frequency positive-class weight capped at 20. The second term minimizes classification loss under the induced exit distribution, helping mitigate the mismatch between all-depth supervision during training and sequential halting at inference. At inference, $a_t > 0$ continues refinement and $a_t \leq 0$ exits; the third pass always exits, giving $T_{\text{max}} = 3$.

3. EXPERIMENTS

3.1. Experimental setup

Data and metrics. Broadly following the training-data selection of Tefic Audio [8], all models use official training splits from 14 public speech datasets, including ASVspoof 2019 LA, ADD 2023, DFADD, and AISHELL-3 [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]; no evaluation or test utterances are used for training. We follow Speech DF Arena [23], reporting pooled EER (primary) and unweighted macro-average EER over its 14 public test sets. RTF is inference time divided by input duration, computed as the input sample count divided by 16 kHz (4 s per sample). Timing uses a single NVIDIA V100 GPU with batch size 1 and FP16.

Models and optimization. We use frozen encoders with 3, 6, 12, and 24 Transformer layers, with LoRA rank $r = 8$, scaling $\alpha = 16$, and dropout 0.05. Loop-specific parameters are trained for 10 epochs using AdamW with learning rate 10^{-4} and batch size 32.

Comparisons. The base detector is trained for 30 epochs with BCE and then frozen. All adapted models start from this checkpoint. We compare CoReLoop with the frozen detector, single-pass LoRA [24], and naive-recurrence controls. Single-pass LoRA uses the same adapter placement, rank, and training schedule as CoReLoop, without recurrent-state modules. Random halting uses the trained refinement model but exits independently with probability 0.5 at each of the first two passes and always at the third. A non-deployable oracle represents an ideal halting head with access to ground-truth labels and future predictions. It uses the current pass’s pooled-EER threshold to assess both current and future predictions, continuing only when a later pass can correct the current error; otherwise, it exits.

3.2. Main cross-domain results

Table 1 shows that CoReLoop at $T = 2$ reduces pooled EER from 4.85% to 3.74%, a 22.9% relative reduction over the frozen baseline, with improvements on 10 of the 14 test sets. It also outperforms single-pass LoRA (4.24% pooled EER), indicating that LoRA adaptation alone does not account for the full gain. A third pass provides only marginal pooled improvement while slightly worsening macro EER, suggesting diminishing returns from further refinement.

Across three random seeds (42, 0, 1) using the same frozen checkpoint, fixed $T=2$, fixed $T=3$, and adaptive halting obtain pooled EERs of $3.81 \pm 0.07\%$, $3.78 \pm 0.08\%$, and $3.79 \pm 0.06\%$ (mean $\pm$ std), respectively, confirming stable performance across random seeds.

Figure 2 examines prediction changes across the pooled evaluation and individual test sets. For each transition, both predictions use the source-depth pooled-EER threshold, shared across all test sets. The pooled results show a favorable balance between corrections and regressions, but this balance varies across domains: refinement can correct



Fig. 2. Sample-level error corrections and regressions across successive refinement passes of the 24-layer model on pooled evaluation data and 14 individual test sets.

existing errors but also overturn correct predictions. This heterogeneity motivates sample-wise depth selection.

CoReLoop-Halting achieves a 23.1% relative reduction in pooled EER with only 18.1% higher RTF than the frozen baseline. It matches fixed $T = 2$ performance while reducing RTF by 44.5% and Transformer-core evaluations by 41%, showing that adaptive halting preserves detection gains with substantially lower inference cost. Its advantage over random halting in both pooled EER and RTF further supports sample-dependent depth selection. The ground-truth-dependent oracle achieves 3.39% pooled EER, suggesting room for better depth selection.

3.3. What makes recurrent refinement effective?

LoopBridge is the most critical ablated component, aligning recurrent inputs with the shared core. Removing it raises pooled EER from 3.74% to 4.56% at $T = 2$, near naive recurrence with loop LoRA (4.57%). ExitBridge aligns outputs with the frozen classifier; its removal raises pooled EER to 3.95%. Removing loop LoRA increases pooled/macro EER by 0.38/0.33 percentage points. These results support complementary alignment and encoder adaptation.

A minimal variant retaining only these three components achieves 3.89% and 3.91% pooled EER at $T = 2$ and $T = 3$, respectively, capturing most of the improvement over the frozen baseline. The full model further reduces pooled EER

Table 2. Mechanism controls and component ablations. Each entry reports pooled/macro EER (%). Lower is better. Minimal retains only LoopBridge, ExitBridge, and loop LoRA.

Variant	$T = 2$	$T = 3$
Zero-shot naive recurrence	10.52/8.02	64.68/58.19
Naive recurrence w/ loop LoRA	4.57/3.47	4.57/3.54
CoReLoop w/o LoopBridge	4.56/3.35	4.45/3.16
CoReLoop w/o ExitBridge	3.95/2.49	3.79/2.44
CoReLoop w/o loop LoRA	4.12/2.80	4.08/2.73
CoReLoop w/o anchor bank	3.90/2.46	3.85/2.41
CoReLoop w/o UpdateGate	3.78/2.58	3.79/2.66
CoReLoop w/o middle inject.	3.87/2.56	3.85/2.49
CoReLoop, 8 anchors	3.96/2.42	3.94/2.45
CoReLoop (minimal)	3.89/2.52	3.91/2.58
CoReLoop (full)	3.74/2.47	3.69/2.49

Table 3. Scaling across different frozen encoder depths. Entries report pooled/macro EER (%). $\bar{T}$ denotes the average number of refinement passes used during adaptive inference.

Layers	Baseline	Fixed $T = 2$	Fixed $T = 3$	Adaptive	$\bar{T}$
3	7.74/5.48	6.23/4.42	6.26/4.36	6.25/4.40	1.36
6	4.86/3.05	4.25/2.79	4.19/2.78	4.38/2.96	1.15
12	4.99/3.17	4.19/2.72	4.17/2.74	4.23/3.11	1.21
24	4.85/2.86	3.74/2.47	3.69/2.49	3.73/2.55	1.18

by 0.15 and 0.22 percentage points, demonstrating the collective benefit of the auxiliary reference and state-control mechanisms. The larger gain at $T = 3$ suggests that these mechanisms help sustain the effectiveness of additional refinement.

3.4. Backbone scaling and adaptive computation

Table 3 evaluates CoReLoop across frozen encoders with 3–24 layers. Fixed $T = 2$ improves pooled and macro EER over every baseline, reducing pooled EER by 0.61–1.51 percentage points, with small changes at $T = 3$. Adaptive inference approaches fixed $T = 2$ pooled EER with 32–42.5% fewer Transformer-core evaluations. These results support recurrent refinement across encoder depths and efficient adaptive inference without updating the original detector parameters.

4. CONCLUSION

We presented CoReLoop, which enables recurrent refinement in a frozen audio deepfake detector through loop-specific LoRA and lightweight state control. Across 14 cross-domain test sets, one additional pass captures most of the gain, while adaptive halting achieves comparable detection performance with substantially fewer shared encoder-core evaluations.

5. REFERENCES

- [1] 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 *The Speaker and Language Recognition Workshop (Odyssey 2022)*, 2022, pp. 112–119.
- [2] Nicolas M. Müller, Nicholas Evans, Hemlata Tak, Philip Sperl, and Konstantin Böttinger, “Harder or Different? Understanding Generalization of Audio Deepfake Detection,” in *Interspeech 2024*, 2024, pp. 2705–2709.
- [3] You Zhang, Fei Jiang, and Zhiyao Duan, “One-class learning towards synthetic voice spoofing detection,” *IEEE Signal Processing Letters*, vol. 28, pp. 937–941, 2021.
- [4] Jonas Geiping et al., “Scaling up test-time compute with latent reasoning: A recurrent depth approach,” 2025.
- [5] Yuxiang Wang, Kunyu Feng, Yingda Shen, Haoning Xu, Junyu Wang, and Zhizheng Wu, “Recurtrace: Adaptive latent reasoning with loop-time memory,” *arXiv preprint arXiv:2609.03379*, 2026.
- [6] 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 *International Conference on Learning Representations*, 2022.
- [7] Kwok-Ho Ng, Tingting Song, Bingwen Feng, and Peiya Li, “Reimu: Efficient heterogeneous hierarchical reasoning for ssl-based speech deepfake detection,” 2026.
- [8] Wan Lin, Li Wang, Jindong Wang, Kunyu Feng, and Zhizheng Wu, “Teffic-audio: Tell fact from fiction,” 2026.
- [9] Jiawei Du et al., “Dfadd: The diffusion and flow-matching based audio deepfake dataset,” 2024.
- [10] Guoguo Chen et al., “Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio,” in *Proc. Interspeech 2021*, 2021.
- [11] Wen Huang, Yanmei Gu, Zhiming Wang, Huijia Zhu, and Yanmin Qian, “Speechfake: A large-scale multilingual speech deepfake dataset incorporating cutting-edge generation methods,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 9985–9998.
- [12] Jee weon Jung et al., “Spoofceleb: Speech deepfake detection and sasv in the wild,” 2025.
- [13] Lantian Li et al., “Cn-celeb: multi-genre speaker recognition,” 2021.
- [14] Nicolas M. Müller et al., “Mlaad: The multi-language audio anti-spoofing dataset,” 2026.
- [15] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.
- [16] Ricardo Reimao and Vassilios Tzerpos, “For: A dataset for synthetic speech detection,” in *SpeD*, 2019, pp. 1–10.
- [17] Yao Shi, Hui Bu, Xin Xu, Shaoji Zhang, and Ming Li, “Aishell-3: A multi-speaker mandarin tts corpus and the baselines,” 2021.
- [18] Xin Wang et al., “Asvspoof 5: Crowdsourced speech data, deepfakes, and adversarial attacks at scale,” 2024.
- [19] Xin Wang et al., “Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech,” 2020.
- [20] Yuankun Xie et al., “The codecfake dataset and countermeasures for the universally detection of deepfake audio,” 2024.
- [21] Jiangyan Yi et al., “Add 2023: the second audio deepfake detection challenge,” 2023.
- [22] Jiangyan Yi et al., “Add 2022: the first audio deep synthesis detection challenge,” 2024.
- [23] Sandipana Dowerah, Atharva Kulkarni, Ajinkya Kulkarni, Hoan My Tran, Joonas Kalda, Artem Fedorchenko, Benoit Fauve, Damien Lolive, Tanel Alumäe, and Mathew Magimai.-Doss, “Speech df arena: A leaderboard for speech deepfake detection models,” *IEEE Open Journal of Signal Processing*, pp. 1–9, 2026.
- [24] Janne Laakkonen, Ivan Kukanov, and Ville Hautamäki, “Generalizable speech deepfake detection via meta-learned lora,” in *ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2026, pp. 19092–19096.