BASS: BENCHMARKING AI-SYNTHETIC SONG DETECTION UNDER EVOLVING GENERATORS AND REAL-WORLD CORRUPTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Rapid updates to AI music generators challenge synthetic song detectors trained on fixed collections, while common audio processing can further degrade their performance. We introduce BASS (**B**enchmarking **AI**-**S**ynthetic **S**ong Detection under Evolving Generators and Real-World Corruptions), a benchmark containing 47,332 synthetic songs from 12 versions of Suno and Mureka, together with rich generation metadata and 23 audio corruptions for robustness evaluation. We further propose CARLA (**C**orruption-**A**ware **R**eal-Density **L**earning and **A**daptation), a two-stage framework that learns a corruption-aware real-music density from human songs and adapts to each emerging generator through a lightweight LoRA adapter trained with at most 100 synthetic support songs. Experiments on BASS show that CARLA achieves state-of-the-art performance while remaining robustness compared with basline models. These results highlight the complementary importance of adapting to new generators and modeling robustness to audio corruption. Code and data are available at `https://anonymous.4open.science/r/BASS-DAF5/`.

1 INTRODUCTION

Recent advances in generative AI music have enabled commercial platforms such as Suno[1] and Mureka[2] to produce increasingly realistic full-length songs. At the same time, AI-generated music is rapidly entering large-scale distribution channels. Deezer reported that fully AI-generated tracks exceeded 50% of daily new music uploads at peak periods, with nearly 90,000 such tracks delivered per day Deezer (2026). This rapid growth raises practical concerns regarding content authenticity, copyright protection, transparency, and fair compensation for human creators. As synthetic songs become increasingly difficult to distinguish from human songs, reliable *synthetic song detection* (SSD) has become increasingly important for maintaining trust in digital music ecosystems.

Despite this importance, SSD remains challenging under rapidly evolving commercial generators. Existing benchmarks such as SONICS Rahman et al. (2025) and Melody or Machine (MoM) Batra et al. (2025) substantially expanded the scale of AI-generated music datasets. However, such datasets inevitably represent fixed snapshots of rapidly evolving generators, and detection performance on earlier generator versions can become increasingly saturated, providing limited evidence about newly released, higher-fidelity models. More recent studies have broadened generator coverage (Vila et al., 2025; Go & Kim, 2026), but they still provide limited support for evaluating rapid version-by-version generator evolution together with fine-grained musical-content variation. More importantly, continuously collecting large-scale synthetic corpora and retraining a complete detector for every newly released generator is costly and quickly becomes outdated. Although existing methods trained on earlier generators exhibit a degree of zero-shot generalization to unseen generators, cross-generator evaluations consistently reveal substantial performance degradation when the generation source changes (Vila et al., 2025; Han et al., 2026b;a). **This motivates a practical question: under a limited training-data budget, can a detector efficiently adapt to emerging generators while preserving reliable detection across generators already represented by the system?**

[1] `https://suno.com/`
[2] `https://www.mureka.ai/`

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.

Table 1: Performance of detectors with their original weights on SONICS and BASS.

Model	SONICS				BASS (ours)			
	EER ↓ (%)	F1 ↑	Sens. ↑	Spec. ↑	EER ↓ (%)	F1 ↑	Sens. ↑	Spec. ↑
SpecTTTra-α (Rahman et al., 2025)	2.66	0.97	0.89	0.99	45.81	0.53	0.94	0.48
MusicDET (Han et al., 2026a)	2.89	0.86	0.97	0.98	38.09	0.60	1.00	0.00
Class-Conditional MusicDET (Han et al., 2026a)	0.00	1.00	1.00	1.00	49.67	0.10	0.39	0.61

Robustness to audio corruptions is another important requirement for SSD. Recording introduces acoustic distortions, and distribution platforms introduce resampling, quantization, and codec compression. Prior studies have shown that such transformations can substantially degrade audio-deepfake and AI-music detectors (Zhang et al., 2025; Li et al., 2026; Shi et al., 2025; Wu et al., 2025; Vila et al., 2025; López-Ayala et al., 2026). Normalizing flows provide tractable density estimation through a sequence of invertible transformations, enabling likelihood-based modeling of complex data distributions (Rezende & Mohamed, 2015; Papamakarios et al., 2021). We leverage this property to model the distribution of real-song representations and extend it to corruption-aware learning using corrupted *real* music. By encouraging clean and degraded human songs to remain compatible with the same learned real-music distribution, the model can acquire robustness to common audio corruptions without requiring large-scale corrupted synthetic training sets for every generator.

To address these limitations, we introduce **BASS** (**B**enchmarking **A**I-**S**ynthetic **S**ong Detection under Evolving Generators and Audio Corruptions), a large-scale benchmark designed around generator evolution, musical diversity, and real-world robustness. Following the artist-aware partitioning principle of MTG-Jamendo Bogdanov et al. (2019), which separates artist identities across data partitions while retaining broad musical-tag coverage, BASS emphasizes both identity-independent and musical diversity. We collect 47,332 synthetic songs from Suno and Mureka, totaling approximately 2,774 hours and covering 12 platform/version collection categories together with diverse musical metadata. Synthetic songs are partitioned through creator- and content-aware grouping so that related songs do not cross training, validation, and test sets. BASS further extends transformations studied in prior audio-deepfake and AI-music research (Zhang et al., 2025; Li et al., 2026; Shi et al., 2025; Wu et al., 2025; Vila et al., 2025; Dugelay et al., 2026) into **23 fixed corruption conditions**, representing common disturbances introduced by recording and playback environments, user editing and post-processing, and platform distribution pipelines. Together with paired clean–corrupted evaluation views, BASS enables controlled analysis across generator versions, musical characteristics, and deployment conditions.

We further propose **CARLA** (**C**orruption-**A**ware **R**eal-density **L**earning and **A**daptation), a two-stage framework designed for evolving generators and corrupted audio. In Stage 1, CARLA learns a corruption-aware real-music density from clean and corrupted human songs using music-aware representations based normalizing-flow density estimation. In Stage 2, each emerging generator is incorporated using only **100 synthetic songs** through lightweight LoRA adaptation Hu et al. (2021), forming an expandable adapter bank that supports generator-identity-free inference. CARLA is therefore designed to adapt efficiently to successive generator updates under limited supervision while preserving the corruption robustness learned from real music.

Our contributions are summarized as follows:

- We introduce **BASS**, a version-aware benchmark containing 47,332 synthetic songs from Suno and Mureka, rich generation and musical metadata, creator-aware partitioning, and 23 audio corruption conditions grounded in real-world processing scenarios.
- We propose **CARLA**, a two-stage framework that combines corruption-aware real-density learning with lightweight generator-specific LoRA adaptation and an expandable adapter bank, using only 100 synthetic songs for each emerging generator.
- We We demonstrate that CARLA achieves **State-of-the-Art** performance than competing baselines across BASS's cross-generator, musical-style, and audio-corruptions to evaluate adaptation to evolving generators and robustness under real-world audio transformations.

Table 2: Comparison of synthetic-song datasets.

Dataset	Full-song fake	Text lyrics	Creator metadata	Detailed music tags	User prompts	Open fake audio	Open real audio	Corruptions	Avg. length (sec.)	# Generators / versions	# Speakers / creators	# Real songs	# Fake songs	# Total songs	# Total hours
FSD (Xie et al., 2024)									216.00	5	60	200	450	650	26
SingFake (Zang et al., 2024b)									13.75		40	634	671	1,305	58
CtrSVDD (Zang et al., 2024a)						✓	✓		4.87	14	164	32,312	188,486	220,798	307
SONICS (Rahman et al., 2025)	✓	✓				✓			176.03	5	9,096+	48,090	49,074	97,164	4,751
MoM (Batra et al., 2025)	✓	✓				✓			195.99	9		65,475	64,960	130,435	6,665
BASS (ours)	✓	✓	✓	✓	✓		✓	✓	**228.83**	**12**	**22,343**	**55,525**	**47,332**	**102,857**	**6,538**

2 Related Work

2.1 Synthetic Song Detection

Spectral-artifact-based Afchar et al. (2025a) methods (Rahman et al., 2025; Li et al., 2024b; Afchar et al., 2025b; Kim & Go, 2025; Oh, 2026) exploit characteristic Fourier-domain traces of synthetic music, but such cues can be sensitive to generator shifts. Music-aware methods such as CLAM Batra et al. (2025) and SOFIA Han et al. (2026b) improve cross-generator generalization and robustness by modeling higher-level musical characteristics, yet performance still degrades substantially on newer, higher-fidelity generators. MusicDET Han et al. (2026a) instead formulates SSD as real-only zero-shot detection using frequency-guided normalizing flows (Dinh et al., 2016; Kingma & Dhariwal, 2018), yet our evaluation on newer commercial generators shows that zero-shot transfer can degrade substantially. CARLA therefore shifts the focus from indefinite zero-shot generalization to *few-shot adaptation*.

2.2 Audio Corruption Robustness

Audio Corruptions arising from recording, processing, communication, and distribution can substantially degrade deepfake detectors. Recent robustness studies evaluate broad corruption families including environmental noise, filtering, temporal and pitch transformations, resampling, quantization, compression, and communication-channel effects (Zhang et al., 2025; Li et al., 2026; Shi et al., 2025; Wu et al., 2025). Similar weaknesses have been observed directly in AI-music detection under filtering, resampling, broadcast conditions, codecs, and speed/pitch modification Vila et al. (2025); López-Ayala et al. (2026); Oh (2026); Dugelay et al. (2026). BASS consolidates these findings into a unified five-group, 23-condition corruption benchmark for full-song evaluation. CARLA learns corruption robustness from augmented real music so that clean and corrupted human songs share the same real-density prior.

2.3 Few-shot Adaptation and Parameter-efficient Tuning

LoRA Hu et al. (2021) enables lightweight adaptation through low-rank parameter updates, while adapter-based methods demonstrate that multiple specialized modules can coexist around a shared frozen representation Pfeiffer et al. (2021). Few-shot adaptation has also begun to appear in SSD Han et al. (2026b). CARLA applies this paradigm specifically to evolving commercial music generators: each generator receives an independent LoRA adapter learned from a small support set, previously learned adapters remain unchanged.

3 METHOD

3.1 BASS DATASET CONSTRUCTION

3.1.1 COLLECTION, METADATA, AND PARTITIONING

BASS contains 47,332 collected synthetic songs: 21,611 from Suno and 25,721 from Mureka, totaling approximately 2,774 hours. We retain 12 platform/version categories in the collection. The category `mureka:early` is reserved as a development-only validation category for hyperparameter tuning; the 11 identified categories define cross-generator evaluation (Appendix A).

Figure 1 summarizes the collection pipeline. We retrieve existing platform songs, Suno records originate from catalog search and feeds, while Mureka records originate from official, creator, and listing pages. Version coverage, collection budgets, and available style information organize collection. Each record links audio to its source URL, track and creator identifiers, platform/version, duration, and available prompts and musical tags.

To limit shared-content leakage, we form connected groups of synthetic records that share a nonempty creator identifier. Each group is assigned intact to a partition, with approximately 60/20/20 train/validation/test targets within each generator category. Group integrity takes precedence over exact proportions. Human reference songs use the official artist-disjoint split-0 of MTG-Jamendo (Bogdanov et al., 2019). Alignment consists of a common schema and audits of genre and duration distributions, rather than one-to-one semantic pairing with synthetic songs. Detailed counts and audit limitations appear in Appendix A.

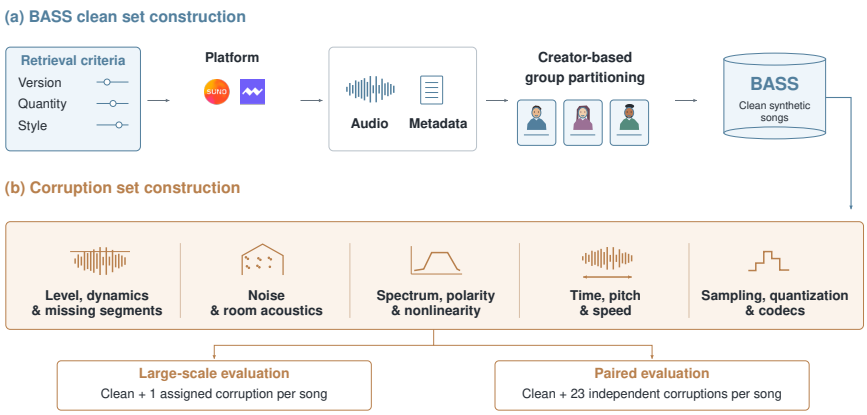


Figure 1: **BASS construction.** Synthetic songs are retrieved with version-, quantity-, and style-based criteria, then partitioned by creator groups to form the clean BASS set. Twenty-three corruptions from five families generate two evaluation views: large-scale evaluation and paired evaluation.

3.1.2 CORRUPTION EVALUATION VIEWS

Grounded in corruption mechanisms commonly considered in audio-deepfake and AI-music robustness studies (Zhang et al., 2025; Li et al., 2026; Wu et al., 2025; Shi et al., 2025; Vila et al., 2025; Dugelay et al., 2026), we define 23 conditions in five full-song-oriented families: (i) level, dynamics, and missing segments; (ii) noise and room acoustics; (iii) spectrum, polarity, and nonlinearity; (iv) time, pitch, and speed; and (v) sampling, quantization, and codecs.Clean audio serves as the uncorrupted control. Each operation is applied independently to the full song only after partitioning, using fixed parameters and resources to enables reproducible evaluation.

Two complementary evaluation set are constructed. The *assigned* setting pairs every retained track with one deterministically assigned corruption. The *paired* setting expands a stratified diagnostic subset into clean plus all 23 independently corrupted views; operations are never chained in this setting. Manifests specify parameters, and resources for on-demand evaluation. The 22-operation

training pool excludes AAC, which is held out for evaluation. Appendix B specifies all fixed conditions and distinguishes evaluation processing from online training augmentation.

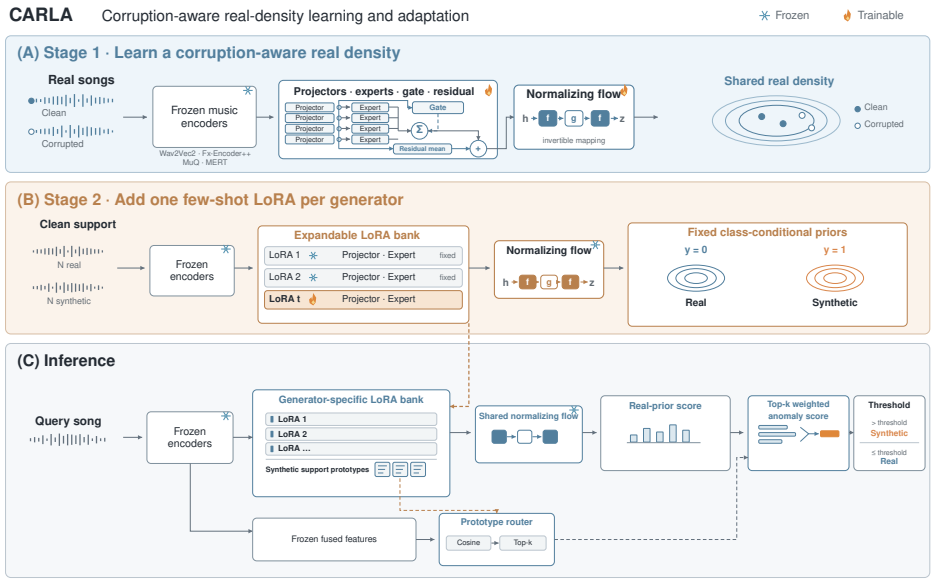


Figure 2: **CARLA overview.** With frozen music encoders, Stage 1 jointly trains the projectors, experts, gating module, and normalizing flow to learn a shared real-music density from clean and corrupted real songs (A). Stage 2 introduces one few-shot LoRA adapter per generator in the projectors and experts (B). During inference, the router selects the top-k adapters according to prototypes and combines their calibrated real-prior scores into the final real score (C).

3.2 CARLA: Real-Density Learning and Adaptation

Problem setup. For a song x with authenticity label $y \in \{0, 1\}$, where 0 denotes real and 1 synthetic, we seek a detector $\mathcal{C}$ that predicts $\hat{y} = \mathcal{C}(T(x))$. Here T is either the identity (clean audio) or an audio corruption, and $\hat{y} \in \{0, 1\}$ is the predicted label. The detector accommodate evolving generators and corrupted recordings. Below, x denotes the input song.

3.2.1 Model Architecture

Music representation. Our SOFIA-inspired network (Han et al., 2026b) combines four frozen encoders E_k, indexed by $k \in \{1, 2, 3, 4\}$, for vocal, audio-effect, and global music cues; the branches use Fx-Encoder++, MuQ, MERT, and Wav2Vec2 representations (Yeh et al., 2025; Zhu et al., 2025; Li et al., 2024a; Baevski et al., 2020). A projector P_k maps each normalized embedding to a common feature space. An expert ψ_k transforms each projected feature h_k, and a softmax gate G produces fusion weights a_k:

$$h_k = P_k(\mathrm{norm}(E_k(x))), \qquad a = G([h_1; \ldots; h_4]),$$
$$h = H_\phi(x) = \sum_{k=1}^{4} a_k \psi_k(h_k) + \frac{1}{4} \sum_{k=1}^{4} h_k. \tag{1}$$

Here H_ϕ produces the fused vector h, norm denotes Euclidean normalization, and brackets denote concatenation. The gate weights satisfy $a_k \geq 0$ and $\sum_k a_k = 1$. Stage 1 trains representation parameters ϕ comprise P_k, ψ_k, and G, whereas Stage 2 adds low-rank updates only to P_k and ψ_k. Appendix C specifies their construction.

Normalizing flow. An invertible flow F_θ, with parameters θ, maps h to a latent vector $z = F_\theta(h)$. Following the class-conditional formulation of MusicDET (Han et al., 2026a), with normalizing-flow density estimation grounded in RealNVP and Glow (Dinh et al., 2016; Kingma & Dhariwal,

2018), we associate each label y with a fixed Gaussian prior $\mathcal{N}(z; \mu_y, I)$: μ_y is its class center and I the identity covariance. The conditional density of the music vector is

$$\log p_\theta(h \mid y) = \log \mathcal{N}(F_\theta(h); \mu_y, I) + \log |\det J_{F_\theta}(h)| . \tag{2}$$

Here $p_\theta(h \mid y)$ is the representation-space density and $J_{F_\theta}(h)$ is the flow Jacobian with respect to h. Its log-determinant accounts for the flow's volume change. We write the negative log-likelihood as $\ell(x, y) = -\log p_\theta(H_\phi(x) \mid y)$; smaller ℓ means greater compatibility with the specified class prior.

Detection. Although adaptation uses both class priors, inference scores every input under the real prior. For a representation path H (the shared or an adapted network), define

$$s_H(x) = \log p_\theta(H(x) \mid y = 0). \tag{3}$$

A high score indicates compatibility with real music; a low score indicates synthetic music. Section 3.2.3 applies this rule to the adapter bank and combines its calibrated scores.

3.2.2 Two-Stage Training

Stage 1: corruption-aware real-density learning. We learn real density using clean songs and online corrupted views. Following the audio RandAugment strategy (Cubuk et al., 2020) (Cubuk et al., 2020), we sample T by choosing one operation uniformly from the training pool and drawing its parameters from operation-specific ranges. The operation is applied with a fixed probability; otherwise T is the identity. For a clean real song x, we minimize

$$\mathcal{L}_{\text{real}}(\phi, \theta) = \mathbb{E}_{x,T}[\ell(x, 0) + \lambda \ell(T(x), 0)] . \tag{4}$$

The expectation is over real training songs and sampled transformations; λ weights the corrupted-view loss. We update the projectors P_k, experts ψ_k, gate G, and flow F_θ, keeping the music encoders fixed. The shared real prior encourages high likelihood for both views, without pointwise feature alignment. Let H_0 denote the frozen Stage 1 fusion mapping without any Stage 2 LoRA adapter; its flow is reused in Stage 2. Sampling details appear in Appendix B.

Stage 2: few-shot generator adaptation. To capture a new generator g, we introduce an independent low-rank adaptation (LoRA) module (Hu et al., 2021) with parameters δ_g in the projectors and experts of H_0. Denote the adapted representation by H_g. Its conditional negative log-likelihood is $\ell_g(x, y) = -\log p_\theta(H_g(x) \mid y)$, evaluated through the frozen Stage 1 flow. Given a clean support set $\mathcal{S}_g$ containing N real and N synthetic songs, adaptation minimizes

$$\mathcal{L}_g(\delta_g) = \frac{1}{2N} \sum_{(x,y)\in\mathcal{S}_g} \ell_g(x, y). \tag{5}$$

Only δ_g is updated. Real support maintains compatibility with the real prior, while synthetic support directs generator-specific features toward the synthetic prior. Each generator contributes its own adapter to the bank, allowing adaptation to new generators without overwriting earlier adapters. Appendix D provides the algorithm and training configuration.

3.2.3 Generator-Identity-Free Bank Inference

Each adapter g has a prototype c_g: the normalized mean of its synthetic support features computed with H_0. The router compares the test feature to these prototypes, then assigns weights to the most similar adapters:

$$d_g(x) = \cos(H_0(x), c_g), \qquad w_g(x) = \frac{\exp(d_g(x)/\tau)}{\sum_{j\in\mathcal{I}(x)} \exp(d_j(x)/\tau)}. \tag{6}$$

Here cos is cosine similarity, d_g the similarity score, and $\tau > 0$ the routing temperature. Weights are defined for $g \in \mathcal{I}(x)$. This set contains up to K adapters with the largest similarities, where K is the routing budget and j indexes selected adapters.

Applying Eq. 3 to H_g gives $s_g(x) = s_{H_g}(x) = -\ell_g(x, 0)$. We combine the support-calibrated scores $\widetilde{s}_g$ as $S(x) = \sum_{g\in\mathcal{I}(x)} w_g(x)\widetilde{s}_g(x)$ and predict $\hat{y} = \mathbf{1}[S(x) < t_{\text{dev}}]$. Here S is the aggregate score, t_{dev} a development-selected threshold, and $\mathbf{1}[\cdot]$ the indicator function. Appendix E specifies the prototypes and calibration. Routing determines the adapter aggregation weights but does not update the LoRA parameters. All adapter scores are computed under the real prior.

4 EXPERIMENTS AND RESULTS

4.1 EXPERIMENTAL SETUP

We designate the development-only Mureka category as `mureka:early`: it supplies synthetic data for baseline pretraining and development data for CARLA hyperparameter and threshold selection. The other 11 generator version groups are the few-shot adaptation targets. For each target, support songs are randomly drawn from its training partition. CARLA uses $N = 100$ synthetic and $N = 100$ real support songs per adapter. Our baselines comprise RawNet3 (Jung et al., 2022), SpecTTTra-$\alpha/\beta/\gamma$ (Rahman et al., 2025), MusicDET and its class-conditional variant (Han et al., 2026a), and robust-deepfake-detector (López-Ayala et al., 2026). Except for the robust detector, which follows its own non-few-shot protocol, all baselines receive the same 11 generator categories and the same per-category training-sample budget as CARLA; model-specific inputs and optimization follow their respective methods.

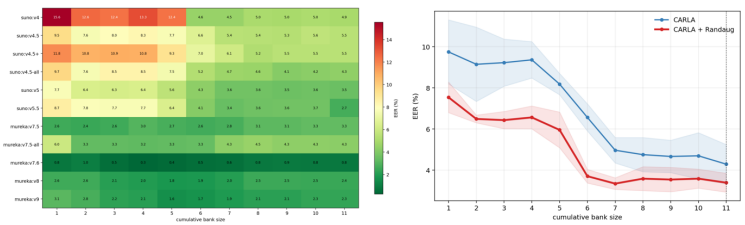
We report equal error rate (EER) as main results. With synthetic songs as the positive class, $F1_{\mathrm{AI}}$ is the harmonic mean of synthetic precision and recall, sensitivity is synthetic recall, and specificity is the true-negative rate for real music.

4.2 CROSS-GENERATOR EVALUATION

Table 3: Clean EER (%)↓ by generator. Avg. is the unweighted mean over the 11 identified BASS groups; the lower block adds Stage-1 RandAug. Dashes mark unavailable scores.

Method	Few-Shot	SUNO						MUREKA					Avg.
		v4	v4.5	v4.5+	v4.5-all	v5	v5.5	v7.5	v7.5-all	v7.6	v8	v9	
Without + RandAug													
RawNet3	✓	14.25	12.02	19.09	15.29	12.08	3.32	7.27	13.64	6.98	9.02	9.30	11.11
SpecTTTra-α	✓	**8.69**	8.53	10.52	7.81	7.41	**2.50**	7.79	9.09	6.83	7.64	7.50	7.66
SpecTTTra-β	✓	13.88	**7.00**	8.18	9.45	6.24	2.83	8.52	9.09	4.43	6.58	7.20	7.58
SpecTTTra-γ	✓	10.28	10.20	11.36	12.38	9.22	2.68	9.78	9.09	7.45	8.73	9.00	9.11
robust_deepfake_detector	×	19.83	18.68	17.51	19.38	17.31	14.07	49.80	39.35	50.45	50.54	47.95	31.35
MusicDET	✓	33.06	44.88	28.55	38.83	27.29	22.98	53.12	55.68	42.87	44.95	40.04	39.29
Class-Conditional MusicDET	✓	49.50	50.25	49.79	50.38	49.72	49.66	50.09	51.86	49.80	49.75	49.95	50.07
CARLA (ours)	✓	9.30	7.82	**8.07**	**6.79**	**5.84**	5.40	**4.74**	**4.55**	**2.48**	**3.60**	**3.74**	**5.67**
With + RandAug													
RawNet3 + RandAug	✓	11.73	7.76	12.34	9.42	8.74	3.08	6.02	9.09	5.09	8.74	8.90	8.27
SpecTTTra-α + RandAug	✓	12.22	7.69	10.89	8.18	6.73	2.56	7.27	13.64	4.97	7.20	7.00	8.03
SpecTTTra-β + RandAug	✓	13.39	12.57	14.55	12.96	10.07	3.91	8.98	13.64	7.45	8.65	7.90	10.37
SpecTTTra-γ + RandAug	✓	12.90	10.69	15.00	10.15	8.89	3.41	8.08	9.49	6.83	7.96	8.45	9.26
MusicDET + RandAug	✓	59.83	54.01	67.26	57.04	65.10	63.03	40.87	36.79	47.83	51.00	56.77	54.50
Class-Conditional MusicDET + RandAug	✓	49.83	50.82	50.66	51.13	50.02	49.94	50.48	51.31	50.47	49.89	50.12	50.43
CARLA + RandAug (ours)	✓	**5.49**	**6.15**	**5.46**	**4.24**	**3.93**	**3.41**	**3.45**	**4.55**	**1.19**	**2.62**	**2.49**	**3.91**

Table 3 shows that CARLA achieves the lowest EER on eight of the 11 target groups in the clean set and a 5.67% mean EER, versus 7.58% for the strongest non-CARLA baseline, SpecTTTra-β (**1.91 percentage points lower**). CARLA+RandAug improves the mean to **3.91%**, a further **1.76 percentage-point reduction** from CARLA and **3.67 percentage points below** SpecTTTra-β. It has the lowest or tied-lowest EER on every target group.



(a) Generator-wise EER across cumulative bank sizes. (b) Clean test EER as the bank grows.

Figure 3: Generator-wise (left) and clean-test (right) EER as the cumulative adapter bank grows.

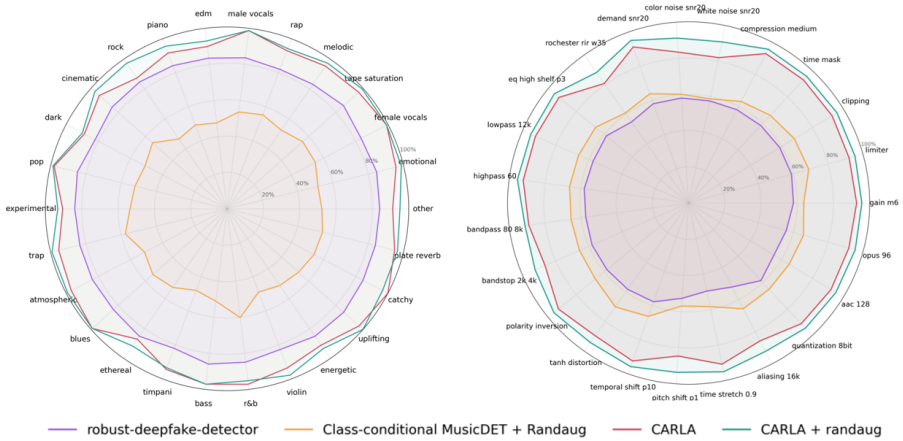
The heatmap in Figure 3(a) shows that adding adapters can also improve detection of generators already in the bank: Suno v4, for example, falls from 15.6% EER with one adapter to 4.9% with 11. These observations support few-shot bank expansion as generator versions evolve. Figure 4(a) reveals substantial differences among Suno's styles: CARLA+RandAug ranges from about 89.9% accuracy on *dark* to 100% on *uplifting*, and its advantage over the baselines varies by style. It motivates reporting style-level performance alongside generator-level results, a dimension less visible in earlier generator-centric evaluations (Vila et al., 2025; Go & Kim, 2026). Additional Jamendo and Mureka style breakdowns appear in Appendix F.

4.3 Robustness Evaluation

Under the 23 fixed corruptions, CARLA+RandAug reduces generator-macro EER from 6.46% to **4.19%** (**2.27 percentage points**) and condition-macro EER from 7.27% to **4.58%** (**2.69 percentage points**) relative to CARLA (Table 4). The strongest non-CARLA generator-macro result is 7.61% from SpecTTTra-α+RandAug. CARLA's $F1_{\mathrm{AI}}$ rises from 0.90 to 0.94, synthetic sensitivity from 0.87 to 0.93, and real-song specificity from 0.95 to 0.96. Thus, audio augmentation improves both detection of corrupted synthetic songs and recognition of real songs.

Table 4: Robustness under 23 fixed corruptions. Gen.-macro EER averages the 11 generators equally; condition-macro metrics average the 23 conditions equally. ΔEER is corruption minus clean Gen.-macro EER on paired data. Dashes mark unavailable scores.

Method	Few-Shot	Gen.-macro EER ↓ (%)	Cond.-macro EER ↓ (%)	$F1_{\mathrm{AI}}$ ↑	Sens. ↑	Spec. ↑	ΔEER ↓ (pp)
Without* + *RandAug							
RawNet3	✓	19.12	20.07	0.78	0.76	0.82	2.07
SpecTTTra-α	✓	9.03	9.23	0.89	0.89	0.89	1.23
SpecTTTra-β	✓	8.56	9.85	0.87	0.88	0.88	0.98
SpecTTTra-γ	✓	9.78	10.82	0.87	0.87	0.89	1.68
robust_deepfake_detector	×						
MusicDET	✓	39.11	39.99	0.66	1.00	0.00	0.18
Class-Conditional MusicDET	✓	28.25	29.11	0.64	0.71	0.60	1.02
CARLA (ours)	✓	**6.46**	**7.27**	0.90	0.87	0.95	0.83
With* + *RandAug							
RawNet3 + RandAug	✓	14.20	14.62	0.85	0.87	0.84	1.81
SpecTTTra-α + RandAug	✓	7.61	8.32	0.90	0.93	0.87	0.94
SpecTTTra-β + RandAug	✓	10.83	11.97	0.87	0.89	0.85	0.48
SpecTTTra-γ + RandAug	✓	9.89	10.87	0.88	0.91	0.86	1.03
MusicDET + RandAug	✓	54.87	55.82	0.00	0.00	0.98	-0.51
Class-Conditional MusicDET + RandAug	✓	29.59	30.50	0.52	0.50	0.78	1.41
CARLA + RandAug (ours)	✓	**4.19**	**4.58**	0.94	0.93	0.96	0.86



(a) Performance across different style on SUNO. (b) Performance across various corruptions.

Figure 4: Detection accuracy by Suno style (left) and corruption condition (right). CARLA+RandAug is generally strongest, while noise and room response remain challenging.

CARLA+RandAug lowers generator-macro corruption EER from 6.46% to 4.19% and condition-macro EER from 7.27% to 4.58% (Table 4). Its $F1_{\mathrm{AI}}$ rises from 0.90 to 0.94, with synthetic sensitivity increasing from 0.87 to 0.93 and specificity from 0.95 to 0.96. The corruption penalty is similar (0.83 versus 0.86 percentage points): augmentation lowers absolute error without eliminating the corruption gap. Figure 4 shows gains across most displayed groups; white noise and room impulse response remain comparatively difficult.

4.4 Ablation Studies

We ablate Stage 1 real-music augmentation to test whether a corruption-aware reference improves both clean and corrupted detection. We then isolate Stage 2 adaptation, LoRA placement in the projectors and experts, and the contribution of bank growth and score normalization/routing. Finally, Hyperparameter analysis over support size, LoRA rank, and optimization steps test whether greater data or model capacity consistently helps few-shot adaptation. Details are provided in Appendix G.

Table 5: Pooled clean and paired-corruption EER for Stage-1 attribution; development-only component-isolation clean EER.

Variant	Clean EER (%)	Corrupt. EER (%)
Formal two-stage attribution, $N = 100$		
CARLA (no RandAug), Stage 2	4.292	7.442
CARLA + RandAug, Stage 2	**3.392**	**4.946**
Development-only component isolation		
CARLA base, w/o Stage 2	55.292	—
CARLA + Rank-4 LoRA, bank/router	5.750	—
LoRA on experts only	6.880	—
LoRA on projectors only	7.491	—
LoRA on experts + projectors	5.895	—
Cumulative bank, $1 \rightarrow 11$	$6.167 \rightarrow 5.750$	—
Raw NLL $\rightarrow$ normalized/router	$5.809 \rightarrow 5.750$	—

Table 6: Hyperparameter analysis of CARLA.

Factor	Setting	Clean	Corrupt.	Penalty
		EER (%) / pp		
Support budget: CARLA + RandAug, rank 4, lr=10^{-4}, 750 steps				
N	50	6.445	6.886	1.600
N	**100**	**4.511**	**5.650**	**1.348**
N	200	9.073	9.236	3.942
LoRA rank: $N = 50$, lr=10^{-4}, 750 steps				
Rank	**4**	**6.172**	**7.019**	1.139
Rank	8	8.449	8.727	2.318
Rank	16	6.525	7.767	2.420
Optimization steps: $N = 50$, rank 4, lr=10^{-4}				
Steps	500	7.133	7.589	1.261
Steps	**750**	6.172	7.019	1.139
Steps	850	5.990	**6.772**	1.122
Steps	1000	**5.857**	6.888	**0.960**

Stage 1 RandAug lowers clean EER from 4.292% to 3.392% and corruption EER from 7.442% to 4.946% (Table 5), supporting the role of the shared corruption-aware real reference. On development data, the full bank substantially improves on the unadapted base, and adapting both experts and projectors outperforms adapting either alone. Table 6 shows that more support is not automatically better: $N = 100$ yields 4.511% clean EER versus 6.445% at $N = 50$ and 9.073% at $N = 200$, supporting a practical few-shot budget. In the separate rank sweep at $N = 50$, rank 4 also outperforms ranks 8 and 16; added capacity may overfit a small support set. We select the $N = 100$, rank-4 recipe for formal evaluation; complete component and optimization-step results are in Appendix G.

5 Conclusion

We introduced **BASS**, a version-aware benchmark for evaluating synthetic song detection across evolving generators, musical styles, and common audio corruptions. We proposed **CARLA**, which learns a shared corruption-aware real-music density and incorporates emerging generators through few-shot, generator-specific LoRA adapters without requiring generator identity at inference. Experiments on BASS show that CARLA achieves the strongest overall performance among the baselines across generator and corruption evaluations. Sequential bank results further show that adding adapters can improve detection of previously represented generators. Together, corruption-aware modeling of real songs and few-shot adaptation provide a practical direction for detection as music generators evolve and audio undergoes real-world transformations.

AI USE STATEMENT

Generative AI systems were used to generate the synthetic-song data used in this study, including songs from Suno and Mureka. We also used AI assistance to clean and reformat dataset metadata and manifests, and to polish and translate parts of the manuscript. All AI-assisted outputs, including data records and manuscript text, were reviewed and verified by the authors. The authors retained responsibility for the dataset definition, split construction, experimental protocol, scientific claims, and final manuscript.

ETHICS STATEMENT

Our research was conducted with careful consideration of responsible data sourcing, copyright, and licensing transparency. BASS contains synthetic songs collected from publicly accessible pages on Suno and Mureka, together with human-produced songs from the MTG-Jamendo dataset (Bogdanov et al., 2019). To avoid redistributing third-party audio content, we do not host or release the original Suno or Mureka audio files. Instead, the public benchmark provides source references, generator/version annotations, benchmark splits, and the metadata required to reproduce our evaluation where such source references remain publicly available. This release strategy follows prior AI-generated music benchmarks that separate benchmark annotations from copyrighted or externally hosted source audio (Rahman et al., 2025; Batra et al., 2025).

For MTG-Jamendo, we rely on its official public distribution and artist-disjoint split protocol. Accordingly, the CC BY-NC 4.0 license of BASS applies only to materials created by us, including benchmark annotations, split manifests, metadata organization, and other non-audio benchmark resources, and does not alter the ownership or licensing terms of the original songs or third-party platform content. The benchmark is intended solely to support non-commercial academic research on synthetic-song detection, robustness, and cross-generator generalization.

REFERENCES

Darius Afchar, Gabriel Meseguer-Brocal, Kamil Akesbi, and Romain Hennequin. A fourier explanation of ai-music artifacts. In *Proceedings of the 26th International Society for Music Information Retrieval Conference (ISMIR)*, pp. 739–746, Daejeon, South Korea, 2025a. doi: 10.5281/ZENODO.17706577.

Darius Afchar, Gabriel Meseguer-Brocal, and Romain Hennequin. Ai-generated music detection and its challenges. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2025b.

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

Arnesh Batra, Dev Sharma, Krish Thukral, Ruhani Bhatia, Naman Batra, and Aditya Gautam. Melody or machine: Detecting synthetic music with dual-stream contrastive learning. *arXiv preprint arXiv:2512.00621*, 2025.

Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. The mtg-jamendo dataset for automatic music tagging. In *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States, 2019. URL `http://hdl.handle.net/10230/42015`.

Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 702–703, 2020.

Deezer. Ai music has surpassed 50% of new music uploads for the first time. `https://newsroom-deezer.com/2026/07/ai-music-exceeds-50-percent-daily-uploads-deezer/`, July 2026. Accessed: 2026-09-23.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Emile Dugelay, Thomas Barand, Aurélien Laouar, Baptiste Campeas, Darius Afchar, and Romain Hennequin. Improved robustness in ai-generated music detection. *arXiv preprint arXiv:2607.27454*, 2026.

Seonghyeon Go and Yumin Kim. Haim: Human-ai music datasets for ai music production tracking benchmark. *arXiv preprint arXiv:2606.01686*, 2026.

Chaolei Han, Hongsong Wang, and Jie Gui. Musicdet: Zero-shot ai-generated music detection. *arXiv preprint arXiv:2605.18072*, 2026a.

Yan Han, Zhibin Wen, Yuan Wang, Shuangrun Shao, Xiaobing Li, Yang Xu, and Wei Li. Beyond artifacts: Towards generalizable synthetic song detection via music-intrinsic features. *arXiv preprint arXiv:2606.16612*, 2026b.

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. *arXiv preprint arXiv:2106.09685*, 2021.

Jee-weon Jung, You Jin Kim, Hee-Soo Heo, Bong-Jin Lee, Youngki Kwon, and Joon Son Chung. Pushing the limits of raw waveform speaker recognition. *arXiv preprint arXiv:2203.08488*, 2022.

Yumin Kim and Seonghyeon Go. Segment transformer: Ai-generated music detection via music structural analysis. In *2025 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 664–669. IEEE, 2025.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Xiang Li, Pin-Yu Chen, and Wenqi Wei. Measuring the robustness of audio deepfake detection under real-world corruption. In *Proceedings of the Sixteenth ACM Conference on Data and Application Security and Privacy*, pp. 145–158, 2026.

Yizhi Li, Ruibin Yuan, Ge Zhang, Yinghao Ma, Xingran Chen, Hanzhi Yin, Chenghao Xiao, Chenghua Lin, Anton Ragni, Emmanouil Benetos, et al. Mert: Acoustic music understanding model with large-scale self-supervised training. In *International Conference on Learning Representations*, volume 2024, pp. 12181–12204, 2024a.

Yupei Li, Manuel Milling, Lucia Specia, and Björn W Schuller. From audio deepfake detection to ai-generated music detection–a pathway and overview. *arXiv preprint arXiv:2412.00571*, 2024b.

David López-Ayala, Asier Cabello, Pablo Zinemanas, Emilio Molina, and Martín Rocamora. Ai-generated music detection in broadcast monitoring. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12342–12346. IEEE, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Heewon Oh. Artifactnet: Detecting ai-generated music via forensic residual physics. *arXiv preprint arXiv:2604.16254*, 2026.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pp. 487–503, 2021.

Awsaf Rahman, Zaber Ibn Abdul Hakim, Najibul Haque Sarker, Bishmoy Paul, and Anowarul Fattah. Sonics: Synthetic or not-identifying counterfeit songs. In *International Conference on Learning Representations*, volume 2025, pp. 21421–21451, 2025.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Jenna Rutowski, Tre DiPassio, Benjamin R Thompson, Michael C Heilemann, and Mark F Bocko. University of rochester room impulse response dataset. University of Rochester, 2024.

Haohan Shi, Xiyu Shi, Safak Dogan, Saif Alzubi, Tianjin Huang, and Yunxiao Zhang. Benchmarking audio deepfake detection robustness in real-world communication scenarios. In *2025 33rd European Signal Processing Conference (EUSIPCO)*, pp. 566–570. IEEE, 2025.

Joachim Thiemann, Nobutaka Ito, and Emmanuel Vincent. The diverse environments multi-channel acoustic noise database (demand): A database of multichannel environmental noise recordings. *Proceedings of Meetings on Acoustics*, 19(1):035081, 2013.

Laura Cros Vila, Bob LT Sturm, Luca Casini, and David Dalmazzo. The ai music arms race: On the detection of ai-generated music. *Transactions of the International Society for Music Information Retrieval*, 8(1):179–194, 2025.

Haolin Wu, Jing Chen, Ruiying Du, Cong Wu, Kun He, Xingcan Shang, Hao Ren, and Guowen Xu. Clad: Robust audio deepfake detection against manipulation attacks with contrastive learning. *Knowledge-Based Systems*, pp. 115179, 2025.

Yuankun Xie, Jingjing Zhou, Xiaolin Lu, Zhenghao Jiang, Yuxin Yang, Haonan Cheng, and Long Ye. Fsd: An initial chinese dataset for fake song detection. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4605–4609. IEEE, 2024.

Yen-Tung Yeh, Junghyun Koo, Marco A Martínez-Ramírez, Wei-Hsiang Liao, Yi-Hsuan Yang, and Yuki Mitsufuji. Fx-encoder++: Extracting instrument-wise audio effects representations from mixtures. *arXiv preprint arXiv:2507.02273*, 2025.

Yongyi Zang, Jiatong Shi, You Zhang, Ryuichi Yamamoto, Jionghao Han, Yuxun Tang, Shengyuan Xu, Wenxiao Zhao, Jing Guo, Tomoki Toda, et al. Ctrsvdd: A benchmark dataset and baseline analysis for controlled singing voice deepfake detection. *arXiv preprint arXiv:2406.02438*, 2024a.

Yongyi Zang, You Zhang, Mojtaba Heydari, and Zhiyao Duan. Singfake: Singing voice deepfake detection. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12156–12160. IEEE, 2024b.

Zirui Zhang, Wei Hao, Aroon Sankoh, William Lin, Emanuel Mendiola-Ortiz, Junfeng Yang, and Chengzhi Mao. I can hear you: Selective robust training for deepfake audio detection. In *International Conference on Learning Representations*, volume 2025, pp. 9728–9746, 2025.

Haina Zhu, Yizhi Zhou, Hangting Chen, Jianwei Yu, Ziyang Ma, Rongzhi Gu, Yi Luo, Wei Tan, and Xie Chen. Muq: Self-supervised music representation learning with mel residual vector quantization. *arXiv preprint arXiv:2501.01108*, 2025.

A Dataset Inventory and Partitioning

Collection snapshot. Table 7 reports the source/version inventory used in the data audit. Duration totals are computed from category counts and mean durations: 1,348.28 hours for Suno and 1,425.44 hours for Mureka, or 2,773.72 hours in total. These are durations of collected source songs, not model input windows or expanded corruption views. Platform-provided labels, including the "all" suffixes, are retained as categories; they are not independently verified statements about distinct internal model architectures. In particular, `unknown` is an unresolved category, not an identified twelfth generator. Excluding its 2,811 tracks leaves 44,521 synthetic tracks across 11 identified categories before split-specific evaluation filtering.

Table 7: Synthetic collection inventory. The unresolved Mureka category is reserved for development and excluded from the formal 11-generator matrix.

Platform	Version label	Tracks	Mean duration (s)
Suno	v4	5,111	206.05
	v4.5	1,869	215.17
	v4.5+	1,097	243.73
	v4.5-all	1,534	222.16
	v5	6,000	232.54
	v5.5	6,000	232.52
Mureka	unknown (development)	2,811	190.53
	v7.5	1,993	188.05
	v7.5-all	111	180.85
	v7.6	806	216.44
	v8	10,000	212.05
	v9	10,000	190.62
Total	12 collection categories	47,332	

Collection provenance and metadata. The Suno snapshot includes 21,455 search records and 156 feed records. Its catalog provenance lists 266 terms drawn from published styles, musical glossary entries, and official examples. Mureka provenance comprises 14,489 official-page, 10,897 creator-page, and 335 listing records. These sources do not imply identical search interfaces or a controlled factorial sampling of version and style. The retained metadata supports stratified analysis but is incomplete and platform-dependent. Empty lyrics fields in the aligned manifests are not treated as transcriptions. We do not assume that prompts or style tags establish perceptual genre equivalence across platforms.

Connected-group partitioning. Construct an undirected graph whose vertices are synthetic records. Connect two records when any retained, nonempty creator identifier,, and use connected components as indivisible groups. The splitter targets 60/20/20 train/validation/test proportions within each category, using 500 deterministic randomized/greedy allocation attempts and a normalized squared-error criterion. Group integrity takes precedence over exact proportions.

Human reference and audit boundary. The human reference contains 55,525 MTG-Jamendo songs with the official `raw_30s` split-0. We retain the official artist-disjoint partitions and audit their identity fields. The supplied audit reports 25 cross-split duplicate-audio-hash groups involving 62 rows in this human reference; Synthetic and human data share field definitions and evaluation partitions, but are neither matched track by track nor guaranteed to share the same genre distribution.

B Corruption Construction and Reproducibility

Literature-grounded corruption taxonomy. The BASS suite consolidates and extends corruption mechanisms that recur across robust audio-deepfake and AI-music detection studies rather than adopting a single prior benchmark verbatim. Selective robust training and systematic corruption

evaluation cover broad noise, signal-modification, temporal, and compression effects (Zhang et al., 2025; Li et al., 2026); CLAD studies inexpensive manipulations including level changes, fading, noise, time stretching, resampling, temporal shifting, and echo (Wu et al., 2025); communication-oriented evaluation highlights codec and channel degradation (Shi et al., 2025); and AI-music studies demonstrate sensitivity to filtering, sample-rate conversion, speed modification, and pitch shifting (Vila et al., 2025; Dugelay et al., 2026). BASS reorganizes these recurring failure modes for full-song evaluation into five families and uses one fixed operating point per condition. Environmental-noise and room-acoustics conditions use established external resources, cited below.

Table 8: The 23 fixed evaluation conditions. Clean is an additional control. Each condition is applied independently. SNR denotes signal-to-noise ratio, RIR room impulse response, EQ equalization, and Q filter quality factor.

Operation	Fixed parameters or resource
Level, dynamics, and missing segments	
Gain	-6 dB
Limiter	Drive 6 dB; ceiling -1 dBTP
Clipping	Drive 3 dB
Time masking	Missing ratio 0.005; frame length 30 ms
Dynamic compression	Threshold -18 dB; ratio 4; attack/release $10/100$ ms
Noise and room acoustics	
White noise	SNR 20 dB
Pink/colored noise	SNR 20 dB
Environmental noise	DEMAND (Thiemann et al., 2013); SNR 20 dB
Room response	Rochester RIR (Rutowski et al., 2024); dry/wet coefficient 0.35
Spectrum, polarity, and nonlinearity	
High-shelf EQ	8 kHz; gain 3 dB; $Q = 1$
Low-pass filter	Cutoff 12 kHz
High-pass filter	Cutoff 60 Hz
Band-pass filter	Passband 80 Hz–8 kHz
Band-stop filter	Stopband 2–4 kHz
Polarity inversion	Sign reversal
Tanh distortion	Drive 3 dB
Time, pitch, and speed	
Temporal shift	Fraction 0.1 with circular rollover
Pitch shift	$+1$ semitone
Time stretch	Speed factor 0.9
Sampling, quantization, and codecs	
Resampling roundtrip	Intermediate sample rate 16 kHz
Quantization	8 bit; no dither
AAC	128 kbps; evaluation only
Opus	96 kbps

Fixed evaluation processing. For each manifest row, we decode the source audio, convert it to stereo at 44.1 kHz, and peak-normalize the clean waveform. The specified corruption is then applied before the model-specific evaluation crop and branch resampling. There is no global peak renormalization after corruption, which would otherwise suppress effects such as gain changes. Resources comprise 176 DEMAND (Thiemann et al., 2013) and 90 Rochester RIR (Rutowski et al., 2024) WAV files. Manifests bind track, operation, parameters, seed, and resource identity. AAC and Opus use FFmpeg.

Assigned and paired manifests. Assigned evaluation derives a stable hash from the mapping seed, split, label, and track identity, and maps it to one of the 23 conditions. Paired evaluation selects a stable, version-stratified subset and expands each selected track into 24 rows. All views retain their source partition.

Online training differs from fixed evaluation. Following the audio RandAugment sampling principle (Zhang et al., 2025; Cubuk et al., 2020), the random transform T in Eq. 4 samples one

operation uniformly from the 22-operation pool, applying it with probability 0.9 and otherwise returning the identity. AAC is excluded from this pool. Parameters are sampled from operation-specific physical ranges, rather than a shared magnitude. Ranges include gain $[-12, 12]$ dB, SNR $[10, 30]$ dB, pitch $[-2, 2]$ semitones, and speed $[0.6, 1.5]$. The selected training configuration applies augmentation to the normalized 60-second model window before encoder-specific resampling; fixed evaluation instead corrupts the source waveform before cropping.

C Architecture and Likelihood Details

Music representation. Equation 1 defines the four-branch fusion used throughout the paper, following the SOFIA-style music-intrinsic feature design (Han et al., 2026b). The frozen branches are Fx-Encoder++ (Yeh et al., 2025), MuQ (Zhu et al., 2025), MERT (Li et al., 2024a), and Wav2Vec2 (Baevski et al., 2020). Encoder E_k has output width d_k; projector P_k maps it to width D. For a vector v, the implementation uses $\mathrm{norm}(v) = v/\max(\|v\|_2, \epsilon)$, where $\|\cdot\|_2$ is the Euclidean norm and ϵ is a numerical floor: 10^{-6} for encoder branches and 10^{-12} for router normalization. The gate in Eq. 1 is

$$G([h_1; \ldots; h_4]) = \mathrm{softmax}(W_a[h_1; \ldots; h_4] + b_a), \tag{7}$$

where W_a and b_a are the gate weight matrix and bias. Softmax normalizes the exponentials of its four logits to give a. The trainable representation parameters ϕ comprise P_k, ψ_k, and G, but exclude the frozen encoders E_k. The gate fuses music features; the separate bank router in Eq. 6 weights adapter scores.

Table 9: Frozen encoder branches. Pooling produces one vector per input window.

Encoder	Rate (kHz)	Channels	d_k	Pooling
Fx-Encoder++	44.1	2	128	Encoder embedding
MuQ	24	1	1,024	Temporal mean
MERT	24	1	768	Layer/time mean
Wav2Vec2	16	1	768	Layer/time mean

Each projector is Linear–LayerNorm–GELU–Dropout with $D = 256$ and dropout probability 0.1. GELU denotes the Gaussian error linear unit. Each expert is a linear map of width D followed by GELU. The gate maps the concatenated features to four weights. The fused representation is not normalized before entering the flow.

Vector flow and priors. The flow comprises eight invertible steps, each containing activation normalization (ActNorm), a reverse permutation, and affine coupling, following standard invertible density-estimation components used by RealNVP and Glow (Dinh et al., 2016; Kingma & Dhariwal, 2018). The coupling network is a multilayer perceptron with widths $128 \to 512 \to 512 \to 256$ and GELU between hidden layers. It predicts a shift and a positive scale for half of the feature coordinates, conditioned on the other half. Log-determinants are accumulated across steps. ActNorm is initialized from a data batch; the last coupling layer is initialized to zero.

The fixed centers are $\mu_0 = R\mathbf{1}_D$ and $\mu_1 = -R\mathbf{1}_D$, where $R = 5$ is the center offset and $\mathbf{1}_D$ is the all-ones vector of length D. Both priors use the D-dimensional identity covariance. For latent coordinate $i \in \{1, \ldots, D\}$, write z_i and $\mu_{y,i}$ for the corresponding vector entries. The Gaussian term in Eq. 2 is

$$\log \mathcal{N}(z; \mu_y, I) = -\frac{1}{2} \sum_{i=1}^{D} \left[(z_i - \mu_{y,i})^2 + \log(2\pi) \right]. \tag{8}$$

All logarithms in the paper are natural. The implementation divides negative log-likelihood by $D \log 2$ to report bits per dimension. This positive constant rescales both training objectives equally; support standardization cancels the corresponding score scale.

Density interpretation. Equation 2 models the music representation, not the waveform: the encoders and projectors are not invertible and supply no waveform Jacobian. The class-conditional loss includes both the Gaussian term and the flow Jacobian. Stage 1 encourages both views to have high real-prior density, but does not impose equality of their features or distributions.

D TRAINING PROCEDURE AND IMPLEMENTATION

Low-rank updates. For each projector or expert linear weight $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, where d_{in} and d_{out} are its input and output widths, adapter g uses the LoRA parameterization (Hu et al., 2021)

$$W_g = W + \frac{\alpha}{r} B_{g,W} A_{g,W}, \qquad A_{g,W} \in \mathbb{R}^{r \times d_{\text{in}}}, \quad B_{g,W} \in \mathbb{R}^{d_{\text{out}} \times r}. \tag{9}$$

Here W_g is the adapted weight, r the update rank, α its scale, and $\mathbb{R}$ the real numbers. The collection of all new matrices $A_{g,W}, B_{g,W}$ forms δ_g; all original weights remain frozen.

Algorithm 1: CARLA training and inference

Input: Real training songs, corruption sampler, frozen encoders, and generator-specific labeled support sets $\mathcal{S}_g$.

1. Initialize the representation and flow using clean real-song negative log-likelihood $\ell(x, 0)$.
2. **Stage 1:** Sample real songs x and transformations T. Update ϕ, θ using Eq. 4; freeze encoders and priors.
3. Retain the real-validation-selected representation H_0 and flow. Initialize the adapter bank $\mathcal{B}$ as empty.
4. **Stage 2:** For each generator g, initialize δ_g with zero effective update. Optimize only δ_g on $\mathcal{S}_g$ using Eq. 5.
5. Compute the support prototype and calibration statistics (Appendix E); add them with δ_g to $\mathcal{B}$.
6. **Inference:** Compute $H_0(x)$, route with Eq. 6 and aggregate the real-prior scores of Eq. 3, and threshold the result.

Table 10: Parameter update scope and primary recipe. Stage 1 starts from a clean real-only initializer.

Item	Stage 1	Stage 2
Training data	Real clean/corrupted pairs	N clean real + N clean synthetic
Pretrained encoders	Frozen	Frozen
Projectors / experts	Original weights updated	Only new LoRA matrices updated
Gate / flow / priors	Gate and flow updated; priors fixed	All frozen
Optimizer	AdamW	AdamW
Learning rate	5×10^{-4}	10^{-4}
Weight decay	10^{-4}	10^{-4}
Duration / crop	60 s; random training crop	60 s; start support crop
Validation / inference crop	Center	Center
Training budget	20 augmentation epochs	750 steps at $N = 100$
Adapter rank / scale	Not applicable	$r = 4$, $\alpha = 16$
Model selection	Real-only validation NLL	Fixed development recipe

Stage 1 uses AdamW (Loshchilov & Hutter, 2017) moments $(0.9, 0.98)$, gradient clipping at 5, and equal clean/augmented loss weights. Frozen-encoder embeddings may be cached per epoch with the augmentation identity retained. The selected encoder compatibility setting is `checkpoint_repaired`, initialization seed 42; legacy embeddings cannot be substituted into the same checkpoints. Stage 2 uses support/training seeds 42, 43, and 44. LoRA matrices $A_{g,W}$ use Kaiming-uniform initialization and $B_{g,W}$ starts at zero, so the initial adapter reproduces the shared base. Dropout is disabled on the Stage 2 evaluation-mode base; low-rank parameters still receive gradients. No original bias or LayerNorm parameter is updated.

E ROUTING, CALIBRATION, AND DEPLOYMENT

Prototype routing. Let $\mathcal{S}_g^y$ denote the waveforms with label y in the labeled support set $\mathcal{S}_g$; each class subset contains N songs. Using the unadapted representation H_0, the prototype is

$$c_g = \operatorname{norm}\left(\frac{1}{N}\sum_{x\in\mathcal{S}_g^1} H_0(x)\right). \tag{10}$$

Normalization follows averaging, as specified in Appendix C. The implemented cosine similarity in Eq. 6 is $d_g(x) = \operatorname{norm}(H_0(x))^\top c_g$, where superscript $\top$ denotes transpose. Let $\mathcal{B}$ be the current nonempty bank and $|\mathcal{B}|$ its size. The router selects $\min(K, |\mathcal{B}|)$ adapters and normalizes weights only over this set. The configuration uses $K = 3$ and $\tau = 0.1$.

Support calibration. The adapter score $s_g(x) = s_{H_g}(x)$ follows Eq. 3. Let $\bar{s}_g$ be its mean over both support classes, m_g its real-support mean, and v_g its sample standard deviation over both classes:

$$\begin{aligned} \bar{s}_g &= \frac{1}{2N}\sum_{(x,y)\in\mathcal{S}_g} s_g(x), & m_g &= \frac{1}{N}\sum_{x\in\mathcal{S}_g^0} s_g(x), \\ v_g &= \sqrt{\frac{1}{2N-1}\sum_{(x,y)\in\mathcal{S}_g}(s_g(x)-\bar{s}_g)^2}, & \widetilde{s}_g(x) &= \frac{s_g(x)-m_g}{v_g}. \end{aligned} \tag{11}$$

Nonfinite or degenerate scales are rejected. Calibration uses clean support only. The aggregate $S(x)$ is a weighted calibrated detection score, not a mixture log-likelihood. With development-selected threshold t_{dev}, the decision is $\hat{y} = \mathbf{1}[S(x) < t_{\text{dev}}]$, where the indicator $\mathbf{1}[\cdot]$ equals one when its condition holds and zero otherwise.

Optimization and adapter capacity. Stage 1 uses $\lambda = 1$. In Stage 2, minibatch losses are averaged within each represented class and then across classes; with both classes present this gives equal class weight, matching Eq. 5. A single-class minibatch uses its class mean. With four projector and four expert updates, the adapter has $r(\sum_{k=1}^{4} d_k + 12D) = 23{,}040$ parameters for $r = 4$. Each adapter starts from the same base. Frozen gate weights can still yield different gate outputs when adapted projector features change.

Evaluation and reproducibility. The unresolved Mureka category is used to fix the adaptation recipe, router settings, and threshold before formal target evaluation. Target support is separate from target validation/test data. Previous adapters and their support statistics remain fixed as the bank grows; changing the selected adapters can nevertheless change predictions. The complete bank-growth trajectory therefore measures retention. The implementation evaluates all adapter scores before selection, so selecting K adapters does not imply that only K forward paths are computed.

Threshold-dependent metrics use the fixed development threshold. Equal error rate (EER) and area under the receiver operating characteristic curve (AUROC) characterize score rankings without reusing test-derived thresholds for deployment. Checkpoint hashes, support identities, low-rank injection keys, seeds, crop conventions, and calibration statistics bind each adapter to its base. Corruption manifests additionally retain parameters and resource identities.

F FINE-GRAINED STYLE RESULTS

The main text reports the Suno style comparison in Figure 4. Table 11 complements it with weighted accuracy across each corpus's own metadata groups. The labels are analyzed independently and do not define matched styles across datasets. CARLA+RandAug is highest on all three aggregates, while the smaller Mureka gain reflects CARLA's already high starting accuracy.

Figure 5 shows that the aggregate does not imply uniform performance within a corpus. CARLA+RandAug ranges from 84.9% on Jamendo hiphop to 99.5% on soundtrack, and from 72.7% on Mureka blues to 100% on country. These gaps motivate reporting group-level diagnostics alongside cross-generator averages.

Table 11: Weighted accuracy (%) across dataset-specific style groups. Groups are formed independently from each corpus's metadata.

Corpus	CARLA	CARLA+RandAug	Robust detector	Class-cond. MusicDET+RandAug
Jamendo	90.40	**94.08**	54.40	79.07
Suno	92.38	**95.78**	84.05	52.33
Mureka	97.39	**98.26**	38.01	59.49

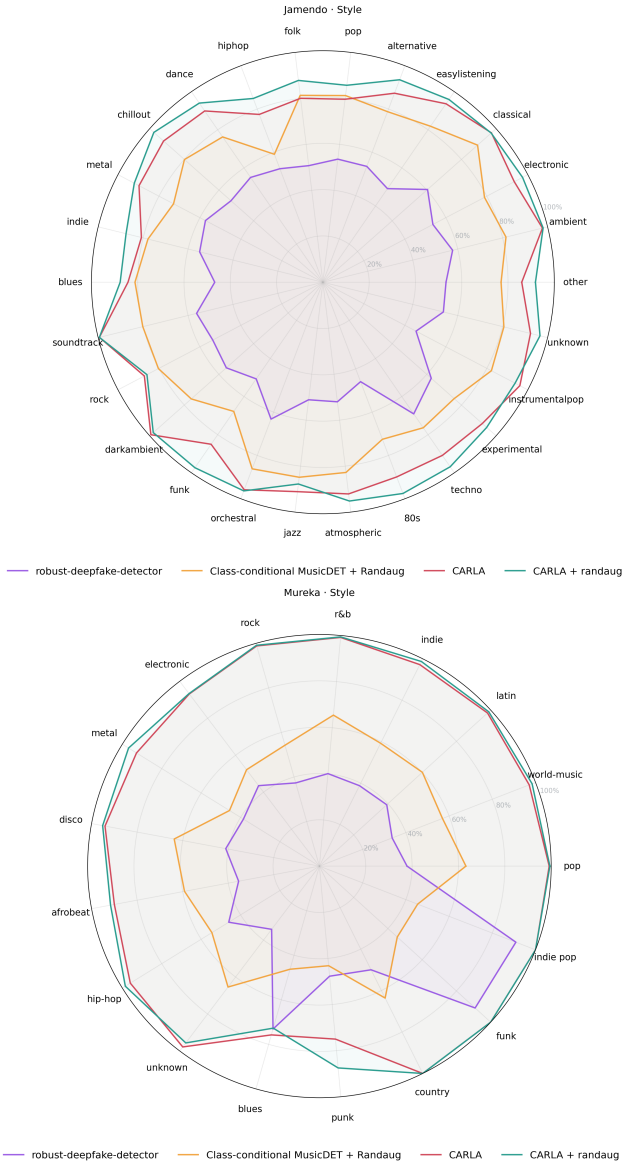


Figure 5: Per-style detection accuracy for Jamendo (top) and Mureka (bottom). The group sets are corpus-specific; lowest groups remain harder despite strong weighted averages.

G Ablation and Hyperparameter Diagnostics

We separate the formal attribution of Stage 1 augmentation from development-only component and hyperparameter diagnostics. In the formal comparison, the Stage 2 recipe is held fixed while Stage 1 is trained with or without real-song RandAug. The component and parameter studies use development data; their EERs should not be interpreted as additional formal test results.

Table 12: Formal Stage 1 attribution and development-only component isolation. The formal rows report pooled clean and paired-corruption EER; the remaining rows are development diagnostics. Dashes indicate unavailable corruption results.

Variant	Clean EER (%)	Corruption EER (%)
Formal two-stage attribution, $N = 100$		
CARLA (no RandAug), Stage 2	4.292	7.442
CARLA + RandAug, Stage 2	**3.392**	**4.946**
Development-only component isolation		
CARLA base, without Stage 2	55.292	—
CARLA + rank-4 LoRA, bank/router	5.750	—
LoRA on experts only	6.880	—
LoRA on projectors only	7.491	—
LoRA on experts and projectors	5.895	—
Cumulative bank, $1 \rightarrow 11$	$6.167 \rightarrow 5.750$	—
Raw NLL to normalized/router score	$5.809 \rightarrow 5.750$	—

The formal comparison in Table 12 isolates the contribution of corruption-aware real-density learning: RandAug reduces clean EER from 4.292% to 3.392% and paired-corruption EER from 7.442% to 4.946%. These pooled EERs use a different aggregation from the generator- and condition-macro robustness results in Table 4. On development data, adding Stage 2 adaptation and the bank/router reduces clean EER from 55.292% for the unadapted base to 5.750%. Adapting both experts and projectors performs better than adapting either component alone. Expanding the bank from one to 11 adapters yields a larger improvement than the subsequent normalization and routing refinement; the latter should therefore not be treated as the principal source of the gain.

Table 13: Developmentset based sweeps of support size, LoRA rank, and optimization steps. Each block specifies its other fixed settings. The robustness penalty is computed using paired clean and corrupted views and need not equal the difference between the two EER columns.

Factor	Setting	Clean EER (%)	Corruption EER (%)	Penalty (pp)
Support: CARLA + RandAug, rank 4, 10^{-4} *learning rate, 750 steps*				
N	50	6.445	6.886	1.600
N	100	**4.511**	**5.650**	**1.348**
N	200	9.073	9.236	3.942
Rank: $N = 50$*,* 10^{-4} *learning rate, 750 steps*				
Rank	4	**6.172**	**7.019**	1.139
Rank	8	8.449	8.727	2.318
Rank	16	6.525	7.767	2.420
Steps: $N = 50$*, rank 4,* 10^{-4} *learning rate*				
Steps	500	7.133	7.589	1.261
Steps	750	6.172	7.019	1.139
Steps	850	5.990	**6.772**	1.122
Steps	1000	**5.857**	6.888	**0.960**