Detecting AI-Generated Music Using LambdaResNet and Swin Transformers on Mel-Spectrograms

Candidate Number: 44535, 40538

Abstract

The rise of generative music models has sparked concerns about authenticity and copyright in the music industry. In this paper, we investigate whether lightweight pretrained vision models can accurately classify short clips of music as real or AI-generated. Using the SONICS dataset, we fine-tune two vision-based deep learning architectures–LambdaResNet (a CNN variant with 11M parameters) and Swin Transformer V2 (a small vision transformer with 51M parameters)–on mel-spectrogram representations of the audio clips. The Swin Transformer achieves the highest classification accuracy, although LambdaResNet also shows strong performance with fewer parameters. This suggests that compact pretrained models are sufficient to detect generative artifacts in short spectrograms without needing complex temporal modeling.

1 Introduction

The emergence of generative AI has transformed many creative industries, including music, as AI models can now create songs that closely resemble compositions created by humans. As these models improve, serious questions have been brought forward about copyright infringement, authenticity, and fair compensation for artists. Platforms may experience difficulty identifying AI-generated music that mimics existing artists or floods streaming services with synthetic content.

This motivates our development of reliable methods to automatically distinguish AI-generated music from human-composed music. In this paper, we frame the detection task as a binary classification problem: given a mel-spectrogram representation of an 5-second audio clip, determine whether it was AI-generated or human-composed. Our aim is not to build the most complex system, but to test how well compact, pretrained architectures can solve this task using limited input. We specifically avoid long-sequence or raw waveform models and focus on two pretrained backbones with fewer than 60 million parameters: LambdaResNet and Swin Transformer V2. Our contribution is to show that even small models are sufficient to detect AI-generated music from short clips.

2 Related Work

2.1 AI-Generated Music Datasets

The SONICS dataset [1] was specifically engineered to introduce *long-form* audio detection: it comprises over 97 K full-length songs sourced from both human composers and state-of-the-art generative models, and evaluates detectors on contiguous 30–60 sec mel-spectrogram segments sampled uniformly across each track. This design highlights challenges in modeling temporal consistency and structural musical patterns over extended durations, rather than merely spotting short-term artifacts. In contrast, most subsequent benchmarks have shifted focus to *short-form* clips to enable rapid prototyping and lighter compute requirements. Xie et al.'s FSD [2] offers 5 sec excerpts from Mandarin pop and folk songs, generated via diffusion-based and RNN-based synthesis pipelines, with balanced positive/negative splits and recommended 80/10/10 train/dev/test partitions. Zang et al.

Preprint. Under review.

contribute two complementary short-form datasets: CtrSVDD [3] provides controlled voice segments (3–8 sec) generated under varying prosody and timbre conditions, along with graph-network baseline results; SingFake [4] curates 3–10 sec singing-voice conversion outputs paired with human renditions, offering both in-domain and cross-domain evaluation protocols. Together, these resources cover a wide spectrum of clip lengths, languages, and synthesis methods, enabling systematic comparisons of long- versus short-form detection strategies.

2.2 Anti-Spoofing and Deepfake Detection Methods

Early efforts borrowed heavily from the speech anti-spoofing community, adapting specialized front-ends and classifier architectures to musical contexts. Jung et al.'s AASIST system [5] employs integrated spectro-temporal graph attention networks: local spectrogram patches are first encoded via CNN layers, then aggregated in a graph where nodes represent frequency bands and edges capture temporal correlations, yielding robust detection of sub-second artifact patterns. Tak et al. demonstrate that RawNet2 [6], which uses learnable sinc-based convolutional filters followed by stacked residual blocks and a bi-directional GRU, excels on clips as short as 2 sec, exploiting raw-waveform representations to bypass feature-engineering biases. Further, embedding features from wav2vec 2.0 [7]—a self-supervised transformer pretrained on large-scale speech corpora—when fine-tuned with targeted data augmentation (pitch shifting, time stretching), significantly improve detection robustness under unseen synthesis configurations. Kawa et al. [8] show that hidden representations from Whisper's encoder further boost performance, likely due to its expansive multilingual and multi-task pretraining. Complementary approaches have explored lightweight LFCC-based ResNet classifiers, hybrid CNN-GNN pipelines, and one-class learning frameworks, all aimed at maximizing discriminative power on very short musical snippets with minimal latency.

2.3 Vision-Style Spectrogram Models

Treating spectrograms as images has unlocked the transfer of powerful vision architectures to the audio deepfake domain. Gong et al.'s Audio Spectrogram Transformer (AST) [9] divides a 3–5 sec mel-spectrogram into 16×16 patches, projects them into a 768-dim embedding space, and applies standard transformer encoder layers with a class token for detection; pretrained on AudioSet, AST fine-tuning yields strong artifact discrimination even with limited labeled data. The Conformer architecture [10], which interleaves convolution modules within transformer blocks to capture both local and global context, transfers effectively to brief music clips when pretrained on speech recognition tasks. More recently, compact vision backbones such as ConvNeXt-Tiny and Swin-Transformer-Small have been repurposed: by resizing mel-spectrograms to 224×224 and applying image augmentations (random crop, frequency masking), these models achieve comparable accuracy to specialized audio networks while reducing parameter counts below 50 M. These findings suggest that general-purpose vision models, when properly adapted to audio representations, provide a flexible and efficient alternative for short-form AI-generated music detection.

3 Methodology

3.1 Dataset and Preprocessing

We constructed our dataset using SONiCS as the base source, selecting 10,000 real and 10,000 AI-generated songs. More specifically, we used the first two zipped files from the SONICS AI generated music dataset and the first 10,000 YouTube links contained in the real_songs.csv file. This was done in the interest of time instead of processing all 100,000 files. We resampled each track to 22.05 kHz and extracted non-overlapping 5-second segments. Each segment was converted to a mel-spectrogram with 256 bands, hop length 512, and FFT size 1024. We converted the power spectrogram to the dB scale (clipped at 80 dB), normalized to [0, 1], resized to 256×256, and duplicated channels to (3, 256, 256) to match ImageNet model input.

3.2 Dataset Splitting

We split the processed tensors using a stratified 60/20/20 train/val/test partition. Each CSV entry contains the tensor path and a binary label. A custom PyTorch Dataset class loads tensors on-the-fly.

3.3 Model Architectures

LambdaResNet26rp_256 LambdaResNet is a variant of the ResNet architecture in which traditional 3×3 convolutions are replaced by lambda layers. Each lambda layer consists of two key steps:

- Feature maps are transformed into queries Q, keys K, and values V via learned 1×1 convolutions.
- Keys are softmax-normalized and used to aggregate global context into a lambda matrix, which is then applied to queries via matrix multiplication:

$$y_n = \Lambda_n^T q_n \tag{1}$$

The lambda layer approximates self-attention by introducing both content-aware and position-aware kernels. This enables the model to attend over a large spatial area without explicitly constructing full attention maps, reducing complexity to linear time and space. LambdaResNet26rpt_256 has approximately 11 million parameters and operates efficiently on 256×256 inputs.

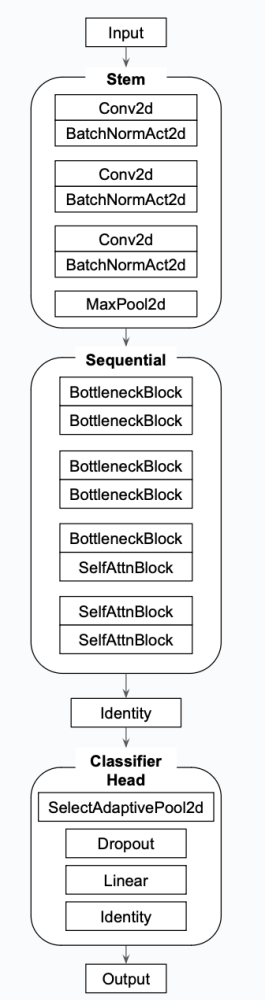

Figure 1: Architecture of LambdaResNet26rpt used in our experiments.

Swin Transformer V2 Small Swin Transformers partition images into non-overlapping windows and apply self-attention within these local contexts. To capture interactions across windows, the windows are shifted in alternating layers. Swin V2 builds upon this by introducing:

- Scaled cosine self-attention
- Logarithmically spaced continuous relative position bias
- Enhanced numerical stability for training at higher resolution

Our SwinV2 Small variant contains approximately 51 million parameters and uses four hierarchical stages, each containing multiple attention blocks. The final feature map is pooled and passed to a classification head.

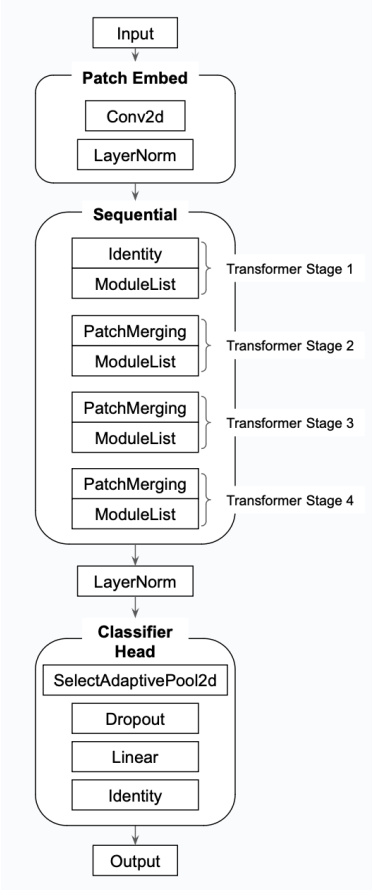

Figure 2: Architecture of Swin Transformer V2 Small used in our experiments.

3.4 Experimental Setup

Hardware and Environment. All training and evaluation was conducted locally using an NVIDIA CUDA-enabled GPU. The implementation was built using PyTorch and the `timm` library for model management. Development was done in Jupyter Notebooks and occasionally on Google Colab for testing and visualization.

Model Training Procedure. We trained each model on mel-spectrograms generated from 5-second audio clips. Files were loaded from disk as precomputed 256×256 tensors (3 channels) using a custom `MelCSVDataset`. Data was split into 60% training, 20% validation, and 20% testing using stratified sampling. We trained for 5 epochs per model, selecting checkpoints based on the lowest validation loss.

Hyperparameters. Both models used the AdamW optimizer and CrossEntropyLoss. Specific settings were:

- LambdaResNet26rpt_256: batch size = 32, learning rate = 1×10^{-5}
- Swin Transformer V2 Small: batch size = 20, learning rate = 1×10^{-4}

No learning rate scheduler or weight decay was applied.

3.5 Evaluation Metrics

We evaluate our models using five standard binary-classification metrics: precision, recall, F1-score, specificity, and area under the receiver operating characteristic curve (AUC–ROC). Let TP, FP, TN, and FN denote the numbers of true positives, false positives, true negatives, and false negatives, respectively.

- **Precision** measures the proportion of positive predictions that are actually correct:

$$\text{Precision} = \frac{TP}{TP + FP}\,.$$

- **Recall** (also known as sensitivity or true positive rate) quantifies the proportion of actual positives that are correctly identified:

$$\text{Recall} = \frac{TP}{TP + FN}\,.$$

- **F1 Score** is the harmonic mean of precision and recall, providing a single measure that balances both:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\,TP}{2\,TP + FP + FN}\,.$$

- **Specificity** (true negative rate) captures the proportion of actual negatives correctly classified:

$$\text{Specificity} = \frac{TN}{TN + FP}\,.$$

- **AUC–ROC** is the area under the receiver operating characteristic curve, which plots the true positive rate against the false positive rate $\big(FPR = FP/(FP+TN)\big)$ at various threshold settings. AUC–ROC summarizes the model's ability to distinguish between classes across all thresholds:

$$\text{AUC–ROC} = \int_0^1 \text{TPR}\big(FPR^{-1}(t)\big)\,dt.$$

3.6 Algorithm Box

Algorithm 1: AI-Generated Music Detection Pipeline

Input: CSV files `real_songs.csv`, `fake_songs.csv`
Output: Trained-model checkpoints; test-set metrics

```
Compute mel-spectrograms from mp3 files
foreach folder in {real, fake} do
    process_melspectrograms(folder, folder_mel)

Create stratified train/val/test splits
create_splits(real_mel, fake_mel, splits)

Train LambdaResNet and SwinV2 models
foreach modelName in {lambda_resnet26rpt, swinv2_small} do
    model = timm.create_model(modelName)
    optimizer = AdamW
    criterion = CrossEntropy
    bestValLoss = ∞
    for epoch = 1 to E do
        trainLoss = train_epoch(model, trainLoader)
        (valLoss, valAcc) = validate_epoch(model, valLoader)
        if valLoss < bestValLoss then
            bestValLoss = valLoss
            save_checkpoint(model, best_model_modelName)

Evaluate best model on test set
load_model(best_model_lambda_resnet26rpt)
evaluate(testLoader)
```

4 Results

Both models achieved extremely high accuracy on the binary classification task. Swin Transformer V2 Small outperformed LambdaResNet26rp_256 slightly in every metric, achieving a perfect AUC-ROC of 1.000. LambdaResNet still delivered strong performance with a much smaller parameter footprint. Table 1 summarizes the performance of both models on the test set:

Table 1: Performance metrics on the test set

Model	Precision	Recall	F1 Score	Specificity	AUC-ROC
LambdaResNet26rp_256	0.9910	0.9925	0.9918	0.9910	0.9996
Swin Transformer V2 Small	**0.9995**	**0.9990**	**0.9992**	**0.9995**	**1.0000**

Figures 3 and 4 show the validation accuracy and loss curves for both models across training epochs.

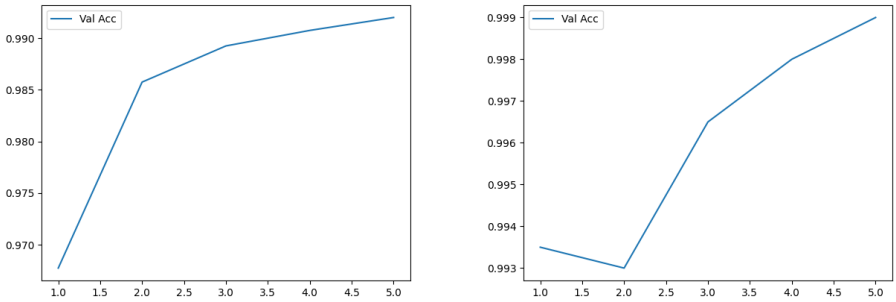

Figure 3: Validation accuracy curves for LambdaResNet (left) and Swin Transformer V2 (right)

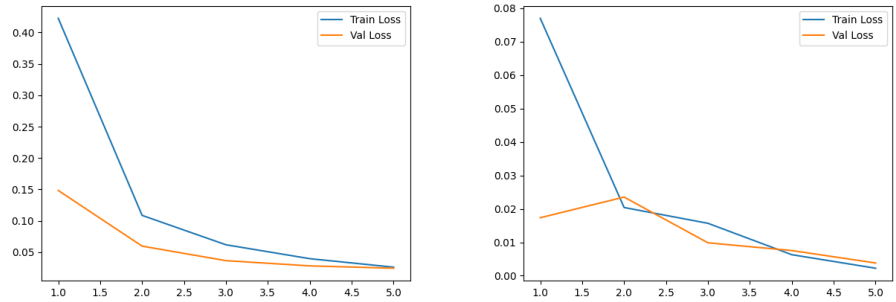

Figure 4: Training and validation loss curves for LambdaResNet (left) and Swin Transformer V2 (right)

Both models exhibited outstanding classification performance, with Swin Transformer V2 Small slightly outperforming LambdaResNet26rpt_256 across all metrics. As shown in Table 1, Swin Transformer achieved perfect AUC-ROC (1.000) and an F1 score of 0.9992, while LambdaResNet delivered strong results with an F1 score of 0.9918 and AUC-ROC of 0.9996. These results confirm that both models can effectively identify AI-generated music using only 5-second mel-spectrograms.

LambdaResNet is notably efficient, with only $\sim$11 million parameters and faster training and inference time. It completed 5 epochs in approximately 77 minutes at 1.62 it/s. In contrast, Swin Transformer, with $\sim$51 million parameters, trained more slowly at 1.76 it/s and required more GPU memory, but consistently converged to higher accuracy. Despite this, the performance gap is relatively narrow, making LambdaResNet an attractive option for real-time or resource-constrained deployment.

Precision and specificity were balanced in both models, indicating low false positive rates—critical in practical detection systems where falsely labeling human music as AI-generated is undesirable. Training dynamics, as shown in the accuracy and loss curves, further highlight this difference: Swin Transformer's accuracy improved steadily and reached 0.999 by epoch 5, with validation loss closely following training loss throughout, indicating strong generalization. LambdaResNet showed more fluctuation early in training, with a slight rise in validation loss at epoch 2, but ultimately achieved excellent performance, converging smoothly by epoch 5. These trends confirm that both models learned efficiently from limited data and that the lambda layer is a highly effective inductive bias for global spectrogram features.

Overall, the Swin Transformer is ideal for high-accuracy scenarios, while LambdaResNet offers a lightweight and efficient alternative with only marginal trade-offs in performance.

5 Conclusion

5.1 Takeaways

Overall, our results suggest that detecting AI-generated music from short 5-second clips is highly effective using compact vision models trained on mel-spectrograms. Both LambdaResNet26rpt_256 and Swin Transformer V2 Small demonstrated outstanding performance, with Swin achieving perfect AUC-ROC and LambdaResNet offering strong results with significantly lower compute demands. This supports the idea that pretrained image models—even without temporal modeling or domain-specific customization—can be repurposed for generative audio detection. Furthermore, the success of lambda layers as global context modules suggests they are a lightweight yet expressive alternative to full self-attention, especially in constrained environments. The use of pretrained vision models provides a practical path toward scalable and accurate music authenticity verification.

5.2 Limitations and Future Directions

Our current models were trained and evaluated solely on the SONICS dataset, which, while diverse, may not cover the full range of generative techniques or musical styles found in real-world settings. Future work could incorporate additional datasets—such as Jukebox, MusicCaps, or FMA—to improve generalizability and robustness across genres and synthesis methods.

In addition, our models were trained only for binary classification. Extending this to multi-class classification (e.g., real, Suno, MusicGen, Riffusion) could provide deeper insight into generative source attribution. Our current pipeline also lacks explainability mechanisms, such as saliency maps or Grad-CAM on spectrograms, which could be useful for debugging or gaining trust in deployment scenarios.

Although our models performed well on 5-second segments, this restriction was due to resolution compatibility with pretrained image models. All spectrograms were cropped to 256×256, limiting the temporal scope of analysis. Future work could explore models that accept higher-resolution or sequential spectrograms to model longer musical structures. Lastly, our dataset included only 20,000 clips, which limits robustness under domain shift. Scaling the dataset—either via additional real examples, broader generative sources, or synthetic augmentation—would likely improve generalization and deployment reliability.

References

[1] Md. Awsafur Rahman, Zaber Ibn Abdul Hakim, Najibul Haque Sarker, Bishmoy Paul, and Shaikh Anowarul Fattah. SONICS: Synthetic or not – identifying counterfeit songs. *arXiv:2408.14080*, 2024.

[2] Yuankun Xie, Jingjing Zhou, Xiaolin Lu, et al. FSD: An initial Chinese dataset for fake song detection. In *ICASSP*, IEEE, 2024.

[3] Yongyi Zang, Jiatong Shi, You Zhang, et al. CtrSVDD: A benchmark dataset and baseline analysis for controlled singing-voice deepfake detection. *arXiv:2406.02438*, 2024.

[4] Yongyi Zang, You Zhang, Mojtaba Heydari, and Zhiyao Duan. SingFake: Singing voice deepfake detection. In *ICASSP*, IEEE, 2024.

[5] Jee-weon Jung, Hee-Soo Heo, Hemlata Tak, et al. AASIST: Audio anti-spoofing using integrated spectro-temporal graph attention networks. In *ICASSP*, IEEE, 2022.

[6] Hemlata Tak, Jose Patino, Massimiliano Todisco, et al. End-to-end anti-spoofing with RawNet2. In *ICASSP*, IEEE, 2021.

[7] Hemlata Tak, Massimiliano Todisco, Xin Wang, et al. Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation. *arXiv:2202.12233*, 2022.

[8] Piotr Kawa, Marcin Plata, Michał Czuba, et al. Improved deepfake detection using Whisper features. *arXiv:2306.01428*, 2023.

[9] Yuan Gong, Yu-An Chung, and James Glass. AST: Audio Spectrogram Transformer. In *Interspeech*, 2021, pp. 571–575.

[10] Anmol Gulati, James Qin, Chung-Cheng Chiu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv:2005.08100*, 2020.

6 Statement about individual contributions

We are a group of 2 members, in which both members have contributed equally (50% each) to the report.