

FULLY FINE-TUNED MUSICFM WITH A LIGHTWEIGHT MLP HEAD FOR GLOBAL KEY DETECTION

Xinwei Jiang

Southern University of Science and Technology Great Bay University Dongguan, Guangdong, China
12633654@mail.sustech.edu.cn

Yaolong Ju

juyaolong@gbu.edu.cn

ABSTRACT

Musical key estimation provides essential harmonic context for Western tonal music, making it a foundational task in music information retrieval (MIR). In this paper, we present a key detection system built upon MusicFM-MSD, a self-supervised music foundation model. Specifically, We fine-tune the encoder end-to-end and attach a lightweight MLP head to predict global keys from mean-pooled frame representations. Our system achieves 72.19% exact accuracy and a 77.62% MIREX weighted score on the GiantSteps Key benchmark.

1. INTRODUCTION

Global key detection is standardly formulated as a 24-way classification task across major and minor keys [1]. Over the past decade, the field has transitioned from rule-based pitch-class profiling [2] and task-specific convolutional networks [3] toward self-supervised representation learning [4]. Extending this paradigm, we adapt MusicFM [5]—a self-supervised music foundation model—for global key estimation. Unlike typical linear probing approaches, we fully fine-tune the MusicFM-MSD encoder end-to-end alongside a lightweight MLP classification head.

2. TRAINING DATA

The training dataset uses the Hooktheory TheoryTab annotations [6] distributed with SheetSage [7], together with the GiantSteps MTG Keys extended collection [8, 9]. It comprises 22,915 training tracks and a separate validation split of 2,433 tracks used exclusively for checkpoint selection. Neither split is included in this submission package.

During data preprocessing, major/minor modes and enharmonic spellings are normalized into a sharp-spelled 24-class label space. Only tracks with valid audio waveforms are retained, omitting all chord text metadata to guarantee a strictly audio-only setup. To enhance generalization, data augmentation via pitch-shifting is applied during training, with key labels transposed accordingly.

MIREX Compliance. To prevent data leakage and ensure strict compliance with MIREX benchmarks, the official GiantSteps Key test set was strictly isolated from training, validation, and hyperparameter tuning. Although GiantSteps MTG Keys was incorporated into our training pool, any tracks overlapping with the original GiantSteps Key test set were explicitly excluded.

3. METHOD

3.1 MusicFM Encoder

We adopt MusicFM-MSD [5, 10] as our backbone architecture—a 12-layer Conformer [11] pretrained on the Million Song Dataset [12] via a masked token prediction objective. Input audio is downmixed to 24 kHz mono and transformed into a 128-band log-mel spectrogram (FFT size 2,048, hop size 240). Two residual convolutional layers downsample the spectrogram to a 25 Hz frame rate, which is then fed into the Conformer with a hidden dimension of 1024. We extract representations from the final layer (Layer 12) and aggregate them via temporal mean-pooling. Unlike frozen-backbone approaches, *all* encoder parameters are fine-tuned end-to-end during training.

3.2 Classification Head and Inference

The 1024-dimensional pooled vector is passed through a classification head comprising LayerNorm, a linear projection to 512 dimensions, GELU activation, dropout ($p = 0.1$), and a final 24-class classification layer. Target labels are standardized into a sharp-spelled format (C, C#, . . . , B for both *major* and *minor* modes), with all enharmonic equivalents mapped accordingly. The output is formatted as a single tab-delimited string (<tonic>\t<mode>).

During inference, audio recordings are evaluated using 12-second windows placed at equal intervals across the track. The predicted logits across these five windows are averaged, and the final key is determined via argmax selection over the 24 classes. This uniform windowing policy is strictly maintained across all experimental evaluations reported in this work.

3.3 Optimization

Training uses random 12-second excerpts and optimizes a 24-way cross-entropy loss with AdamW under bfloat16 autocast. Following the MusicFM fine-tuning recipe, the pretrained encoder and the classification head use separate



Hyperparameter	Value
Crop length	12 s
Batch size	64
Epochs	3
Optimizer	AdamW
Encoder / head LR	10^{-5} / 10^{-4}
Weight decay	10^{-4}
LR schedule	5% linear warmup, cosine decay
Max gradient norm	1.0
Precision	bfloat16 autocast
Head dropout	0.1
Augmentation	pitch shift $-4 \dots +4$ semitones, with key transposition
Random seed	42

Table 1. Training configuration of the submitted checkpoint. Gradient checkpointing is enabled in the Conformer to fit batch size 64 on a single GPU.

System	Accuracy (%)	MIREX (%)
MusicFM+MLP (ours)	72.19	77.62
AllConv (retrained) [3]	65.73	72.76
madmom [13]	67.88	74.62
S-KEY (1M) [4]	64.40	72.09
KeyFinder [2]	60.43	69.04

Table 2. Results on the 604-excerpt GiantSteps Key test set. The MIREX score weights exact matches by 1.0, perfect fifths by 0.5, relative major/minor by 0.3, and parallel major/minor by 0.2 [1].

learning rates. The released checkpoint was selected based on the validation MIREX score. Table 1 details the hyperparameter configuration used for this submission.

4. RESULTS ON GIANTSTEPS KEY

Table 2 reports performance on the official GiantSteps Key test set (604 excerpts of electronic dance music). All systems were evaluated on the same excerpts with a single whole-file prediction per excerpt. The fully fine-tuned system reaches 72.19% exact accuracy and a 77.62% MIREX weighted score, outperforming all compared baselines, including the profile-based KeyFinder [2], the supervised madmom key detector [13], the self-supervised S-KEY (1M) [4], and a CNN trained from scratch on log-frequency spectrograms [3]. The performance gain is most pronounced in exact accuracy, demonstrating that fine-tuning the foundation model yields more reliable tonic and mode predictions than traditional profile-based or frozen-feature methods.

5. RESOURCE REQUIREMENTS

Model. The system contains approximately 330M parameters; the released checkpoint is 1.3 GB and includes both the fine-tuned encoder and the MLP head.

Inference. Inference requires less than 6 GB of GPU VRAM and about 4 GB of CPU memory. On an NVIDIA V100, a single 2-minute track takes 0.2–0.7

s after GPU warm-up, requiring approximately 3 minutes for the full 604-excerpt benchmark in batch mode. We use bfloat16 autocast on supported GPUs and float32 otherwise. A CPU-only fallback (`predict.py -cpu -threads N`) is also supported; using 32 threads, processing takes 3 s per track and 30–40 minutes for the entire dataset. The implementation relies on PyTorch’s native scaled dot-product attention, removing any external dependence on `flash-attn`.

Training. The checkpoint was trained for 3 epochs on 22,915 tracks using a single NVIDIA H100 with bfloat16 autocast and gradient checkpointing.

The submission package includes the trained weights, inference entry point (`predict.py`), training scripts, and a README with build and execution instructions.

6. CONCLUSION

In this paper, we presented an audio-only key detection framework that fully fine-tunes the MusicFM-MSD foundation model coupled with a lightweight MLP head. Moving beyond traditional handcrafted pitch-class profiles and frozen-feature probing, our results on the GiantSteps Key benchmark (72.19% exact accuracy and a 77.62% MIREX score) demonstrate that end-to-end adaptation of self-supervised music representations substantially enhances global harmonic estimation. Remarkably, this performance is achieved without task-specific architectural modifications or tonal heuristics, highlighting both the expressiveness of general-purpose music encoders and the practical feasibility of lightweight deployment. Future work will explore extending this fine-tuning recipe to multi-task MIR settings such as joint key and chord recognition.

7. REFERENCES

- [1] MIREX, “MIREX 2026: Audio key detection,” https://music-ir.org/mirex/wiki/2026:Audio_Key_Detection, 2026, accessed 2026-09-22.
- [2] I. Sha’ath, “Estimation of key in digital music recordings,” Birkbeck, University of London, MSc Computer Science Project Report, 2011. [Online]. Available: <https://www.ibrahimshaath.co.uk/keyfinder/KeyFinder.pdf>
- [3] F. Korzeniowski and G. Widmer, “Genre-agnostic key classification with convolutional neural networks,” in *Proceedings of ISMIR*, 2018, pp. 264–270.
- [4] Y. Kong, G. Meseguer-Brocal, V. Lostanlen, M. Lagrange, and R. Hennequin, “S-KEY: Self-supervised learning of major and minor keys from audio,” in *Proceedings of ICASSP*. IEEE, 2025, pp. 1–5.
- [5] M. Won, Y.-N. Hung, and D. Le, “A foundation model for music informatics,” in *Proceedings of ICASSP*. IEEE, 2024, pp. 1226–1230.

- [6] Hooktheory, “TheoryTab: Tabs that show the theory behind songs,” <https://www.hooktheory.com/theorytab>, accessed 2026-09-29.
- [7] C. Donahue, J. Thickstun, and P. Liang, “Melody transcription via generative pre-training,” in *Proceedings of ISMIR*, 2022, pp. 485–492. [Online]. Available: <https://github.com/chrisdonahue/sheetsage>
- [8] P. Knees, Á. Faraldo, P. Herrera, R. Vogl, S. Böck, F. Hörschläger, and M. Le Goff, “Two data sets for tempo estimation and key detection in electronic dance music annotated from user corrections,” in *Proceedings of ISMIR*, Málaga, Spain, 2015, pp. 364–370.
- [9] GiantSteps Project, “GiantSteps MTG keys: An extension of the GiantSteps key dataset,” <https://github.com/GiantSteps/giantsteps-mtg-key-dataset>, accessed 2026-09-22.
- [10] M. Won, Y.-N. Hung, and D. Le, “MusicFM: Official model repository,” <https://github.com/minzwon/musicfm>, musicFM-MSD checkpoint documentation; accessed 2026-09-29.
- [11] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, “Conformer: Convolution-augmented transformer for speech recognition,” in *Proceedings of Interspeech*, 2020, pp. 5036–5040.
- [12] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” in *Proceedings of ISMIR*, 2011. [Online]. Available: <https://ismir2011.ismir.net/papers/OS6-1.pdf>
- [13] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, “madmom: A new python audio and music signal processing library,” in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2016, pp. 1174–1178.