

MIREX 2026: KEY DETECTION VIA LLM PROCESSING OF CHORD RECOGNIZER OUTPUTS

Jeffrey Gordon

New York University

jeffrey.gordon@nyu.edu

ABSTRACT

The harmony of a song is closely tied to its key, and human musicians often consider a song’s chords when determining its key. We present a key detection system inspired by this, prompting a large language model with the output of a chord recognizer, which requires no additional training. On FMAKv2, this approach outperforms the deep learning baseline AllConv (0.68 vs. 0.66 accuracy). Analyzing where the two systems differ, we build a hybrid system that combines predictions, falling back to AllConv when the LLM’s predictions are likely to be untrustworthy. Because the hybrid was tuned and tested on the same data, we submit it to MIREX 2026 for independent evaluation.¹

1. INTRODUCTION

Key detection remains a challenging music information retrieval task due to the limited availability of labeled data. Key detectors also suffer from comparatively few opportunities for learning — a 60 second audio clip may have dozens of chord changes but only a single key. To sidestep these issues, researchers have recently explored large-scale self-supervised methods for key detection, which do not require labeled data to learn [1, 2].

Another solution to this problem is to take advantage of the representations learned by supervised methods for other MIR tasks which are less bottlenecked by data availability. The hope is that these intermediate representations might provide useful information for key detection. Chord recognizers hold promise in this regard, not only because chord annotations provide more learning opportunities, but also because a song’s chords are highly correlated with its key. Indeed, when humans determine a song’s key, they often listen for *V-I* cadences or repeating harmony.

We can take advantage of the fact that chord recognizers output text to predict key from chords with a large language model. Commercial LLMs which have encountered music theory texts during pre-training are particularly well-suited to this task.

¹ Code: github.com/jeffgord/mirex2026-chord-llm (MIREX submission) and github.com/jeffgord/llm-key-detection (experiments).

2. METHOD

The proposed method first runs a song’s audio through a Chord Recognizer. Our implementation uses the model from [3]. We then feed the output to an LLM after some light pre-processing. The experimental results reported here use *Gemini 3.1 Flash Lite*². After noticing high variance in predictions on ambiguous cases, however, we opted for the more powerful *Gemini 3.8 Flash*² in the final submission. The prompt looks roughly like this:

```
Determine the key of this song based
off its chords:
{CHORDS}
```

```
The chords came from a chord
recognizer, and therefore may contain
mistakes or confusing enharmonic
spellings. Also, because the passage
may have been trimmed, you should not
assume that it starts or ends at a
phrase boundary.
```

The setup described above constitutes the system’s core prediction method. We further extend this system for robustness to common failure modes. For instance, the above method can fail on chord progressions which are diatonic to multiple keys. To remedy this, we also include chroma activations alongside the chord representation. We compute these activations by averaging a song’s chromagram across the entire passage, inspired by the Krumhansl-Schmuckler key-finding algorithm [4].

There are also situations where the chord representation is wholly uninformative for key prediction. For instance, chord recognizers may fail to produce useful output for a song with a solo vocalist. We detect these situations by calculating the proportion of the song’s duration for which the chord recognizer labeled the harmony as *no chord*. When this proportion rises above a threshold, we treat the LLM’s key prediction as untrustworthy. We incorporate a fallback deep learning method for this situation, using its predictions instead of the LLM’s. In particular, we use the AllConv model [5], as implemented in madmom [6].

Further, a thorough error analysis revealed that the LLM is prone to making parallel key errors (see Figure 1). We hypothesize that this is due to chord recognizer issues that propagate to the LLM. For instance, the chord recognizer tends to assign a major or minor quality to chords with ambiguous thirds. These tenuously-assigned chord labels can heavily influence the prediction made by the LLM.

² Model cards: deepmind.google/models/model-cards



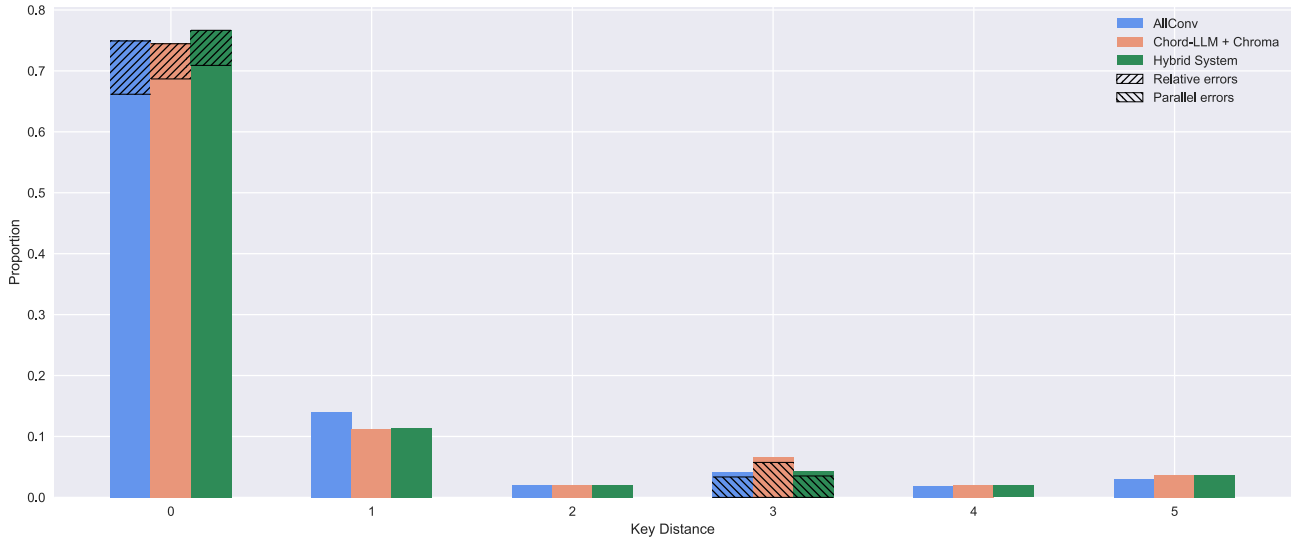


Figure 1. Predictions made by different methods on FMAKv2, stratified by key distance (the number of notes shared by the key signatures of the predicted and actual key). While the Chord-LLM + Chroma system makes more correct predictions than AllConv, it is prone to parallel errors. The hybrid system explicitly corrects for this.

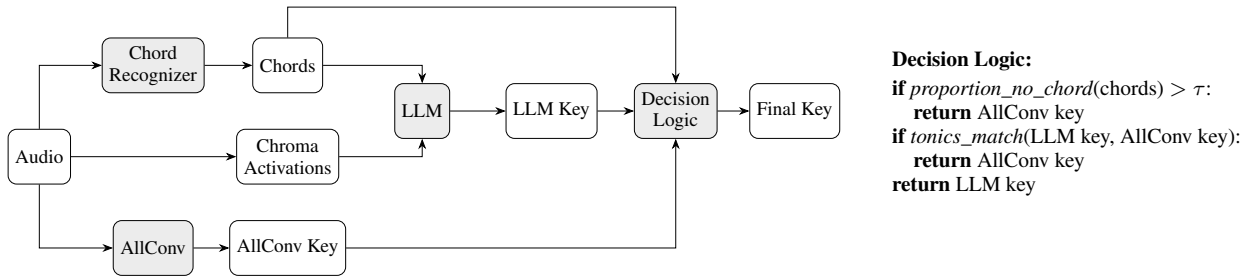


Figure 2. Overview of the hybrid system. Data (white) flows through processors (gray). Pseudocode for the Decision Logic processor is shown at right.

Method	Accuracy	Avg. MIREX
AllConv (baseline)	0.6617	0.7414
Chord-LLM	0.6846	0.7490
+ Chroma Activations	0.6868	0.7515
+ Parallel Correction	0.7091	0.7700
+ No Chord Threshold	0.7151	0.7760

Table 1. Performance on FMAKv2 as each component is added to the system. MIREX score punishes musically plausible errors less heavily than raw accuracy.

To reduce the amount of parallel key errors, we compare the LLM’s prediction with AllConv’s, choosing AllConv’s prediction as the final output when both tonics match.

This chord-LLM/AllConv hybrid system constitutes our submission to MIREX. The architecture as described here is pictured in Figure 2. We report metrics on FMAKv2 [1, 7] in Table 1.

3. DATASETS

This hybrid system does not require further training. However, it was iteratively improved using insights from the FMAKv2 dataset, and therefore the reported metrics (Ta-

ble 1) should not be taken as a measure of real world performance.

Data used for training the system’s components should also be considered training material. The chord recognizer was trained on songs from the Isophonics, Billboard, and MARL datasets [3]. The AllConv model used as a fallback was trained on GiantSteps MTG, Billboard, and an unpublished classical music dataset [5]. Note that the GiantSteps MTG set specifically excludes the 604 GiantSteps tracks used for evaluation in the MIREX competition.

Additionally, because we rely on Google’s commercial AI API, we cannot be certain exactly what data was used to train the LLM. The prompt includes only derived chords and chromagrams, so it is highly unlikely that the LLM has seen this data before.

4. RESULTS

We leave thorough evaluation of this system to the 2026 MIREX competition.

5. AI USAGE STATEMENT

The author discloses the use of Claude Code as a pair programmer in building the described system and running ex-

117 periments. In addition, Claude Code was used to help pre-
118 pare aspects of the manuscript, though the actual text is
119 almost entirely human written with only light AI usage for
120 spelling/grammar checks.

121 6. REFERENCES

- 122 [1] Y. Kong, V. Lostanlen, G. Meseguer-Brocal, S. Wong,
123 M. Lagrange, and R. Hennequin, “STONE: Self-
124 supervised tonality estimator,” in *Proc. of the 25th Int.*
125 *Society for Music Information Retrieval Conf.*, 2024.
- 126 [2] Y. Kong, G. Meseguer-Brocal, V. Lostanlen, M. La-
127 grange, and R. Hennequin, “S-KEY: Self-supervised
128 learning of major and minor keys from audio,” in
129 *ICASSP 2025 - 2025 IEEE Int. Conf. on Acoustics,*
130 *Speech and Signal Processing (ICASSP)*. IEEE, 2025.
- 131 [3] J. Jiang, K. Chen, W. Li, and G. Xia, “Large-
132 vocabulary chord transcription via chord structure de-
133 composition,” in *Proc. of the 20th Int. Society for Mu-*
134 *sic Information Retrieval Conf.*, Delft, The Nether-
135 lands, 2019, pp. 644–651.
- 136 [4] D. Temperley, “What’s key for key? The Krumhansl-
137 Schmuckler key-finding algorithm reconsidered,” *Mu-*
138 *sic Perception*, vol. 17, no. 1, pp. 65–100, 1999.
- 139 [5] F. Korzeniowski and G. Widmer, “Genre-agnostic key
140 classification with convolutional neural networks,” in
141 *Proc. of the 19th Int. Society for Music Information Re-*
142 *trieval Conf.*, Paris, France, 2018.
- 143 [6] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and
144 G. Widmer, “madmom: A new Python audio and music
145 signal processing library,” in *Proc. of the 24th ACM Int.*
146 *Conf. on Multimedia*, Amsterdam, The Netherlands,
147 2016, pp. 1174–1178.
- 148 [7] S. Wong and G. Hernandez, “FMAK: A dataset of key
149 and mode annotations for the free music archive,” in
150 *Extended Abstracts for the Late-Breaking Demo Ses-*
151 *sion of the 24th Int. Society for Music Information Re-*
152 *trieval Conf.*, Milan, Italy, 2023.