From Isolated Feature to Orbits: Discovering Music Concepts via Multi-SAE Alignment

Liwei Lin
Computer Science Department, NYU Shanghai
Machine Learning Department, MBZUAI
liwei.lin@nyu.edu

Gus Xia
Machine Learning Department, MBZUAI
Gus.Xia@mbzuai.ac.ae

Abstract

How can we understand what a music foundation model has learned *internally*? Most interpretability approaches, such as probing and Sparse Autoencoders (SAEs), focus on identifying individual features with minimal structural assumptions. We argue that many concepts are better understood as *structured relations* rather than isolated features. This is especially prominent in music, where tonal structures are organized in the space of pitch and time. For example, concepts such as chords or keys are naturally expressed as structured sets (e.g., the 12 transpositions of a chord or the diatonic system within a key), rather than isolated features. In this study, **we shift from feature identification to structure-based analysis**, asking whether the learned inner representations of music foundation model emerge as organized structures over features. To this end, we introduce a framework that uses pitch transposition as an inductive bias to induce ordered orbits via multi-view SAE alignment. Concretely, we generate pitch-shifted input pairs and align their SAE representations to discover structured groups of pitch-related features. Experimental results show that this approach recovers orbit structures corresponding to chords, keys, and melodic patterns across two state-of-the-art music foundation models, while requiring only minimal grounding (e.g., a few anchor examples) to interpret entire concept families.

1 Introduction

Pretrained music audio foundation models [1, 2, 3, 4, 5, 6] learn representations that support a wide range of downstream tasks [7], yet what musical information these representations encode and how such information is organized in the architecture or activations remain difficult to interpret.

Existing work has approached the interpretability of music audio representations from several directions. For self-supervised music audio models, their representation quality can be evaluated by downstream benchmarks across music understanding tasks [8], providing indirect evidence of what musical information their representations may capture. Probing and intervention studies [7, 9, 10, 11] have tested whether music audio generative models encode and use symbolic musical concepts. Recently, sparse autoencoders (SAEs) [12] have emerged as a promising tool for decomposing neural activations into sparse, potentially interpretable features [13]. In music, recent work has applied SAEs to generative models [14, 15], identifying features that can be linked to musical concepts.

However, these approaches mostly stop at individual concept-feature correspondences. Probing detects predefined concepts using labeled examples, while SAE-based analyses typically assign meanings post hoc by matching individual features to external labels. This is limiting in the music domain, where concepts are often organized in relational structures. Chords, for example, form a

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

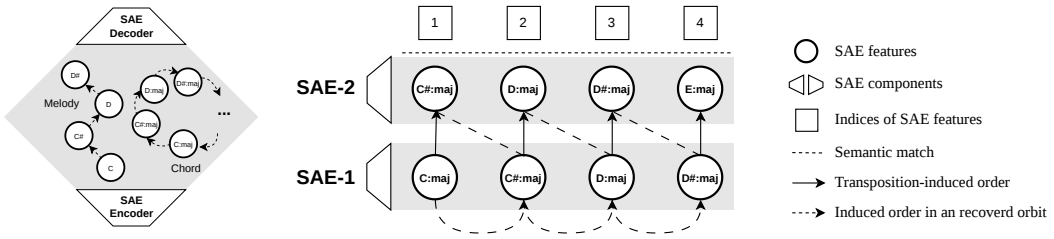


Figure 1: Recovered orbits over SAE features (left) and orbit recovery via two aligned SAEs (right). The i-th feature in SAE-2 is the one-semitone-up version of the i-th feature in SAE-1. We pair each feature i in SAE-2 with its semantically closest counterpart in SAE-1, measured by decoder vector similarity. This semantically matched feature in SAE-1 is defined to be the *successor* of the original i-th feature. Iteratively applying this successor relation recovers ordered pitch orbits.

cyclic ordered set under transposition along the pitch-class cycle[1]. If we can identify groups of related compound features with their relation structure such as

$$\textit{C major chord} \rightarrow \textit{C}\sharp \textit{ major chord} \rightarrow \textit{D major chord} \rightarrow \ldots,$$

then interpreting a small number of features in the group can help infer the semantics of the remaining features through that structure. Moreover, the invariant shared by all members of the group constitutes an abstract concept, in this case, *major chord*. Our goal is therefore to recover not only learned features, but also the relational structure among them.

Towards this goal, we focus on pitch transposition as a fundamental transformation in music: many pitch-specific music concepts transform equivariantly under transposition. For example, transposing a *C major chord* upward by one semitone yields a *C*$\sharp$ *major chord*, where the pitch concept component shifts but the chord quality concept component does not. We utilize this property of pitch transposition [16] to induce ordered **orbits** [17] over SAE features, thereby recovering pitch-related concepts and their relational structure, as illustrated in Figure 1 (left).

Our method constructs multiple pitch-shifted views of each sample, trains a dedicated SAE for each shift, and encourages views of the same sample to activate the same feature indices across SAEs via an alignment training strategy. Consequently, a feature in one SAE is expected to represent the one-semitone transposition of the same-index feature in the next SAE, e.g., from *C major chord* to *C*$\sharp$ *major chord*. As illustrated in Figure 1 (right), after training, we recover orbits by turning this cross-SAE transposition relation into a successor relation of feature indices within a single SAE. Once an orbit and its order are recovered, a few labeled anchors can propagate semantics to the remaining features, while the invariant shared across the orbit reveals the underlying abstract concept, e.g., *major chord*. In this way, interpretation shifts from isolated features to structured musical concepts.

Our contributions are as follows:[2]

- Conceptually, we extend SAE interpretability from the identification of individual features to structure-based analysis. This shift enables the interpretation of higher-level concepts, even when such concepts do not emerge as single SAE features.
- Methodologically, we introduce a music-informed self-supervised framework for recovering pitch-related feature orbits through multi-SAE alignment. The method is highly label-efficient: instead of requiring extensive labeled data and manual annotation of individual features during post-hoc analysis, it can use one or a few anchor examples to interpret an entire orbit of related features, and in some cases supports one-shot transfer to downstream understanding tasks.
- Empirically, we demonstrate the framework on two state-of-the-art music foundation models and recover musically meaningful orbits associated with chords, keys, and melodic structure. In particular, our experimental results provide new evidence on how these models encode key-related concepts.

[1] For readers unfamiliar with foundational music theory concepts (e.g., pitch classes, chords, pitch transposition, and the Circle of Fifths), we provide a brief primer in Appendix A.

[2] Source code is available at `https://github.com/Kikyo-16/sae-alignment`.

2 Related Work

2.1 Interpretability of Music Foundation Models

The rapid deployment of music foundation models has motivated a shift toward understanding their internal representational dynamics. Early efforts focused on linear probing, where supervised classifiers are trained on hidden activations to detect musical attributes such as genre, emotion, or pitch [7, 9, 10, 11]. While probing confirms that these models encode a wealth of musical information, it is fundamentally limited by polysemanticity, where individual neurons represent multiple unrelated concepts [18]. Mechanistic interpretability aims to disentangle these representations using Sparse Autoencoders (SAEs) [12]. Recent work [14] has applied SAEs to generative music models like MusicGen, discovering interpretable features such as timbre, texture, and stylistic markers. However, these studies focus on individual features, overlooking the structured relations present in polysemantic music concepts.

2.2 Evolution of SAE Architectures and Sparsity

The quality of discovered features is highly sensitive to the sparsity mechanism of the SAE. Traditional L_1-regularized autoencoders often suffer from the "shrinkage problem", where feature magnitudes are suppressed during reconstruction [13]. To address this, Top-K SAEs [12] have been introduced, which use an activation function that preserves exactly K latents, thereby improving the reconstruction-sparsity frontier. Our work employs the same Top-K mechanism.

2.3 SAE Alignment and Relational Interpretability

Our work is situated at the intersection of mechanistic interpretability and representational alignment. Recent frameworks like Universal Sparse Autoencoders (USAE) [19] jointly train dictionaries across multiple models to discover universal concept spaces. In a complementary direction, representational similarity metrics have been used to match features between models post-hoc [18]. We extend this by recovering entire orbits of pitch-related features through multi-view SAE alignment. This shifts the focus from identifying what a model has learned to recovering the relational structures that define its understanding of music.

3 Methodology

We induce ordered orbits over SAE features via pitch transposition. Specifically, we train multiple SAEs per music foundation model. Each SAE is trained on an adjacent pitch-shifted view of the dataset. We ensure that feature indices remain compatible across SAEs by synchronizing the top-K selection during SAE training. This treatment implies a zig-zag path through features across two SAEs, which is further compressed into a successor map in one SAE. Using this successor map as a generator then identifies orbits or partial orbits consisting of SAE features.

3.1 Transposition-Induced Orbits of Music Concepts

Let $\mathcal{C}$ denote the space of musical concepts, and let $\mathsf{T}_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ denote transposition upward by one semitone. Repeatedly applying $\mathsf{T}_{\mathcal{C}}$ to a concept $c \in \mathcal{C}$ induces its orbit:

$$\mathrm{Orb}(c) = \left\{ \mathsf{T}_{\mathcal{C}}^{i}(c) \mid i \in \mathbb{Z} \right\} \subseteq \mathcal{C}. \tag{1}$$

The behavior of a concept under pitch transposition gives rise to two cases. A *pitch-invariant concept*, such as instrumentation or a rhythmic pattern, remains unchanged under transposition:

$$\mathsf{T}_{\mathcal{C}}(c) = c, \qquad \mathrm{Orb}(c) = \{c\}. \tag{2}$$

Such a concept is a fixed point of pitch transposition, and its orbit is a singleton.

In contrast, a *pitch-specific concept*, such as a melody, chord, or key, transforms equivariantly under pitch transposition. Repeated application of the transposition operator produces a non-trivial ordered orbit:

$$\mathsf{T}_{\mathcal{C}}(c) \neq c, \qquad c \to \mathsf{T}_{\mathcal{C}}(c) \to \mathsf{T}_{\mathcal{C}}^{2}(c) \to \cdots. \tag{3}$$

For pitch-class concepts in twelve-tone equal temperament, this orbit is typically cyclic with period twelve. For example,

$$\text{C major chord} \to \text{C}\sharp \text{ major chord} \to \text{D major chord} \to \cdots \to \text{B major chord} \to \text{C major chord}. \tag{4}$$

Along this orbit, the pitch component shifts while the chord-quality component remains invariant. The invariant shared across the orbit constitutes the underlying abstract concept, in this case, *major chord*.

3.2 Multi-SAE Alignment with Shared Top-K Support

Before recovering transposition-induced orbits within a single SAE, we first establish an index-wise correspondence across multiple SAEs trained on pitch-shifted views of the same audio.

Let $\mathcal{A}$ denote the audio space, and let $\mathsf{T}_{\mathcal{A}}$ denote upward transposition by one semitone. Given an audio segment $a \in \mathcal{A}$, we construct S adjacent pitch-shifted views:

$$a^{(s)} = \mathsf{T}_{\mathcal{A}}^{s}(a), \qquad s = 0, \ldots, S-1. \tag{5}$$

A frozen music foundation model H extracts a sequence of frame-level representations from each view, yielding a matrix

$$X^{(s)} = \left[x_1^{(s)}, x_2^{(s)}, \ldots, x_L^{(s)}\right]^\top = H\left(a^{(s)}\right) \in \mathbb{R}^{L \times d}, \tag{6}$$

where L is the number of frames and d is the representation dimensionality, and $x_t^{(s)} \in \mathbb{R}^d$ denotes the representation at frame t. Because pitch transposition preserves temporal alignment, representations at the same frame position across the S views form a group of corresponding transformed examples.

For each pitch-shifted view s, we instantiate a Top-K SAE whose latent feature set is

$$\mathcal{F}^{(s)} = \left\{f_1^{(s)}, \ldots, f_M^{(s)}\right\}, \tag{7}$$

where all SAEs have the same latent dimensionality M, while their encoder and decoder weights are learned independently. They additionally share a learnable pre-encoder bias $b \in \mathbb{R}^d$, which provides a common center for all pitch-shifted representation spaces. For clarity, we omit the frame index t below. Given a representation $x^{(s)} \in \mathbb{R}^d$, the s-th SAE produces the non-negative pre-sparsification activation

$$u^{(s)} = \mathrm{ReLU}\left(W_{\mathrm{enc}}^{(s)}\left(x^{(s)} - b\right)\right) \in \mathbb{R}^M_{\geq 0}. \tag{8}$$

If Top-K selection were performed independently for each view, the S SAEs could assign arbitrary latent indices to corresponding concepts. We instead perform support selection jointly across all transformed views.

Because activation scales may differ across SAEs, we first normalize each activation vector by its root-mean-square magnitude:

$$\alpha^{(s)} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(u_i^{(s)}\right)^2}, \qquad \bar{u}^{(s)} = \frac{u^{(s)}}{\max\left(\epsilon, \alpha^{(s)}\right)}. \tag{9}$$

Here, $\epsilon > 0$ is a small constant used for numerical stability; the same ϵ is used for decoder normalization later.

We then compute an index-wise shared score by taking the maximum normalized activation across the view dimension:

$$u_i^{\mathrm{shared}} = \max_{s \in \{0,\ldots,S-1\}} \bar{u}_i^{(s)}, \qquad i = 1, \ldots, M. \tag{10}$$

The K largest shared scores define a common binary support mask:

$$m = \mathrm{TopKMask}\left(u^{\mathrm{shared}}, K\right), \qquad m \in \{0,1\}^M, \qquad \sum_{i=1}^{M} m_i = K. \tag{11}$$

The same mask is applied to the original, unnormalized activations of every SAE:

$$z^{(s)} = m \odot u^{(s)}, \qquad s = 0, \ldots, S-1. \tag{12}$$

The normalization in Equation 9 therefore affects only which indices enter the shared support; the view-specific activation magnitudes used for reconstruction are preserved.

Each SAE reconstructs the representation associated with its own pitch view:

$$\widehat{x}^{(s)} = W_{\text{dec}}^{(s)} z^{(s)} + b. \tag{13}$$

The shared pre-encoder bias is subtracted before encoding and restored after decoding, so all SAEs model deviations from the same learned center. The S SAEs are jointly optimized using the mean reconstruction loss

$$\mathcal{L}_{\text{rec}} = \frac{1}{S} \sum_{s=0}^{S-1} \mathbb{E}_{a,t} \left[\left\| \widehat{x}_t^{(s)} - x_t^{(s)} \right\|_2^2 \right]. \tag{14}$$

After each parameter update, every decoder column is normalized to unit norm. Writing

$$W_{\text{dec}}^{(s)} = \left[w_1^{(s)}, \ldots, w_M^{(s)} \right], \tag{15}$$

we apply

$$w_i^{(s)} \leftarrow \frac{w_i^{(s)}}{\max\left(\epsilon, \left\| w_i^{(s)} \right\|_2\right)}. \tag{16}$$

This prevents arbitrary scaling of decoder vectors and enables their directions to be compared after training.

How shared Top-K support encourages alignment. At initialization, the shared mask imposes only index-level co-selection: all S SAEs reconstruct each group of pitch-shifted representations using the same latent indices. During training, whenever index i is selected, its encoder and decoder parameters across the SAEs are jointly updated on temporally aligned examples related by pitch transposition. This couples the otherwise independent permutations of the SAE dictionaries and encourages same-index features $f_i^{(s)}$ and $f_i^{(s+1)}$ to encode adjacent transposed realizations of a shared concept family. However, this alignment remains an inductive bias rather than a guarantee, since reconstruction ambiguity, polysemanticity, and imperfect pitch equivariance may still produce mismatches. We therefore use a simple algorithm to recover the learned structures after training.

3.3 Recovering Rings and Sequences

After training, we recover transposition-induced structures directly from the learned decoder dictionaries, without using input examples or labels. The first step is to construct a successor relation within a reference SAE, such that the successor of a feature represents its one-semitone-up counterpart.

We first consider two adjacent models, **SAE-0** and **SAE-1**, and take **SAE-0** as the reference space. Shared-support alignment encourages the same feature index across the two SAEs to encode adjacent transpositions. Starting from $f_i^{(0)}$, we therefore first move to the same-index feature $f_i^{(1)}$, which approximates its one-semitone-up realization, and then match it back to a feature in SAE 0:

$$f_i^{(0)} \xrightarrow{\text{alignment}} f_i^{(1)} \xrightarrow{\text{semantic match}} f_j^{(0)}. \tag{17}$$

The semantic match is computed via two constraints. Using the unit-normalized decoder vectors, we define the confidence of the candidate transition $f_i^{(0)} \to f_j^{(0)}$ as

$$v_{\uparrow}^{(0)}(i,j) = \cos\left(w_i^{(1)}, w_j^{(0)} \right). \tag{18}$$

The tentative successor of $f_i^{(0)}$ is the feature with the highest confidence:

$$j^\star(i) = \arg\max_j v_{\uparrow}^{(0)}(i,j). \tag{19}$$

The transition is accepted only if $v_{\uparrow}^{(0)}(i, j^\star(i)) \geq \tau$; otherwise, the successor of $f_i^{(0)}$ is left undefined.

When at least three SAEs are available, we take an interior SAE s, where $0 < s < S-1$, as the reference space. A candidate transition $f_i^{(s)} \to f_j^{(s)}$ can then be evaluated from both neighboring pitch shifts. We define

$$v_{\uparrow}^{(s)}(i,j) = \cos\left(w_i^{(s+1)}, w_j^{(s)}\right), \tag{20}$$

$$v_{\downarrow}^{(s)}(i,j) = \cos\left(w_i^{(s)}, w_j^{(s-1)}\right). \tag{21}$$

Here, $v_{\uparrow}^{(s)}(i,j)$ measures whether the upward-transposed source matches the candidate destination, while $v_{\downarrow}^{(s)}(i,j)$ measures whether the downward-transposed destination matches the source.

We select the destination using the mean of the two directional similarities:

$$j^\star(i) = \arg\max_j \frac{v_{\uparrow}^{(s)}(i,j) + v_{\downarrow}^{(s)}(i,j)}{2}. \tag{22}$$

For the selected destination, its confidence is given by the stronger of the two directional matches:

$$v^{(s)}(i) = \max\left\{v_{\uparrow}^{(s)}(i, j^\star(i)), v_{\downarrow}^{(s)}(i, j^\star(i))\right\}. \tag{23}$$

Accept the transition if $v^{(s)}(i) \geq \tau$; otherwise, reject it. In our experiments, we use $S = 3$ and take the middle SAE, $s = 1$, as the reference space, which is the minimum setup for double-direction evaluation.

The accepted transitions initially define a partial successor relation. When multiple transitions point to the same feature, we retain only the transition with the highest confidence and reject the others. The remaining transitions define the feature-space transposition map

$$\mathsf{T}_{\mathcal{F}}\left(f_i^{(s)}\right) = f_{j^\star(i)}^{(s)}. \tag{24}$$

Repeated application of $\mathsf{T}_{\mathcal{F}}$ produces an ordered trajectory

$$f,\ \mathsf{T}_{\mathcal{F}}(f),\ \mathsf{T}_{\mathcal{F}}^2(f),\ \ldots. \tag{25}$$

If the trajectory returns to its starting feature after more than one step, it forms a non-trivial ring. If a feature maps directly to itself, it forms a fixed point corresponding to a candidate pitch-invariant concept. If the trajectory terminates because its next successor is undefined, it forms an open sequence. Once one or a few features in a recovered structure are grounded using interpretable examples, the successor relation iteratively interprets the remaining features.

4 Experiments

In this section, we describe the experimental setup and present the concepts discovered by our method in the foundation-model representations.

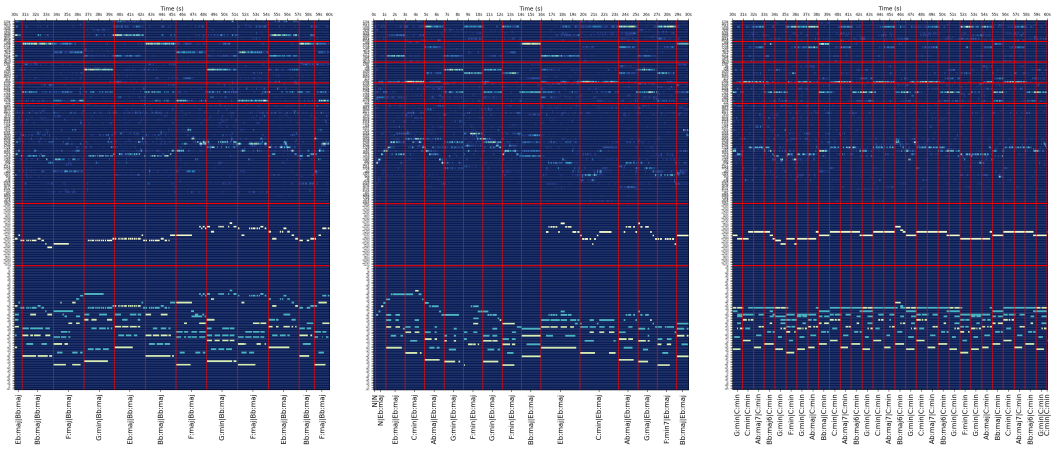
4.1 Datasets and Configuration

We use 326 and 164 songs from the training split of Slakh2100 [20] as our training and validation sets, respectively. Slakh2100 is a synthetic multi-instrument dataset synthesized using high-quality sound fonts. See Appendix B for usage details. We extract frame-level audio representations from two pretrained music foundation models: MuQ [6] and MusicFM [5]. Both models comprise 12 layers with a hidden dimensionality of $d = 1024$. Their frame rate is 25 fps. During training, we use 32 30-second representations per batch to train our SAEs, with a learning rate of 1e-3.

4.2 Recovered Orbits

We recover several pitch-invariant orbits and fixed points in the MuQ and MusicFM representations[3]. We found that both models imply some similar 12-size rings and chains, such as the major and

[3]Demo page with listening samples is available at `https://Kikyo-16.github.io/sae-alignment-demo-page`.



(a) B♭ major. (b) E♭ major. (c) C minor.

Figure 2: Activations of MuQ SAE features extracted from different-key songs, organized according to recovered orbits. The horizontal axis is song time with annotated chord roots. The vertical axis is feature indices. We append the the piano-roll ground truth as the bottom row. Vertical grid lines separate chord segments, and horizontal grid lines separate different orbits. Four 12-sized orbits are shown to correspond with the chord progression and one long chain mimics the melody.

minor chord ring, and the pianoroll-like chain, as shown in Figure 2. We select these three realistic multi-instrument vocal recordings from POP909 dataset. More activation examples including failure expamles are included in Appendix G.4 and I.3.

We uncover a previously unreported representation pattern: a subdominant-centered orbit encoding relative key structure. More specifically, we identify a 12-feature ring that appears to encode scale-degree information relative to the key. For example, as shown in Figure 2a, the first row corresponds to this ring. In a piece in *B♭ major*, frames with chord root *E♭* (scale degree $\hat{4}$ in *B♭ major*) activate one feature in the ring, while other frames do not. If we label this feature as *E♭*, then the remaining features are labeled following the cyclic order of the ring. We tested this ring on additional samples in different keys and observed similar behavior in the major keys. In minor keys, the ring tends to follow the relative major key rather than the tonic minor key. The occurrence of this relative-major subdominant provides new evidence for how a foundation model encodes the concept of key in local frames.

Motivated by this discovery, we conduct additional experiments in Section 5 to assess the quality of rings discovered from different foundation models. These include the relative-major subdominant ring, the major chord ring, and the minor chord ring.

5 Downstream Task Evaluation

We identified several prominent orbits in the SAE feature space that correspond to musically meaningful concepts, including major chords, minor chords, and musical keys, by inspecting the activations of selected anchor samples. In this section, we leverage these discovered orbits for two downstream music understanding tasks, with the goal of evaluating the quality of the discovered features and comparing the interpretability of the two foundation models.

5.1 Baselines

For key detection, we train a supervised single-layer linear probe on the same training dataset as the baseline. Motivated by the recovered relative-major subdominant orbit, the probe has 12 outputs and is trained to detect frames whose chord root matches the relative-major subdominant of the current key. For example, in an *B♭ major* sample, this root is *E♭*; frames with chords rooted in *E♭* receive a one-hot target at the *E♭* entry, while all others receive the zero vector. The probe is trained with binary cross-entropy loss. For chord recognition, we additionally train a supervised single-layer linear probe as a baseline. The probe predicts 25 classes, corresponding to 12 major chords, 12 minor

Table 1: Weighted Chord Recall (WCR) across Different Models.

Models	Representations	Root WCR ↑			MajMin WCR ↑		
		POP909	RWC	Slakh2100	POP909	RWC	Slakh2100
s-SAE (Est. Bd)	MusicFM (4)	71.08	67.56	70.32	65.93	64.68	68.39
	MuQ (2)	77.42	78.08	78.10	73.11	76.92	76.53
o-SAE (Est. Bd)	MusicFM (4)	74.01	69.65	75.35	70.09	67.68	73.99
	MuQ (2)	80.03	79.79	78.45	73.68	77.40	75.61
s-SAE (Oracle Bd)	MusicFM (4)	80.31	78.43	79.85	75.14	75.82	79.67
	MuQ (2)	83.84	84.91	84.08	79.45	83.97	82.47
o-SAE (Oracle Bd)	MusicFM (4)	80.39	77.49	81.11	76.52	75.63	80.05
	MuQ (2)	**85.29**	86.21	84.67	78.96	83.76	82.06
Probe (Oracle Bd)	MusicFM (4)	83.60	82.80	85.49	80.13	80.19	85.66[4]
	MuQ (2)	84.17	86.01	**86.05**	**81.49**	84.66	**86.42**
LVCR	-	84.25	**87.96**	85.65	81.37	**87.47**	83.81

Table 2: Key Detection Accuracy (ignoring relative major/minor difference).

Models	Representations	Song-level Accuracy ↑			Segment-level Accuracy ↑		
		FMAKv2	Gtzan	GS	FMAKv2	Gtzan	GS
Probe	MusicFM (3)	50.92	48.31	40.89	44.92	48.31	36.39
	MuQ (5)	54.66	48.80	47.68	48.33	48.80	40.33
s-SAE	MusicFM (3)	35.09	30.60	25.99	32.07	30.60	25.50
	MuQ (5)	56.61	48.80	47.19	50.00	48.80	41.20
o-SAE	MuQ (5)	**56.92**	**51.57**	**50.50**	**50.32**	**51.57**	**44.56**

chords, and a no-chord class. We further compare with two task-specific models, S-Key [21] for key detection and LVCR [22] for chord recognition. For S-Key and LVCR, we directly use their open-source checkpoints, which are trained on external datasets. See Appendix C for baseline details.

5.2 Metrics

We report segment- and song-level key accuracy, excluding relative major–minor errors, for evaluating the subdominant rings. For the other key detection evaluations, we report standard key accuracy and relaxed MIREX accuracy. For chord recognition, we report root weighted recall over 13 classes and root-and-quality (MajMin) weighted recall over 25 classes. See Appendex E for metric details.

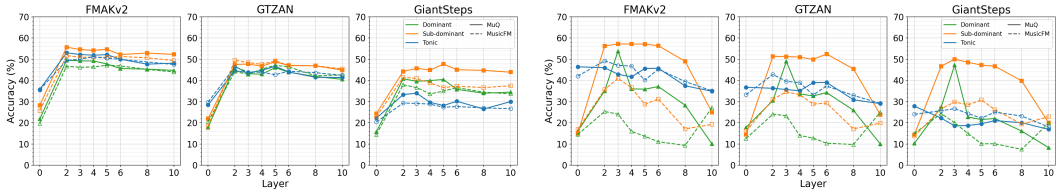
5.3 Ablations

To evaluate whether our algorithm preserves the semantic quality of general SAE features, we train a standard SAE as an ablation baseline. Since its features do not have an explicit orbit-like structure, we use the weight vectors of the frozen linear probes to retrieve corresponding SAE encoder weight vectors by cosine similarity. These matched features are then used for key detection and chord recognition. In the result table, the standard SAE and our proposed orbit-based SAE are denoted as *s-SAE* and *o-SAE*, respectively. Additional ablation results on hidden states from different layers, varying sparsity K, different τ, and different random seeds are provided in Appendix H and I.

5.4 Results

We discovered a subdominant ring in the MuQ representations, but not in the MusicFM representations, whereas major- and minor-chord rings emerged in both models. Across all evaluation settings, the orbits discovered in the MuQ representations were consistently of higher quality than those discovered in the MusicFM representations. All SAE results reported in the tables use a sparsity level of $K = 48$.

[4]Because root-with-quality evaluation excludes unsupported labels (e.g. suspended chords), it assesses fewer frames. As a result, its score can be lower than the root-only score.



(a) Probe (b) s-SAE

Figure 3: Key detection accuracy comparing probing with standard SAE. Given labels, probing yields generally high accuracy regardless of base model and concept nature. In contrast, SAE uncovers high-contrast but consistent results that are specific to music concept.

Table 3: Key Detection Results Derived from Chord-based Features.

		MIREX Accuracy ↑			Accuracy ↑		
Methods	Representations	FMAKv2	Gtzan	GS	FMAKv2	Gtzan	GS
Probe	MusicFM (2)	62.75	60.29	65.98	53.26	49.76	56.95
	MuQ (2)	66.13	63.22	69.70	56.90	52.89	61.92
s-SAE	MusicFM (2)	68.89	63.76	63.96	60.42	55.66	54.47
	MuQ (2)	68.87	66.63	69.34	61.08	58.92	62.42
o-SAE	MusicFM (2)	69.89	64.40	61.76	61.76	56.51	52.32
	MuQ (2)	70.17	67.09	71.14	64.07	58.67	62.35
S-KEY	-	**73.24**	**74.22**	**72.09**	**64.92**	**65.54**	**64.40**

5.4.1 Chord Rings

Table 1 reports SAE-based results using chord boundaries estimated from SAE features (*Est. Bd*) and reference chord boundaries (*Oracle Bd*). The number in parentheses after each representation name indicates the layer index. Details of the chord-prediction post-processing procedure and additional layer-wise results are provided in Appendices D and H, respectively.

The discovered major- and minor-chord rings support effective chord recognition, achieving competitive performance against fully supervised methods despite requiring minimal label supervision. Moreover, rings derived from MuQ representations consistently outperform those derived from MusicFM representations.

5.4.2 Subdominant Ring

Key-detection performance in Table 2 demonstrates that our SAE discovers subdominant rings that capture musically meaningful key information. Across all evaluations, rings derived from MuQ representations consistently outperform those from MusicFM.

Intuitively, one would expect the algorithm to recover tonic or dominant rings alongside the subdominant, where in fact it does not. To verify whether tonic and dominant signals are present in the underlying representations, we conduct linear probing experiments. Figure 3a confirms the presence of both signals, albeit weaker than the subdominant signal.

Are they dropped during sparse feature selection? We identify s-SAE encoder weight vectors closest to the linear probe weight vectors. Figure 3b shows that the subdominant features selected this way maintain strong performance, whereas the tonic and dominant features remain absent. This discrepancy indicates that while standard linear probes detect weak tonic and dominant signals, the SAE does not evaluate them as active or salient features.

5.4.3 SAE Evaluates Features Saliency Beyond Linear Recoverability

The divergence between probing and SAE-based extraction suggests that tonic and dominant information may not exist as explicit, readily accessible concepts within the foundation model, but are instead resynthesized by the supervised linear probe.

To shed more light on this hypothesis, we examine whether key information can be inferred from readily represented chord progressions. We develop a music-informed rule-based algorithm (detailed in Appendix D.3) to infer major and minor keys by aggregating frame-level chord activations (from SAEs or probes) and weighting the corresponding tonic, subdominant, and dominant functions.

The chord-infer-key evaluation (Table 3) shows that keys can be effectively reconstructed through simple manual linear combination of chord activations. These results show that key information can be dispersed in other concepts. It implies that linear probes can predict tonic signals even when the underlying representation lacks a distinct, explicit tonic concept. In contrast, the unsupervised orbit-based SAE discovers only the salient structure: the subdominant ring.

To generalize, linear recoverability does not guarantee that the corresponding information is organized as an explicit concept in the representation, while SAEs capture concepts that are salient and sparse. This motivates orbit-based analysis, which examines not only recoverability but also the structural organization of musical concepts.

5.4.4 Global Recoverability vs. Local Structure

Previous probing studies [7, 9, 10, 11] often aggregate frame-level representations into a global representation using max or average pooling and then train a supervised probe to predict musical key. While this evaluates whether key information is recoverable, it does not reveal how key-related concepts are organized and encoded over time. In contrast, our method identifies an explicit subdominant concept, a local representation that implies the key. Notably, even linear probing shows that the subdominant is the most salient local concept for key perception (Figure 3a), an enlightening result that previous global-key studies missed.

6 Limitations

The orbits in this study are characterized by pitch transposition. Pitch-invariant concepts, such as rhythm, cannot be organized without additional inductive biases. In parallel, our method focuses on the frame level, and it remains unexplored how foundation models organize and aggregate frame-level concepts over time. Additionally, although 12-sized rings emerge from unsupervised SAE alignment, it is potentially related to the pitch transposition step size being $1/12$ of an octave. That, combined with the distribution of data we use, yields mostly Western music concepts. How other transformations may interpret other forms of music is yet to be studied.

7 Conclusion and Future Work

In conclusion, we presented a **structure-aware interpretability** framework that leverages pitch transposition as an inductive bias to recover orbit structures over SAE features via multi-view alignment. These orbits reveal relational organization in learned representations, enabling the interpretation of higher-level musical concepts such as chords, keys, and melodic patterns with minimal supervision. Notably, we uncover a subdominant-centered orbit that encodes relative key structure—a previously unreported representation pattern that resembles the "circle of fifth" commonly used in tonal theory.

More broadly, we believe this structure-centric perspective points to several promising directions. First, the proposed framework is not limited to music: incorporating known transformations as inductive biases may enable structure-based interpretability in other domains, and can naturally complement existing feature-level methods. Second, while our approach does not eliminate polysemanticity at the level of individual features, it offers robustness to such ambiguity by shifting the unit of analysis to structured groups, where relational consistency reduces feature-level noise. Finally, the presence of coherent orbit structures suggests a potential criterion for evaluating learned representations: models that exhibit consistent transformation-induced structure may reflect a deeper level of conceptual organization.

We hope this work motivates a broader shift toward structure-aware interpretability, where understanding emerges not only from individual features, but from the relational systems they collectively form.

Acknowledgments

We would like to thank **Daniel Chin** for his helpful research discussions and assistance with writing, and **Junyan Jang** for providing valuable music theory support. We also thank Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), with special thanks to **Prof. Qirong Ho** and his team for providing computational resources.

References

[1] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, "Simple and controllable music generation," *Advances in neural information processing systems*, vol. 36, pp. 47 704–47 720, 2023.

[2] L. Yizhi, R. Yuan, G. Zhang, Y. Ma, X. Chen, H. Yin, C. Xiao, C. Lin, A. Ragni, E. Benetos *et al.*, "Mert: Acoustic music understanding model with large-scale self-supervised training," in *The Twelfth International Conference on Learning Representations*, 2023.

[3] S. Ghosh, A. Goel, J. Kim, S. Kumar, Z. Kong, S.-g. Lee, C.-H. H. Yang, R. Duraiswami, D. Manocha, R. Valle *et al.*, "Audio flamingo 3: Advancing audio intelligence with fully open large audio language models," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[4] S. Wu, G. Zhancheng, R. Yuan, J. Jiang, S. Doh, G. Xia, J. Nam, X. Li, F. Yu, and M. Sun, "Clamp 3: Universal music information retrieval across unaligned modalities and unseen languages," in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 2605–2625.

[5] M. Won, Y.-N. Hung, and D. Le, "A foundation model for music informatics," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 1226–1230.

[6] H. Zhu, Y. Zhou, H. Chen, J. Yu, Z. Ma, R. Gu, Y. Luo, W. Tan, and X. Chen, "Muq: Self-supervised music representation learning with mel residual vector quantization," *IEEE Transactions on Audio, Speech and Language Processing*, 2025.

[7] R. Castellon, C. Donahue, and P. Liang, "Codified audio language modeling learns useful representations for music information retrieval," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021. [Online]. Available: https://archives.ismir.net/ismir2021/paper/000010.pdf

[8] R. Yuan, Y. Ma, Y. Li, G. Zhang, X. Chen, H. Yin, Y. Liu, J. Huang, Z. Tian, B. Deng *et al.*, "Marble: Music audio representation benchmark for universal evaluation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 39 626–39 647, 2023.

[9] J. Koo, G. Wichern, F. G. Germain, S. Khurana, and J. Le Roux, "Understanding and controlling generative music transformers by probing individual attention heads," in *IEEE ICASSP Satellite Workshop on Explainable Machine Learning for Speech and Audio (XAISA)*, 2024.

[10] M. Wei, M. Freeman, C. Donahue, and C. Sun, "Do music generation models encode music theory?" in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, 2024.

[11] C. Papaioannou, E. Benetos, and A. Potamianos, "Universal music representations? evaluating foundation models on world music corpora," in *Proceedings of the 26th International Society for Music Information Retrieval Conference (ISMIR)*, 2025.

[12] A. Makhzani and B. Frey, "K-sparse autoencoders," *arXiv preprint arXiv:1312.5663*, 2013.

[13] L. Gao, T. D. la Tour, H. Tillman, G. Goh, R. Troll, A. Radford, I. Sutskever, J. Leike, and J. Wu, "Scaling and evaluating sparse autoencoders," in *The Thirteenth International Conference on Learning Representations*, 2024.

[14] N. Singh, M. Cherep, and P. Maes, "Discovering and steering interpretable concepts in large generative music models," in *AI for Music Workshop*, 2025.

[15] G. Aparin, T. Sadekova, A. Rukhovich, A. Yermekova, L. Kushnareva, V. Popov, K. Kuznetsov, and I. Piontkovskaya, "Audiosae: Towards understanding of audio-processing models with sparse autoencoders," *arXiv preprint arXiv:2602.05027*, 2026.

[16] M. Gotham, K. Gullings, C. Hamm, B. Hughes, B. Jarvis, M. Lavengood, and J. Peterson, "Transposition," in *Open Music Theory*, version 2 ed. GitHub, 2021, an open educational resource for music theory. [Online]. Available: https://openmusictheory.github.io/transposition.html

[17] S. Chen, E. Dobriban, and J. H. Lee, "A group-theoretic framework for data augmentation," *Journal of Machine Learning Research*, vol. 21, no. 245, pp. 1–71, 2020.

[18] H. Cunningham, A. Ewart, L. Riggs, R. Huben, and L. Sharkey, "Sparse autoencoders find highly interpretable features in language models," *arXiv preprint arXiv:2309.08600*, 2023.

[19] H. Thasarathan, J. Forsyth, T. Fel, M. Kowal, and K. G. Derpanis, "Universal sparse autoencoders: Interpretable cross-model concept alignment," in *Forty-second International Conference on Machine Learning*, 2025.

[20] E. Manilow, G. Wichern, P. Seetharaman, and J. Le Roux, "Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019.

[21] 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 *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.

[22] J. Jiang, K. Chen, W. Li, and G. Xia, "Large-vocabulary chord transcription via chord structure decomposition." in *ISMIR*, 2019, pp. 644–651.

[23] Z. Wang*, K. Chen*, J. Jiang, Y. Zhang, M. Xu, S. Dai, G. Bin, and G. Xia, "Pop909: A pop-song dataset for music arrangement generation," in *Proceedings of 21st International Conference on Music Information Retrieval, ISMIR*, 2020.

[24] M. Goto, S. Balke, and M. Mueller, "Rwc music database," May 2026. [Online]. Available: https://doi.org/10.5281/zenodo.18656623

[25] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[26] P. Knees, Á. Faraldo Pérez, P. Herrera Boyer, 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 the 16th International Society for Music Information Retrieval Conference (ISMIR); 2015 Oct 26-30; Málaga, Spain.[Málaga]: International Society for Music Information Retrieval, 2015. p. 364-70.* International Society for Music Information Retrieval (ISMIR), 2015.

[27] Y. Kong, V. Lostanlen, G. Meseguer-Brocal, S. Wong, M. Lagrange, and R. Hennequin, "STONE: Self-supervised Tonality Estimator," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, San Francisco, United States, Nov. 2024. [Online]. Available: https://hal.science/hal-04675781

[28] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "Mir_eval: A transparent implementation of common mir metrics." in *ISMIR*, vol. 10, 2014, p. 2014.

A Primer on Music Theory Fundamentals

To make this work self-contained for readers without a formal background in music theory, this section provides a brief primer on the core musical concepts referenced throughout the paper, including pitch classes, pitch transposition, chords, harmonic functions, relative keys, and the Circle of Fifths.

A.1 Pitch, Pitch Classes, and Equal Temperament

In Western tonal music, sound frequencies are discretized into musical pitches. Under the standard twelve-tone equal temperament (12-TET) system, an octave is divided into 12 logarithmically equal intervals called semitones. Pitches that differ by one or more full octaves share the same chroma or harmonic quality. Abstracting away octave height yields the concept of a **pitch class**. The set of pitch classes is naturally represented modulo 12:

$$\mathcal{P} = \{0, 1, 2, \ldots, 11\} \cong \mathbb{Z}_{12}, \tag{26}$$

corresponding to the pitch names $\{\mathrm{C}, \mathrm{C}\sharp/\mathrm{D}\flat, \mathrm{D}, \ldots, \mathrm{B}\}$.

A.2 Chords and Pitch Transposition

A **chord** is a combination of two or more distinct pitch classes played simultaneously or sequentially. For instance, a *C major triad* consists of the pitch classes $\{\mathrm{C}, \mathrm{E}, \mathrm{G}\}$. **Pitch transposition** is an operator $T_k : \mathcal{P} \to \mathcal{P}$ that shifts every pitch class upward by k semitones:

$$T_k(p) = (p + k) \pmod{12}. \tag{27}$$

When applied to a chord, pitch transposition shifts its pitch constituents while preserving its internal interval structure (i.e., chord quality such as major or minor). Repeatedly applying a 1-semitone transposition induces a cyclic ordered set (an orbit) of size 12 over pitch-class space.

A.3 Harmonic Functions and the Circle of Fifths

In tonal harmony, pitch classes and chords revolve around functional roles relative to a central pitch called the **Tonic** (I, scale degree 1), which acts as the tonal "home" or resolution point. Two fundamental harmonic functions anchored to the tonic are:

- **Dominant** (V, scale degree 5): Located $+7$ semitones (a perfect fifth) above the tonic (e.g., *G* in *C major*). It creates strong harmonic tension that pulls directly toward the tonic.
- **Subdominant** (IV, scale degree 4): Located $+5$ semitones above (or -7 semitones below, a perfect fourth) the tonic (e.g., *F* in *C major*). It provides a fundamental harmonic framework and expands the tonal center.

The **Circle of Fifths**[5] as shown in Figure 4 organizes all 12 chromatic pitch classes geometrically according to the interval of a perfect fifth (7 semitones). Moving clockwise around the circle corresponds to stepping up by $+7$ semitones (toward the Dominant), whereas moving counter-clockwise corresponds to stepping down by -7 semitones / up by $+5$ semitones (toward the Subdominant). As illustrated in Figure 4, the Tonic, Subdominant, and Dominant occupy three adjacent positions on the circle, reflecting maximum harmonic overlap and structural proximity.

A.4 Relative Keys and Key Signatures

A **key signature** designates the set of sharps or flats shared by a diatonic scale. Every major key shares its exact key signature and constituent pitch class set with a corresponding **relative minor key** (located 3 semitones / a minor third below the major tonic, e.g., *C major* and *A minor*). On the Circle of Fifths (Figure 4), relative major and minor keys are geometrically paired together in the outer and inner rings, respectively.

[5] Image source: `https://en.wikipedia.org/wiki/Circle_of_fifths`.

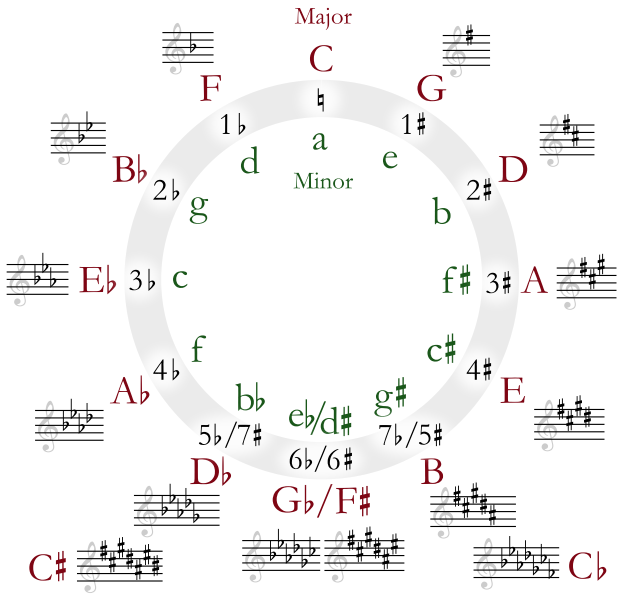


Figure 4: **The Circle of Fifths.** A geometric representation of the relationships among the 12 chromatic pitch classes, harmonic functions, and relative key signatures arranged by intervals of perfect fifths. Outer circle labels represent major keys; inner circle labels represent relative minor keys.

B Datasets

The SAEs and supervised probes are trained exclusively on Slakh2100 Dataset, while all reported downstream results are obtained on realistic-recording datasets or on the official Slakh2100 test split. No evaluation track is used for training or model selection. Table 4 summarizes the datasets used in our experiments.

B.1 Training and Validation Data

Slakh2100. We construct the training and validation sets from Slakh2100 [20], a multitrack dataset synthesized from aligned MIDI performances using high-quality sample-based soundfonts. A notable difference from real-world recordings is the absence of vocals. Because vocal signals contain strong phonetic and speech-related structure, they may occupy substantial representational capacity while being unrelated to the musical concepts studied here. Slakh2100 therefore provides a controlled setting for analyzing transposition-induced harmonic and melodic structure.

Nevertheless, synthesized audio does not capture the full acoustic variability of real recordings, including expressive articulation, room acoustics, microphone characteristics, mixing, and mastering. Its MIDI arrangements, however, originate from realistic musical scores and performances, preserving much of their symbolic and structural content. The remaining domain gap is thus primarily acoustic, and we evaluate transfer to real recordings in our downstream experiments.

Key-balanced split construction. We select subsets of the Slakh2100 training and validation splits to obtain approximately balanced coverage of the 24 major and minor keys. Tracks are grouped by their pseudo key labels, and up to 16 tracks are sampled from each class. For classes with fewer than 16 tracks, all available tracks are retained without duplication. Using a fixed random seed of 42 yields 326 training tracks and 164 validation tracks.

Pseudo frame-level chord and key annotations are derived from the aligned MIDI files using the MIDI-AutoLabel-Dataset toolkit. They are used for split balancing, supervised probe training, and validation metrics, but not as supervision for SAE training.

Table 4: Overview of the datasets used in our experiments. The reported numbers correspond to tracks retained after label-validity filtering.

Dataset	Usage	Audio domain	Tracks
Slakh2100	Training	Synthesized, instrumental	326
Slakh2100	Validation	Synthesized, instrumental	164
FMAKv2	Key detection	Real recordings	5,434
GTZAN-Key	Key detection	Real recordings	830
GiantSteps	Key detection	Real recordings	604
POP909	Chord recognition	Real recordings	810
RWC-POP-100	Chord recognition	Real recordings	100
Slakh2100 test	Chord recognition	Synthesized, instrumental	151

Table 5: Major–minor distribution of the key-detection datasets.

Dataset	Major	Minor	Total
FMAKv2	2,777	2,657	5,434
GTZAN-Key	357	473	830
GiantSteps	93	511	604

Segmentation and pitch shifting. Each track is divided into 30-second segments with a 10-second hop; silent segments are removed. We generate five views at $\{-2, -1, 0, +1, +2\}$ semitones using the `torchaudio` phase vocoder with a Hann window, $n_{\text{fft}} = 1024$, and hop length 256. For a shift of s semitones, the phase-vocoder rate is $2^{-s/12}$, after which the waveform is linearly resampled to its original length to preserve temporal alignment. When aligning $S = 3$ SAEs, three consecutive views are randomly sampled for each training example. Validation tracks are divided into non-overlapping 30-second segments.

B.2 Evaluation Data

We evaluate both in-domain performance and transfer from the synthesized, vocal-free training data to real recordings. Chord recognition is evaluated on Chinese, Japanese, and western popular music, while key detection is evaluated across genre-diverse and electronic music collections.

B.2.1 Chord-Recognition Datasets

POP909. POP909 [23] contains Chinese popular songs with MIDI arrangements aligned to the original audio. We use tracks 100–909, yielding 810 songs, and evaluate the original vocal mixes using the provided time-aligned chord annotations.

RWC-POP-100. We use all 100 chord-annotated songs from RWC-POP-100 [24]. The professionally produced Japanese popular-music recordings contain vocals, expressive performances, and production effects absent from Slakh2100. Together, POP909 and RWC-POP-100 test whether the discovered chord concepts transfer from synthesized instrumental audio to real recordings.

Slakh2100 test split. We use all 151 songs in the official Slakh2100 test split. These songs are disjoint from the training and validation data and provide an in-domain evaluation under the same synthesized, instrumental conditions.

B.2.2 Key-Detection Datasets

We evaluate on three commonly used key-detection datasets. Table 5 reports their major–minor distributions, which provide important context for interpreting the results, particularly for the strongly minor-dominated GiantSteps dataset.

Table 6: Genre distribution of GTZAN tracks.

Genre	Tracks	Genre	Tracks
Blues	98	Jazz	79
Country	96	Metal	93
Disco	97	Pop	94
Hip-hop	79	Reggae	97
Rock	97		
Total	830		

FMAKv2. FMAKv2 contains genre-diverse real recordings with song-level major and minor key labels. We retain 5,434 tracks with matched labels and successfully processed audio, including 2,777 major- and 2,657 minor-key tracks.

GTZAN-Key. GTZAN-Key extends the GTZAN genre collection [25] with global key annotations. Following the nine-genre evaluation protocol of S-KEY [21], we exclude the classical subset and evaluate only tracks with a unique global key, enabling direct comparison with this baseline. After audio preprocessing, 830 tracks remain across the nine genres listed in Table 6.

GiantSteps. GiantSteps [26] contains 604 electronic dance music excerpts with global key labels. We divide each excerpt into non-overlapping 30-second segments for evaluation. Its production-heavy timbres and strongly imbalanced mode distribution provide a distinct test of acoustic and genre transfer.

B.3 Evaluation Preprocessing and Exclusion Criteria

For evaluation, audio is divided into non-overlapping 30-second segments, and annotations are aligned to the frame-level representation timeline. All filtering rules are fixed before model evaluation. We exclude tracks that cannot be decoded, are shorter than one evaluation segment, lack matching annotations, or contain key changes, since key detection assumes a single global key per song. Accordingly, GiantSteps entries with multiple key labels and GTZAN-Key entries marked as modulation or unknown are removed.

For chord recognition, frames labeled as unknown are excluded from all metrics. Root WCR evaluates the remaining chord frames using only the root pitch class, regardless of chord quality. MajMin WCR follows the `mir_eval` vocabulary: reference frames that cannot be reduced to a major or minor triad are excluded from this metric.

C Baselines

We compare our method with supervised linear probes and task-specific music-analysis systems. The probes are trained on the same data and frozen foundation-model representations as our method. The task-specific systems are evaluated using their publicly released checkpoints, which were trained on external datasets.

Throughout this appendix, n indexes songs, t indexes frames, p indexes pitch classes, and s indexs sub-SAEs. We reserve i and j for SAE feature indices and use I to denote a temporal segment or interval. Let $\mathcal{P} = \{0, \ldots, 11\}$ denote the pitch-class set.

C.1 Chord Recognition

LVCR. LVCR [22] is a supervised large-vocabulary chord transcription system. It predicts chord components, including root, bass, triad, seventh, and extensions, using multitask learning and structured decoding. Its predictions are reduced to the Root and MajMin vocabularies at evaluation time.

Linear probe. Given a frozen frame-level representation $x_t \in \mathbb{R}^d$, we train a single affine layer

$$h_t^{\mathrm{ch}} = W^{\mathrm{ch}} x_t + b^{\mathrm{ch}}, \qquad W^{\mathrm{ch}} \in R^{25 \times d}, \tag{28}$$

where the 25 outputs correspond to 12 major chords, 12 minor chords, and the no-chord class N. Reference labels are reduced to the MajMin vocabulary before training. Extensions that preserve an underlying major or minor triad are folded into the corresponding triad, whereas unsupported chord qualities and unknown annotations are ignored. The no-chord label is retained as the twenty-fifth class.

Let $\mathcal{T}_{\mathrm{ch}}$ denote the set of valid training-frame indices and let $y_t^{\mathrm{ch}} \in \{0, \ldots, 24\}$ denote the reduced chord target. The probe is optimized using frame-level cross-entropy:

$$\mathcal{L}_{\mathrm{ch}} = -\frac{1}{|\mathcal{T}_{\mathrm{ch}}|} \sum_{t \in \mathcal{T}_{\mathrm{ch}}} \log \left[\mathrm{softmax}\left(h_t^{\mathrm{ch}}\right)\right]_{y_t^{\mathrm{ch}}}. \tag{29}$$

C.2 Key Detection

S-KEY. S-KEY [21] is a self-supervised key-estimation model extending STONE [27] with transposition-invariant chroma pseudo-labels. This enables major–minor key prediction without human key annotations. We evaluate its released checkpoint without fine-tuning.

Linear probes. We independently train frame-level probes for the tonic, subdominant, and dominant functions. Each probe has 12 outputs corresponding to pitch classes:

$$h_t^{(q)} = W^{(q)} x_t + b^{(q)}, \qquad q \in \{\mathrm{ton}, \mathrm{sub}, \mathrm{dom}\}, \tag{30}$$

where q denotes the target harmonic function. Let $r_t^{\mathrm{rel}} \in \mathcal{P}$ denote the relative-major key signature at frame t. The pitch class associated with harmonic function q is

$$\xi_t^{(q)} = \left(r_t^{\mathrm{rel}} + \delta_q\right) \bmod 12, \qquad \delta_q = \begin{cases} 0, & q = \mathrm{ton}, \\ 5, & q = \mathrm{sub}, \\ 7, & q = \mathrm{dom}. \end{cases} \tag{31}$$

Given the reference chord root $r_t^{\mathrm{ch}} \in \mathcal{P}$, the binary target for output pitch class p is

$$y_{t,p}^{(q)} = \mathbb{1}\left[r_t^{\mathrm{ch}} = \xi_t^{(q)}\right] \mathbb{1}\left[p = \xi_t^{(q)}\right]. \tag{32}$$

Thus, matching frames receive a one-hot target, whereas all other frames receive an all-zero target. Let $\mathcal{T}_{\mathrm{key}}$ denote the set of valid training frames. Each probe is trained independently using binary cross-entropy:

$$\mathcal{L}^{(q)} = -\frac{1}{12|\mathcal{T}_{\mathrm{key}}|} \sum_{t \in \mathcal{T}_{\mathrm{key}}} \sum_{p \in \mathcal{P}} \left[y_{t,p}^{(q)} \log \sigma\left(h_{t,p}^{(q)}\right) + \left(1 - y_{t,p}^{(q)}\right) \log\left(1 - \sigma\left(h_{t,p}^{(q)}\right)\right)\right]. \tag{33}$$

The main comparison uses the subdominant probe, while the tonic and dominant probes are included in additional analyses.

D Downstream-Task Inference

We evaluate whether the discovered SAE orbits can be used directly for downstream prediction, without fitting an additional classifier on their activations. For controlled comparison, probe outputs are processed using the same aggregation and decision rules whenever possible.

D.1 Chord Recognition

The major- and minor-chord orbits each contain 12 features ordered by chord root. We smooth their frame-level activations using a nine-frame moving average. We consider two chord boundary settings. Under *Oracle Bd*, the reference chord annotations gives the boundaries. Under *Est. Bd*, we use local peaks in the activation of a discovered chord-boundary feature. The chord boundaries divide the song into segments, each segment I containing some contiguous frames $t \in I$.

Let $u_{t,p}^m$ denote the smoothed activation at frame t for root pitch class p and quality $m \in \{\mathrm{maj}, \mathrm{min}\}$. We first average the activation of each chord feature over time within each segment I:

$$\bar{u}_{I,p}^m = \frac{1}{|I|} \sum_{t \in I} u_{t,p}^m. \tag{34}$$

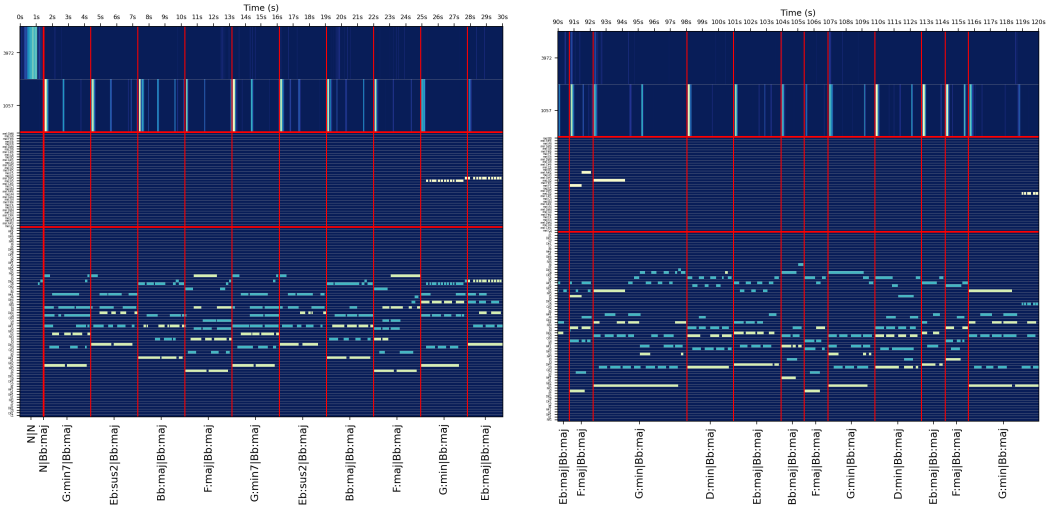


Figure 5: Activations of the selected silence and chord-boundary features. Both features are identified from a few music samples; in batched experiments, we approximate the same manual procedure using a small set of labeled anchors.

The estimated chord quality $\hat{m}_I$ and root pitch $\hat{p}_I$ for the segment are jointly selected by maximizing the segment-averaged activation:

$$(\hat{m}_I, \hat{p}_I) = \arg \max_{\substack{m \in \{\text{maj},\text{min}\} \\ p \in \mathcal{P}}} \bar{u}^m_{I,p}. \tag{35}$$

Under *Est. Bd*, the no-chord evidence is obtained from the discovered silence features. A segment is labeled as no-chord when its accumulated silence evidence exceeds both its accumulated major- and minor-chord evidence. Under *Oracle Bd*, it is obtained from reference frames labeled as no-chord.

For the supervised chord probe, we apply softmax to its 25 outputs. The first 12 probabilities represent major chords, the next 12 represent minor chords, and the final probability represents no-chord. We apply the same smoothing as above, use the oracle boundaries, and follow the same decoding procedure as above.

Chord-boundary and silence features. As shown in Figure 5, We identify silence features and chord boundary features using a small number of anchor examples.

D.2 Key-Signature Detection from Scale-Degree Features

The tonic, subdominant, and dominant methods each produce a ring of 12 pitch-class activations. For a 30-second segment, we average the ring over time and obtain a 12-size vector. Then we teat the most active pitch as the the detected harmonic-function root and infer the corresponding tonic. This method predicts only the 12-class key signature and does not distinguish relative major and minor keys. For the corresponding linear probes, again we use the same method but treating probe outputs as feature activations. At the song level, the 12-dimensional signature-score vectors are averaged across segments before selecting the final prediction.

D.3 Key Detection from Chord Orbits

To investigate whether key information is implied by chord-feature activations, we design a **music-theory-driven, template-based** method for estimating musical key from chord activations.

For each segment I, we aggregate the activations over time, apply square-root compression, and normalize the two chord rings separately:

$$b^m_{I,p} = \frac{\left(\bar{u}^m_{I,p}\right)^{1/2}}{\sum_{q\in\mathcal{P}}\left(\bar{u}^m_{I,q}\right)^{1/2}}, \qquad m \in \{\mathrm{maj}, \mathrm{min}\}. \tag{36}$$

This normalization prevents the overall activation strength of one chord quality from dominating the key score. For a candidate tonic k, let

$$M_d = b^{\mathrm{maj}}_{I,(k+d) \bmod 12}, \qquad m_d = b^{\mathrm{min}}_{I,(k+d) \bmod 12}, \tag{37}$$

where d denotes the pitch offset in semitones relative to the candidate tonic. Following common-practice tonal harmony, we compute a weighted sum of these normalized activations. For a major key with tonic k, we emphasize the tonic (M_0), subdominant (M_5), and dominant (M_7) chords, with additional support from either the minor supertonic (m_2) or minor submediant (m_9):

$$\mathrm{ScaleScore}_{\mathrm{maj}}(k) = M_0 + 0.4M_5 + 0.4M_7 + 0.1\max(m_2, m_9). \tag{38}$$

For minor keys, we take the maximum over three templates corresponding to common natural-, harmonic-, and melodic-minor chord patterns:

$$\begin{aligned}\mathrm{ScaleScore}_{\mathrm{min}}(k) = \max\Big\{&m_0 + 0.4m_5 + 0.4m_7 + 0.1M_8,\\ &m_0 + 0.4m_5 + 0.4M_7 + 0.1M_8,\\ &m_0 + 0.1m_2 + 0.4M_5 + 0.4M_7\Big\}.\end{aligned} \tag{39}$$

The predicted tonic and mode are selected jointly from the 24 candidates:

$$(\hat{k}_I, \hat{m}_I) = \arg \max_{\substack{k\in\mathcal{P}\\ m\in\{\mathrm{maj},\mathrm{min}\}}} \mathrm{ScaleScore}_m(k). \tag{40}$$

At the song level, the 24 candidate scores are averaged across segments before the final decision is made. For the supervised chord probe, we use the same scoring and decision procedure except treating the rectified major- and minor-chord logits as feature activations.

E Evaluation Metrics

E.1 Chord Recognition

We evaluate chord recognition using Root and MajMin Weighted Chord Symbol Recall (WCSR), following the vocabularies implemented in `mir_eval` [28]. Root WCSR evaluates only whether the predicted chord root is correct and ignores chord quality. MajMin WCSR requires both the root and the major or minor quality to be correct. Chord extensions that preserve an underlying major or minor triad are mapped to the corresponding major or minor class. Reference chords outside the MajMin vocabulary are excluded from the MajMin denominator. Chords with unsupported qualities may still be evaluated under Root when their root pitch class is defined. Unknown reference annotations are excluded from all chord metrics, while the no-chord label is retained as an evaluated class.

E.2 Key Detection

The tonic, subdominant, and dominant methods predict one of 12 key signatures, where relative major and minor keys are treated as equivalent. Major keys are represented by their tonic, while minor keys are mapped to their relative-major tonic, three semitones above. We report exact key-signature accuracy, with no partial credit for harmonically related errors. The metrics are computed and reported at both the segment and song levels.

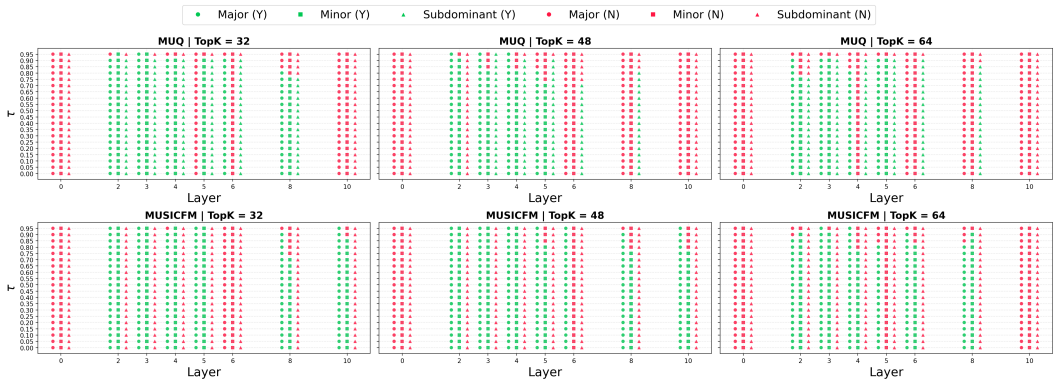


Figure 6: The recovery of three core orbits across different topK settings and different-layer representations under different τ.

For methods that predict both tonic and mode, we additionally report exact 24-class key accuracy and the standard MIREX weighted score:

$$\text{MIREXScoreWeight} = \begin{cases} 1.0, & \text{exact key}, \\ 0.5, & \text{fifth-related key in the same mode}, \\ 0.3, & \text{relative major–minor key}, \\ 0.2, & \text{parallel major–minor key}, \\ 0, & \text{otherwise}. \end{cases} \tag{41}$$

F Batched Post-hoc Evaluation Protocol

Our qualitative analyses assign semantic labels to recovered orbits by inspecting their activations on one or a few annotated music excerpts. To evaluate many model layers, sparsity settings, and random seeds efficiently and consistently, we batch the same post-hoc procedure. Using a small set of labeled anchors, the scripts match recovered structures to musical concepts, resolve their cyclic offsets, and select auxiliary features such as chord-boundary and silence features.

Importantly, this procedure is applied only after SAE training and orbit recovery. It does not introduce an additional training objective, modify the recovered graph, or affect orbit membership. It therefore approximates manual inspection without influencing how the orbit structures emerge.

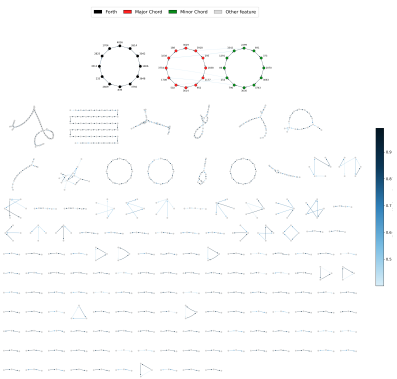
The supervised probe baselines use validation performance for checkpoint selection. To provide a similarly stable criterion for SAE checkpoints, we use a fixed trained chord probe only as a post-hoc semantic reference. At each checkpoint, the SAE features with the highest cosine similarity to the 12 major- and 12 minor-chord probe weight vectors are selected and evaluated on the validation set. The resulting chord-recognition accuracy is used solely to rank checkpoints; neither gradients nor probe supervision are introduced during SAE training. The chord orbits themselves emerge independently of this selection procedure.

G Visualization of Recovered Orbits

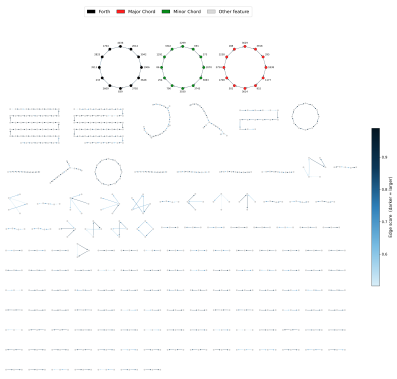
We further analyze the recovered feature orbits from three perspectives: their sensitivity to the recovery threshold, their structures in a less restrictive graph, and their activation patterns across music samples.

G.1 Sensitivity to the Recovery Threshold

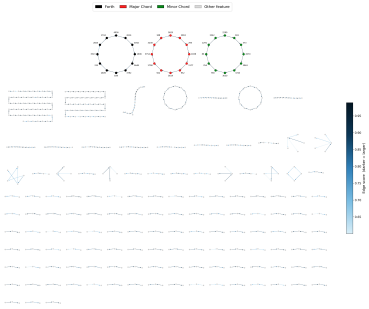
The confidence threshold τ controls which feature transitions are retained during orbit recovery. We vary τ and record whether the major-chord, minor-chord, and subdominant orbits are recovered across different model layers and sparsity settings. Figure 6 summarizes the results and shows how consistently each core orbit emerges across the range of thresholds. Interestingly, no minor-chord



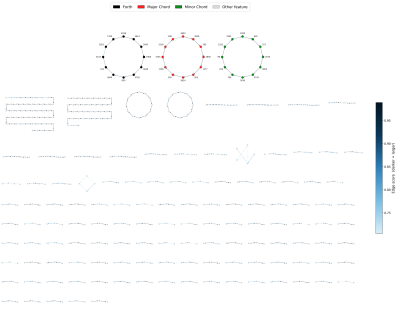
(a) MuQ (2), $\tau = 0.4$



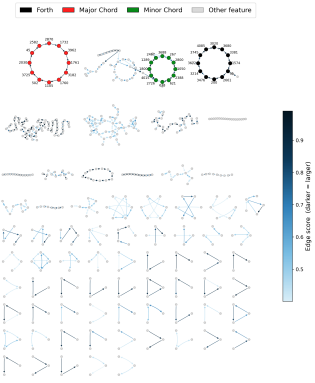
(b) MuQ (2), $\tau = 0.5$



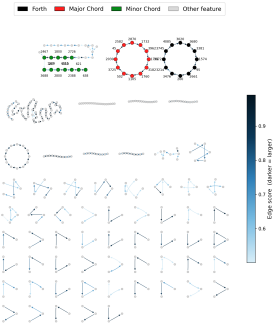
(c) MuQ (2), $\tau = 0.6$



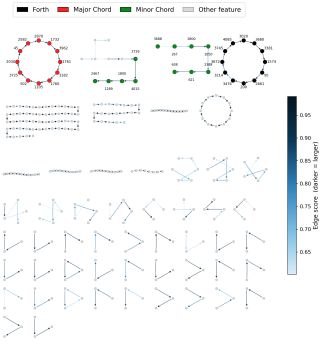
(d) MuQ (2), $\tau = 0.7$



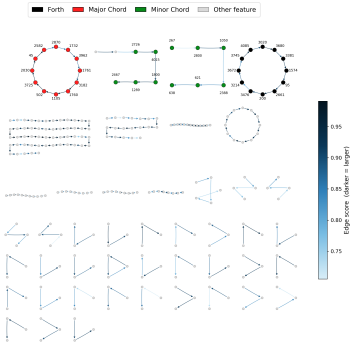
(e) MuQ (6), $\tau = 0.4$



(f) MuQ (6), $\tau = 0.5$

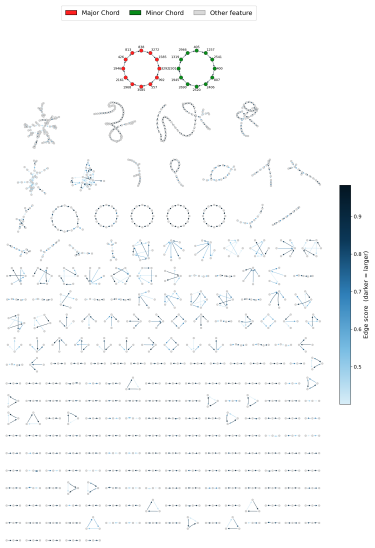


(g) MuQ (6), $\tau = 0.6$

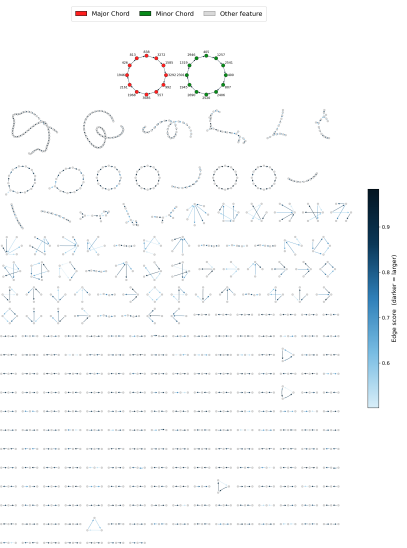


(h) MuQ (6), $\tau = 0.7$

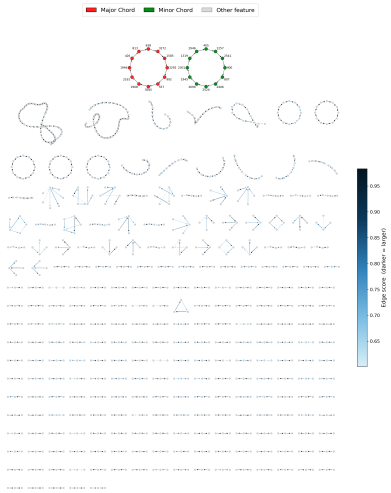
Figure 7: Directed orbit graphs recovered from MuQ at $K = 32$ with $n_e = 10$ and varying τ. Fixed points and components with fewer than three nodes are omitted, and "forth" refers to the subdominant.



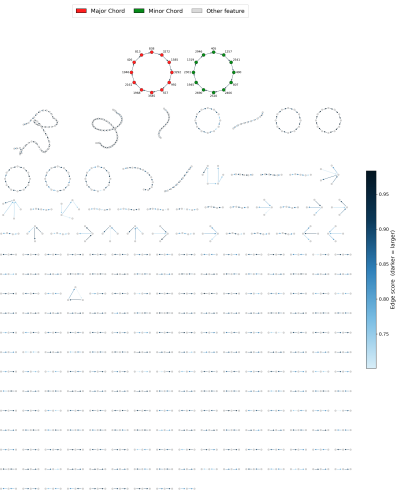
(a) MusicFM (2), $\tau = 0.4$



(b) MusicFM (2), $\tau = 0.5$



(c) MusicFM (2), $\tau = 0.6$



(d) MusicFM (2), $\tau = 0.7$

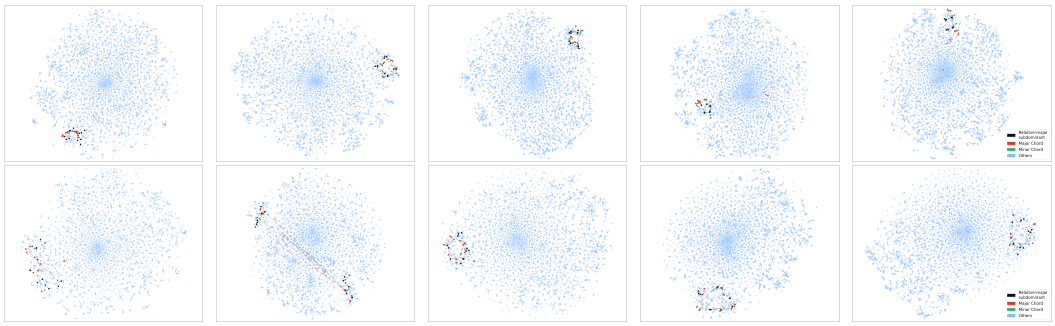
Figure 8: Directed orbit graphs recovered from MusicFM at $K = 32$ with $n_e = 10$ and varying τ. Fixed points and components with fewer than three nodes are omitted.

orbit is recovered from MuQ layer 6 at any tested threshold. We investigate this result further in the section below.

G.2 Sparse Directed Orbit Graphs

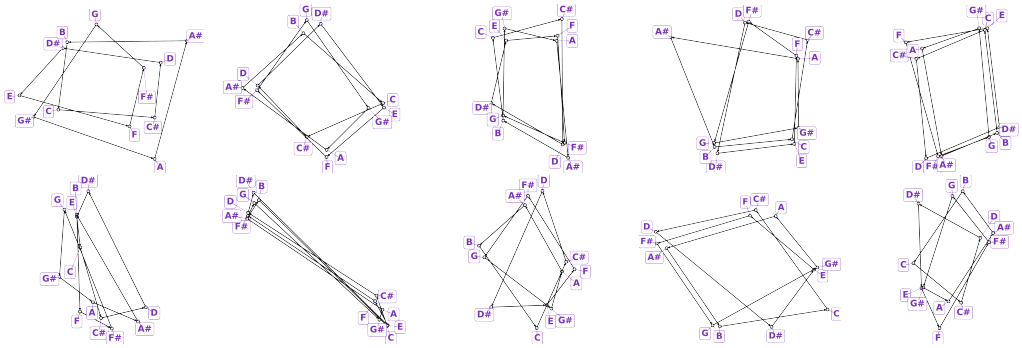
The orbit-recovery procedure in the main text retains at most one incoming and one outgoing edge for each feature. To inspect the broader feature structure, we generalize this constraint using n_e, the maximum number of retained incoming and outgoing edges per node. The original procedure is the special case $n_e = 1$.

Figures 7 and 8 visualize the resulting graphs with $n_e = 10$ and $\tau \in \{0.4, 0.5, 0.6, 0.7\}$. Some core orbits that do not form exact rings under $n_e = 1$ still exhibit nearly cyclic structures when more candidate edges are retained. This suggests that some orbit structures learned by the SAEs may be imperfect and therefore cannot be recovered directly by our strict algorithm, yet remain detectable through a simple and intuitive post-hoc graph analysis. We do so in the downstream task evaluation



(a) MuQ (2) (b) MuQ (3) (c) MuQ (4) (d) MuQ (5) (e) MuQ (6)

Figure 9: t-SNE projections of normalized SAE decoder vectors across different MuQ layers at $K = 32$. The columns correspond to different layers; the top row shows o-SAE, and the bottom row shows s-SAE. Black, red, and green points denote features in the subdominant, major-chord, and minor-chord orbits, respectively, while blue points denote the remaining active features.



(a) MuQ (2) (b) MuQ (3) (c) MuQ (4) (d) MuQ (5) (e) MuQ (6)

Figure 10: t-SNE visualization of the subdominant features across MuQ layers at $K = 32$. The top row shows o-SAE and the bottom row shows s-SAE. Directed edges follow the orbit order, and labels denote pitch classes.

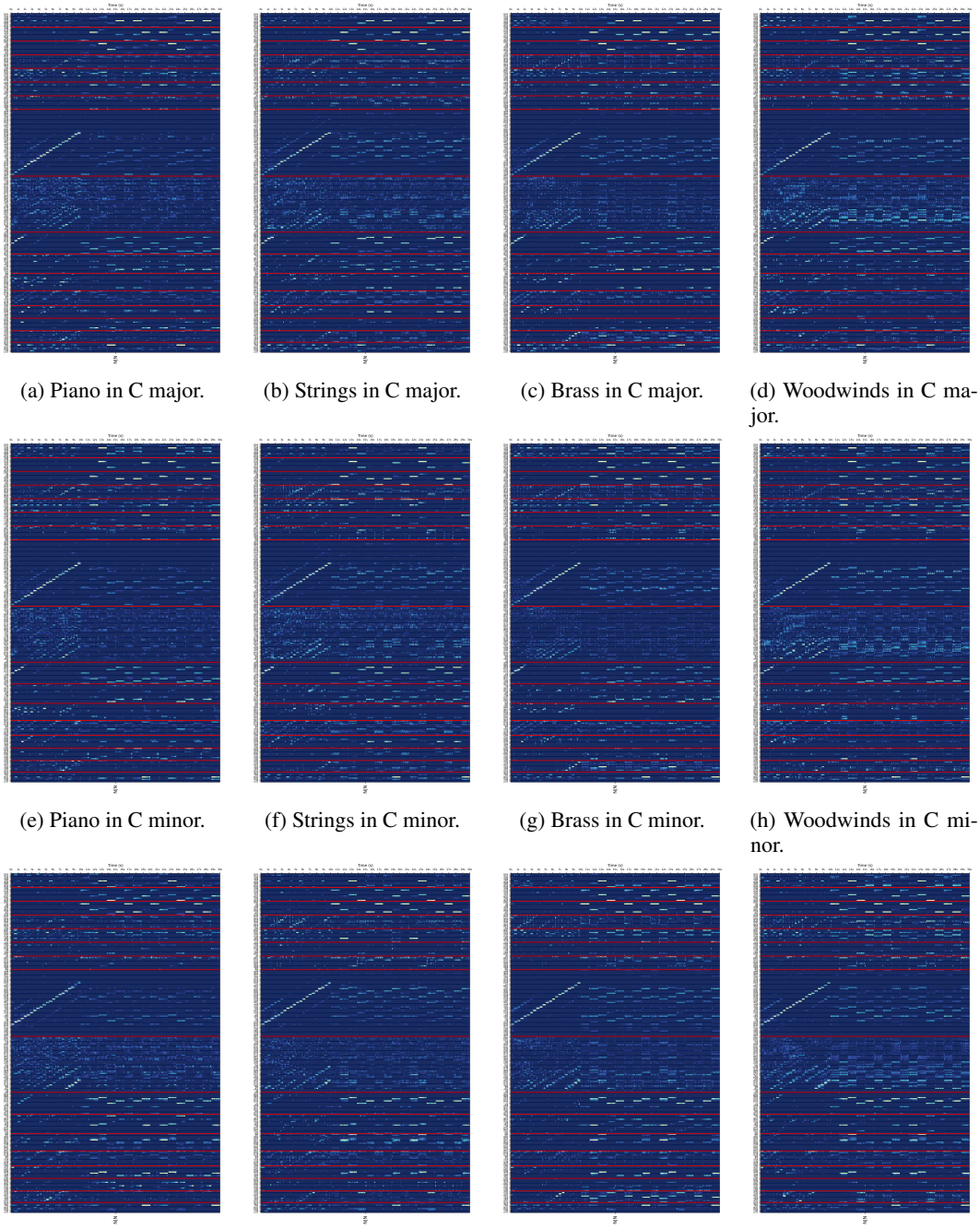
so that we can have a comprehensive evaluation of every potential core ring for each foundation model layer.

The graphs also contain many small connected components, commonly consisting of only three or four features. These components may reflect fragmented or local feature relations and are left for future investigation.

G.3 Geometric Structure of the Subdominant Ring

We visualize the SAE feature dictionary using t-SNE and examine the internal geometry of the recovered subdominant ring. We first normalize the decoder vector of each feature, reduce the vectors to 50 dimensions using PCA, and then project them into two dimensions using t-SNE. To reduce outliers, we include features that are activated in at least 2,000 segments in noth training and validation samples.

Figure 9 shows the three core orbits together with the remaining active o-SAE and s-SAE features across different MuQ layers at top-K $K = 32$. We add directed edges between successive subdominant features according to the recovered orbit order and annotate them by pitch class. The visualization as shown in Figure 10 reveals the relative arrangement of the subdominant features within the broader SAE dictionary. Because t-SNE primarily preserves local neighborhoods, we use this plot as qualitative evidence of feature organization rather than interpreting global distances quantitatively.



(a) Piano in C major. (b) Strings in C major. (c) Brass in C major. (d) Woodwinds in C major.

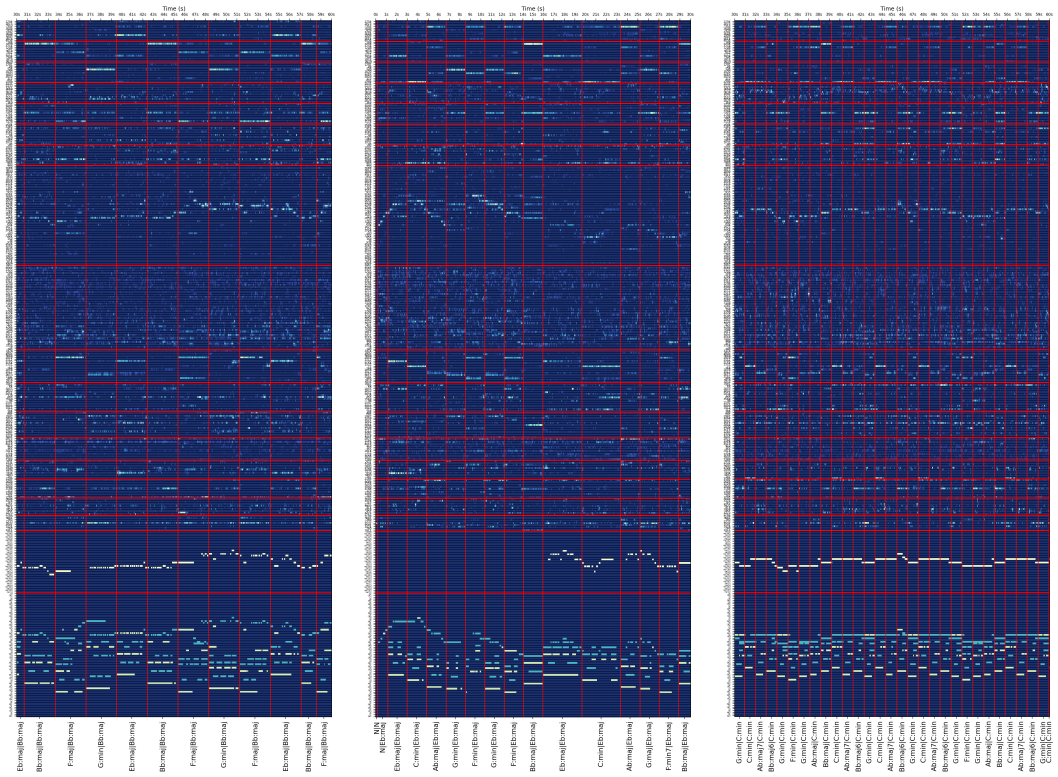
(e) Piano in C minor. (f) Strings in C minor. (g) Brass in C minor. (h) Woodwinds in C minor.

(i) Piano in A minor. (j) Strings in A minor. (k) Brass in A minor. (l) Woodwinds in A minor.

Figure 11: Frame-level activations of recovered MuQ (2) orbits on controlled synthetic excerpts in C major and A minor, rendered with piano, strings, brass, and woodwinds, using $K = 32$ and $\tau = 0.5$. The comparison isolates the effects of key and mode across different instrument families.

G.4 Orbit Activation Patterns

We visualize the frame-level activations of all the recovered orbits shown in Figure 7b with at least 10 features across music excerpts.



(a) B♭ major. (b) E♭ major. (c) C minor.

Figure 12: Frame-level activations of all recovered MuQ (2) orbits with more than 10 features for three POP909 excerpts, using $K = 32$ and $\tau = 0.5$. The horizontal axis shows time with chord annotations, while feature rows are grouped by recovered orbit. All excerpts are real multi-instrument vocal recordings with complex arrangements rather than synthetic audio.

Controlled synthetic diagnostic dataset. To separate pitch and harmonic effects from instrumentation and arrangement, we additionally construct a controlled diagnostic dataset. This dataset is used only for post-hoc analysis of the trained SAE and is not included in SAE training. It contains 96 audio samples, covering four instrument families, 12 tonic pitch classes, and two modes:

$$4 \text{ families} \times 12 \text{ roots} \times \{\text{major}, \text{minor}\}.$$

The four families are piano, strings, woodwinds, and brass. Piano samples use an acoustic grand piano, while the ensemble families distribute the musical parts across multiple instruments: cello, viola, and violins for strings; bassoon, clarinet, oboe, and flute for woodwinds; and trombone, French horn, and trumpets for brass.

Each sample lasts exactly 30 seconds. The first 10 seconds contain an ascending major or natural-minor scale spanning three octave regions. The 22 scale notes are evenly distributed over this interval. For the ensemble families, the low, middle, and high regions are assigned to different instruments, whereas all three regions are performed by the piano in the piano condition.

The remaining 20 seconds contain two repetitions of an eight-chord Canon progression. Major samples use I–V–vi–iii–IV–I–IV–V, while minor samples use i–v–VI–III–iv–i–iv–V. The final dominant in the minor progression is major and therefore introduces the raised seventh scale degree. Each chord lasts 1.25 seconds and is rendered as a four-note voicing. For strings, woodwinds, and brass, the four chord voices are distributed across different instruments within the family. The MIDI files are rendered to audio using FluidSynth and a General MIDI SoundFont.

Activation patterns in synthetic diagnostic dataset. Figure 11 shows the frame-level activations of recovered MuQ (2) orbits on controlled synthetic excerpts in *C major*, *C minor*, and *A minor* using $K = 32$ and $\tau = 0.5$. The comparison isolates the effects of key and mode across different

instrument families. The top three orbits in each subfigure are subdominant, major chord, and minor chord orbits, respectively. Each of these three orbits starts from root *C* to *B*.

Across the ascending and descending scale sections, similar ordered activation patterns recur over different instrument families. During the final chord progression, some orbit activations become more sustained and reflect the simultaneous harmonic content of each block chord. The recurrence of related patterns indicates that the recovered ordering is not specific to a single instrumental timbre. Nevertheless, activation strength and selectivity vary across families, showing that these features retain some sensitivity to timbre and orchestral voicing. Interestingly, it is quite easy for a human to distinguish a key from a measure-scale ascending or descending melody, but we can not find evidence that the model does so. Consistent with what we discuss throughout the paper, the activations of subdominant rings reveal that the model does detect the key where there is a submominant chord.

Activation patterns in real recordings. In the main text, we presented four selected orbits from three audio excerpts. Here, we expand the visualization to include all recovered orbits containing more than 10 features under the same MuQ (2) model with $K = 32$ and $\tau = 0.5$, as shown in Figure 12. The three excerpts are taken from POP909 and are real multi-instrument vocal recordings with complex arrangements. They therefore contain singing voices, overlapping instruments, non-chord tones, and production effects, rather than the simplified acoustic conditions of synthetic examples.

In the *B♭ major* and *E♭ major* excerpts, several orbit blocks exhibit localized activation changes aligned with chord boundaries. Within these blocks, different ordered features become prominent as the chord root changes, supporting the pitch-class interpretation of the recovered structures. A longer orbit also follows the activity of the piano-roll reference more directly, suggesting that some recovered structures encode local melody content rather than chord or key identity alone.

The *C minor* excerpt provides a less ideal example. Its activation patterns are broader and overlap more strongly across neighboring chord segments, and some expected orbit–chord correspondences are weak or incomplete. This example illustrates that the recovered semantics are not perfectly disentangled at every frame. Dense instrumentation, vocal melody, non-chord tones, and the ambiguity between minor keys and their relative majors can all affect which features dominate locally. We retain this example to show both the interpretability and the limitations of the recovered orbits in realistic audio. Additional activation examples across musical genres are discussed in Section I.3.

Together, the POP909 and synthetic examples provide complementary evidence. The POP909 excerpts demonstrate that orbit semantics remain visible in realistic vocal music with complex arrangements, while the synthetic dataset isolates their responses to scale degree, mode, register, and instrument family.

H Ablation Studies

We study the effects of orbit alignment, foundation-model layer, and SAE sparsity.

H.1 s-SAE and Layer-Wise Comparison

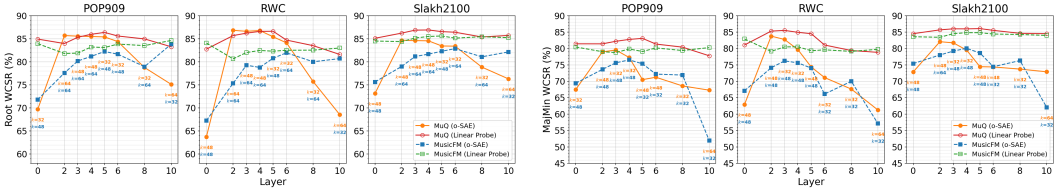
As an ablation of the proposed orbit-based SAE (o-SAE), we train a standard single Top-K SAE, denoted as s-SAE, using the same representations and training data. Unlike o-SAE, s-SAE does not use pitch-shifted views or shared-support alignment and therefore does not directly produce transposition-induced orbits.

For downstream inference, we retrieve semantic features from s-SAE using frozen linear probes. Let w_c denote the weight vector of probe output c and e_i the encoder vector of SAE feature i. We select

$$i^\star(c) = \arg\max_i \cos(w_c, e_i). \tag{42}$$

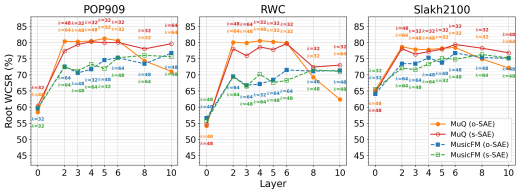
The retrieved features are ordered according to the probe outputs and processed using the same inference procedure as the corresponding o-SAE orbits. This matching is performed only after training and does not update the SAE parameters.

Because different representations may perform best at different sparsity levels, each layer-wise figure reports the best observed result across $K \in \{32, 48, 64\}$ for each representation, SAE type, layer, and metric. This provides a direct comparison across layers; results for all individual sparsity levels are reported in Section H.2.

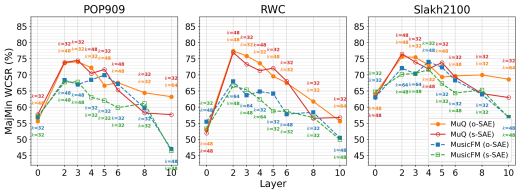


(a) Root WCSR under *Oracle Bd.*

(b) Majmin WCSR under *Oracle Bd.*

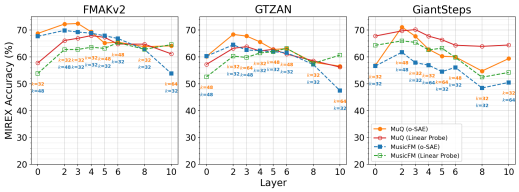


(c) Root WCSR under *Est. Bd.*

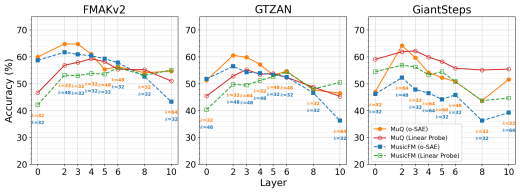


(d) Majmin WCSR under *Est. Bd.*

Figure 13: Chord recognition results.



(a) MIREX accuracy.



(b) Strict key accuracy.

Figure 14: Layer-wise key detection results derived from major- and minor-chord features.

Chord Recognition. Figure 13 compares the linear probe, s-SAE, and o-SAE using Root and MajMin WCSR on POP909, RWC-POP-100, and the Slakh2100 test split. For s-SAE, the 12 major- and 12 minor-chord probe outputs retrieve two pitch-class-ordered sets of SAE features, which are used in place of the recovered o-SAE chord rings.

Key Detection via the Major- and Minor-Chord Rings. Figure 14 compares key detection from chord-feature activations. For o-SAE, predictions are derived from the recovered major- and minor-chord rings; for s-SAE, they are derived from the two feature sets retrieved using the chord probe. We report MIREX and strict key accuracy on FMAKv2, GTZAN-Key, and GiantSteps.

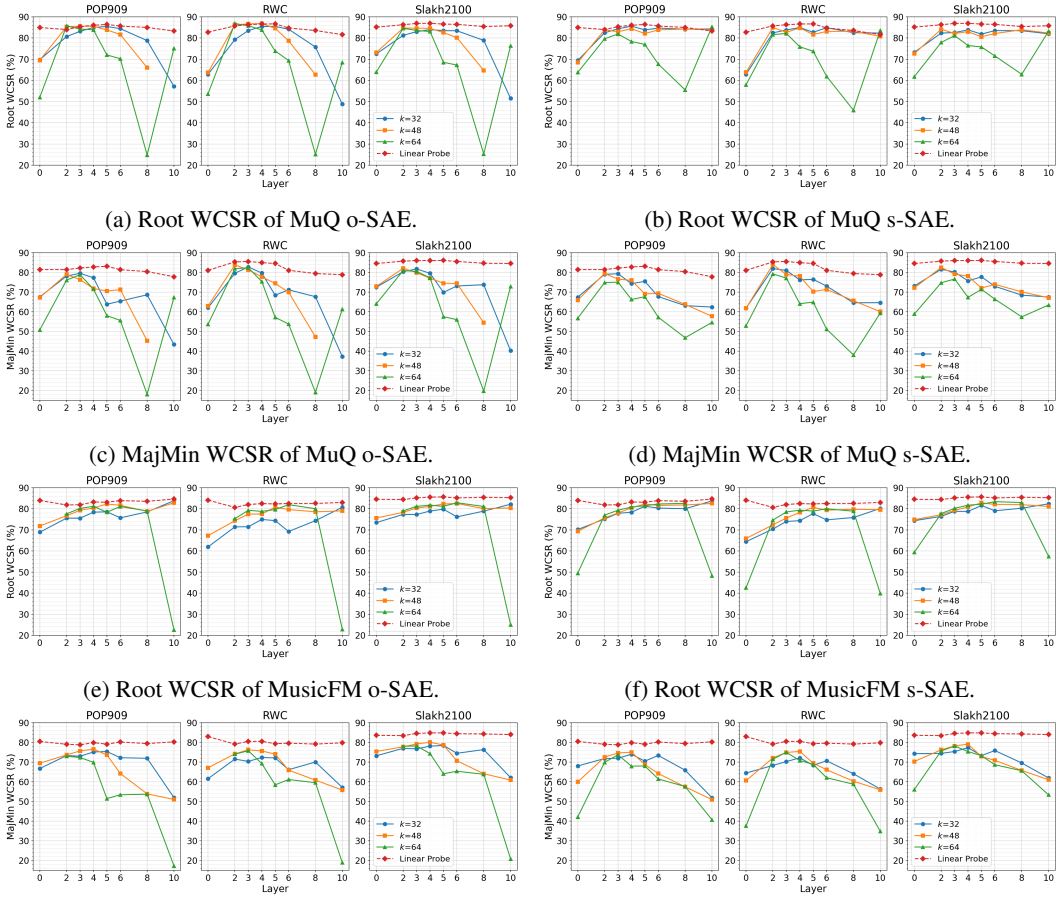
H.2 Effect of Top-K Sparsity

We study the effect of SAE sparsity by varying $K \in \{32, 48, 64\}$ while keeping the remaining training and inference settings fixed. Unlike the layer-wise comparison above, which reports the best observed result across K, the following figures show every sparsity setting separately. The corresponding linear probes are included as references and are independent of K.

Chord Recognition. Figure 15 reports Root and MajMin WCSR with oracle chord boundaries for o-SAE and s-SAE across MuQ and MusicFM layers. Each curve corresponds to one value of K, allowing us to compare how chord-feature quality changes with SAE sparsity across representations, layers, and datasets.

Key Detection via the Major- and Minor-Chord Rings. Figure 16 reports major–minor key detection derived from chord-feature activations. Results are shown separately for MuQ and MusicFM, o-SAE and s-SAE, and all three values of K. We report both strict key accuracy and MIREX accuracy.

Key Detection via the Subdominant Ring. Figure 17 shows key-signature detection from subdominant features for MuQ o-SAE and s-SAE. We report the three values of K separately across model layers, together with the corresponding subdominant probe as a reference.



(a) Root WCSR of MuQ o-SAE.

(b) Root WCSR of MuQ s-SAE.

(c) MajMin WCSR of MuQ o-SAE.

(d) MajMin WCSR of MuQ s-SAE.

(e) Root WCSR of MusicFM o-SAE.

(f) Root WCSR of MusicFM s-SAE.

(g) MajMin WCSR of MusicFM o-SAE.

(h) MajMin WCSR of MusicFM s-SAE.

Figure 15: Effect of Top-K sparsity on layer-wise chord-recognition performance with *Oracle Bd*. Results are shown for $K \in \{32, 48, 64\}$, with the linear probe included as a reference.

Table 7: Performance of the model trained on POP909

(a) Subdominant orbit: song-level key accuracy

Dataset	Accuracy (%)
GTZAN	54.81
GiantSteps	49.83
FMAKv2	59.26

(b) Major- and minor-chord orbits: WCSR

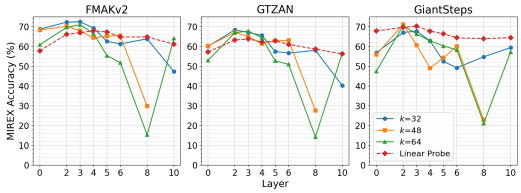
	Oracle Bd		Est. Bd	
Dataset	**Root**	**Majmin**	**Root**	**Majmin**
RWC	86.58	75.60	79.40	69.37
Slakh2100	84.31	75.91	78.95	71.17

H.3 Training Dataset Ablation

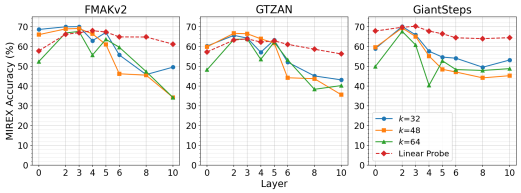
We additionally trained our aligned SAEs on a subset of real-world POP909 recordings, which contain vocals and complex production acoustics.We again recover the three core orbits: major chord, minor chord, and subdominant, under a setting of MuQ layer 2 and $k = 32$. The corresponding downstream results are shown in Table 7. All three core orbits still emerge and achieve comparable downstream performance, supporting the robustness of our discovery procedure across SAE-training distributions.

I Robustness Analysis

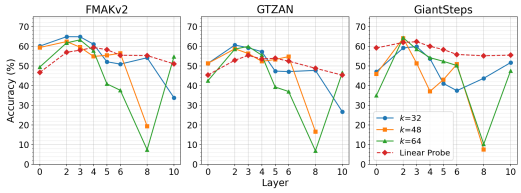
We further examine whether the discovered structures are robust to random initialization and variations in musical style.



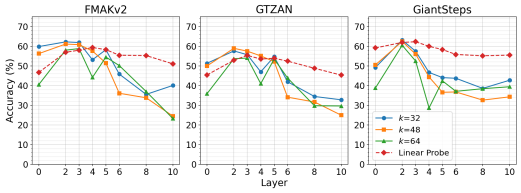
(a) MIREX accuracy of MuQ o-SAE.



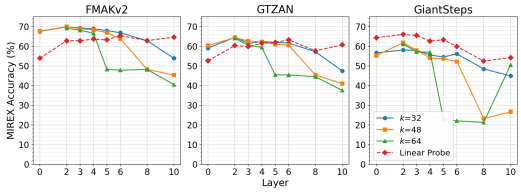
(b) MIREX accuracy of MuQ s-SAE.



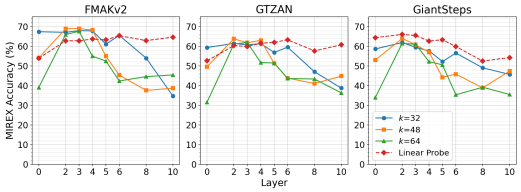
(c) Strict key accuracy of MuQ o-SAE.



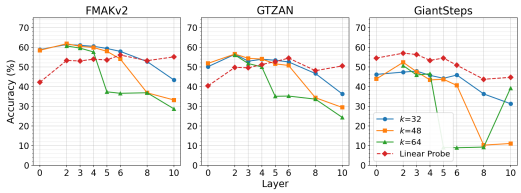
(d) Strict key accuracy of MuQ s-SAE.



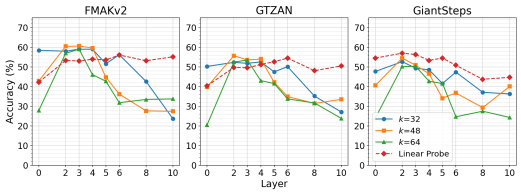
(e) MIREX accuracy of MusicFM o-SAE.



(f) MIREX accuracy of MusicFM s-SAE.

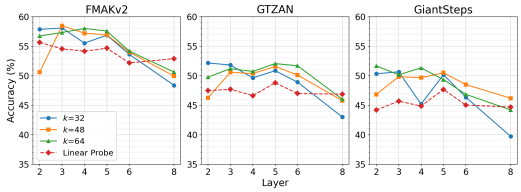


(g) Strict key accuracy of MusicFM o-SAE.

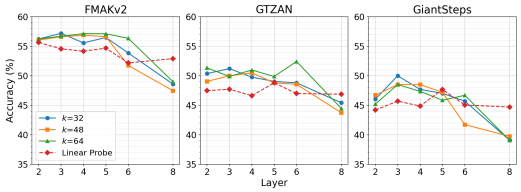


(h) Strict key accuracy of MusicFM s-SAE.

Figure 16: Effect of Top-K sparsity on layer-wise key-detection performance derived from chord features. Results are shown for $K \in \{32, 48, 64\}$, with the linear probe included as a reference.



(a) Key-signature accuracy of MuQ o-SAE.



(b) Key-signature accuracy of MuQ s-SAE.

Figure 17: Effect of Top-K sparsity on layer-wise key-signature detection from subdominant features in MuQ. Results are shown for $K \in \{32, 48, 64\}$, with the linear probe included as a reference.

I.1 Orbit-Aware Initialization

Each SAE decoder is initialized with Kaiming-uniform weights and normalized column-wise, while the corresponding encoder is initialized as the transpose of the decoder. All sub-SAEs share a common pre-encoding bias, which is initialized to the geometric median of sampled training representations.

For the multi-SAE model, the first sub-SAE serves as a randomly initialized anchor. We apply the anchor SAE to aligned frames from the original and pitch-shifted views. Suppose that anchor feature i has the largest activation for the original view and anchor feature j has the largest activation for a shifted view. The decoder vector of feature j is then accumulated to initialize feature i in the corresponding shifted sub-SAE, weighted by the geometric mean of their activation magnitudes.

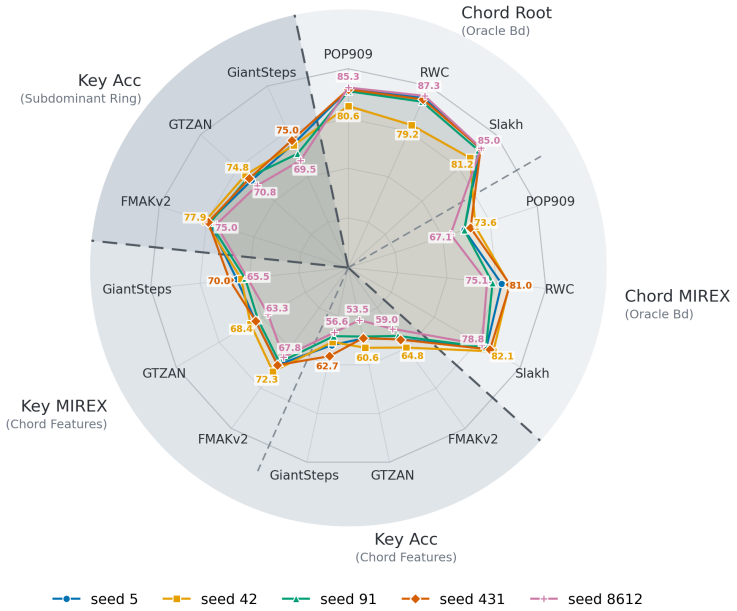


Figure 18: Downstream performance across random seeds for MuQ (2) o-SAE at $K = 48$.

Features with insufficient observations fall back to the corresponding anchor decoder vector with a small random perturbation. Finally, all decoder vectors are normalized, and the encoder weights are again initialized as their transposes.

This initialization does not assign fixed semantic identities to individual features, since the anchor dictionary remains randomly initialized. Instead, it establishes an initial correspondence among the sub-SAEs from paired pitch-shifted observations. Note that this initialization is not necessary for orbit emergence; rather, it improves the initial correspondence among sub-SAEs and reduces sensitivity to random initialization. Consequently, feature IDs may differ across runs, while the relative transposition structure is less sensitive to independent random initialization.

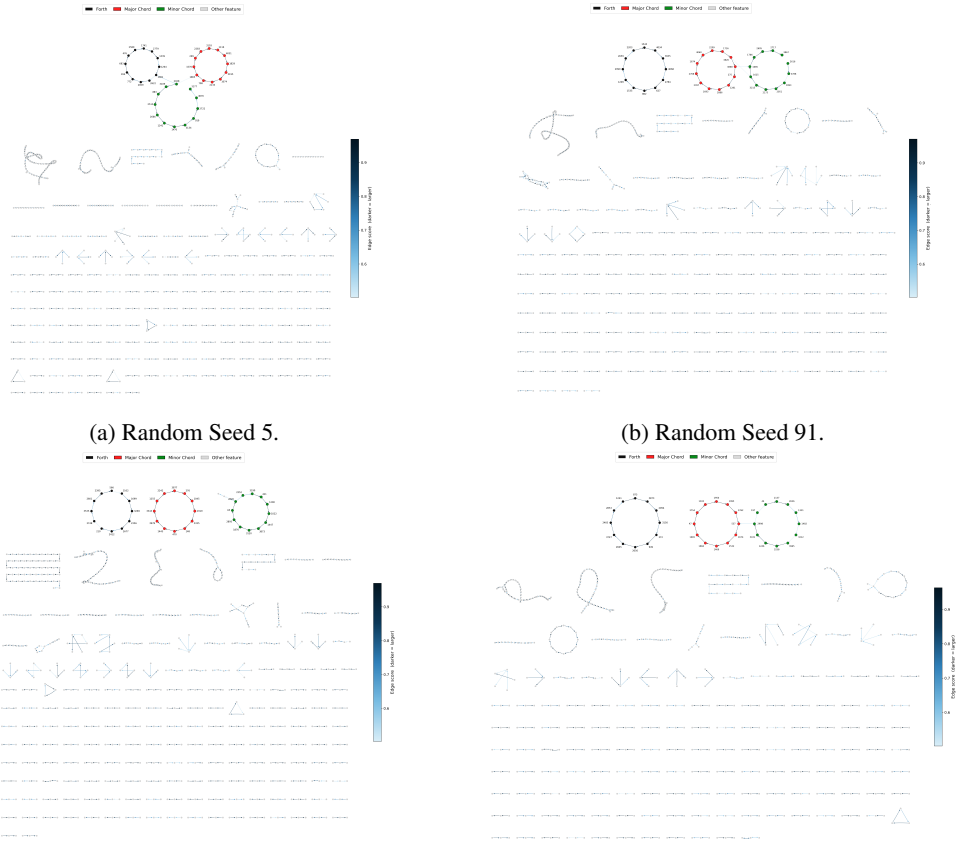
I.2 Robustness across Random Seeds

We train the same o-SAE configuration using multiple random seeds and compare its downstream performance on chord recognition, chord-derived key detection, and key-signature detection from the subdominant ring. Figure 18 summarizes the results for MuQ layers 2 at $K = 48$. Each polygon represents one random seed, and each axis corresponds to one metric–dataset pair. The similar performance profiles across seeds indicate that the downstream utility of the discovered features is not tied to a particular initialization.

We additionally examine structural robustness by visualizing the directed orbit graphs recovered from four random seeds. As shown in Figure 19, the exact feature IDs and local graph connections vary across runs, as expected under random initialization. Nevertheless, the major-chord, minor-chord, and subdominant structures remain identifiable in the graphs recovered with $n_e = 10$ and $\tau = 0.5$. This suggests that the core orbit semantics are reproducible up to feature permutation, even when the strict one-edge recovery procedure does not yield an identical graph for every seed.

I.3 Genre Sensitivity

We examine whether the discovered structures remain useful across different musical styles. We group the GTZAN-Key predictions by genre and compute the evaluation metrics independently for each group. We keep the same trained models, recovered orbits, and inference settings for all genres, without genre-specific feature selection or parameter tuning.



(a) Random Seed 5.

(b) Random Seed 91.

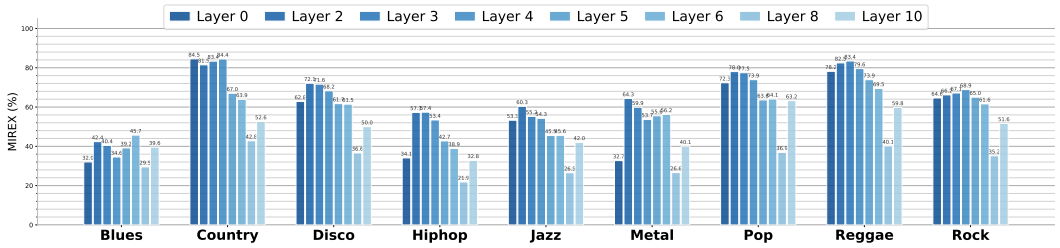
(c) Random Seed 431.

(d) Random Seed 8612.

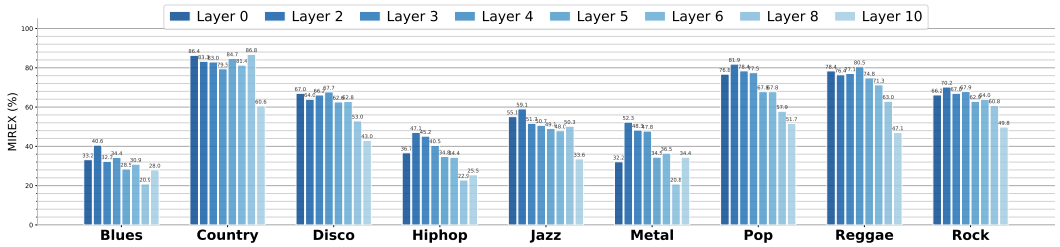
Figure 19: Directed orbit graphs recovered from MuQ across random seeds at $K = 48$, $n_e = 10$, and $\tau = 0.5$. Fixed points and components with fewer than three nodes are omitted. All seeds are sampled randomly one-shot.

Figure 20 reports chord-derived MIREX accuracy for MuQ and MusicFM, together with key-signature accuracy obtained from the MuQ subdominant ring. The recovered structures remain informative across all 9 GTZAN genres, although performance varies. Blues shows the lowest overall performance, whereas pop, country, and reggae generally achieve strong results.

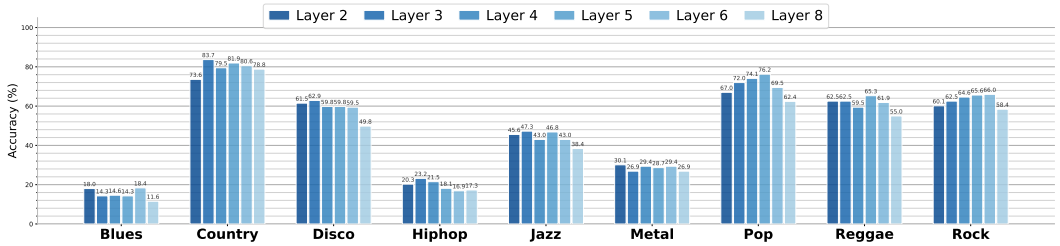
To complement these aggregate results, Figure 21 visualizes frame-level activations for excerpts from Gtzan Dataset. All examples use the same MuQ (2) o-SAE with $K = 32$ and the same orbits recovered at $\tau = 0.5$. We found that the activations of blues and hip-hop are harder to interpret than other genres, which is consistent with the key detection performance. Together, the quantitative and qualitative results suggest that the recovered orbit semantics generalize across musical styles, while remaining sensitive to genre-dependent acoustic and harmonic complexity.



(a) Chord-derived MIREX accuracy with MuQ.

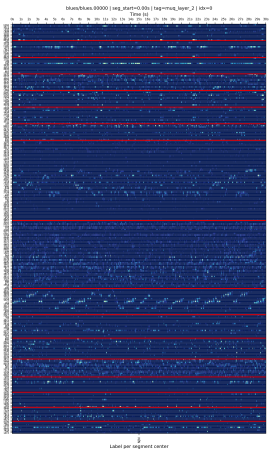


(b) Chord-derived MIREX accuracy with MusicFM.

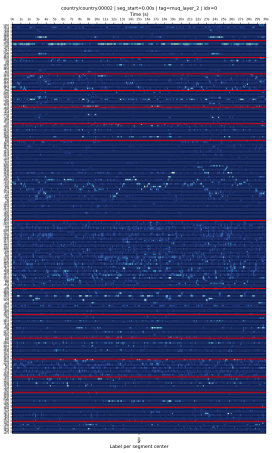


(c) Key-signature accuracy from the subdominant ring with MuQ.

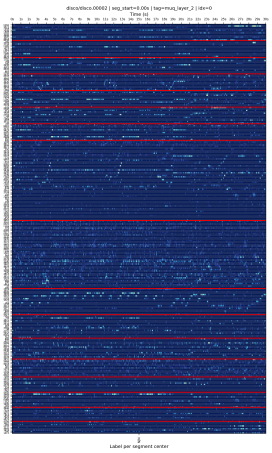
Figure 20: Genre-wise key-detection results on GTZAN-Key.



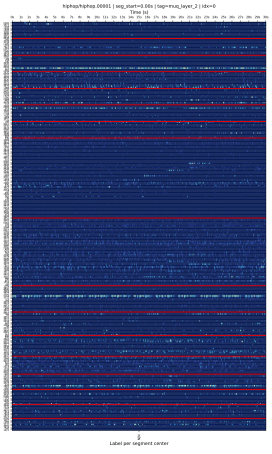
(a) Blues.



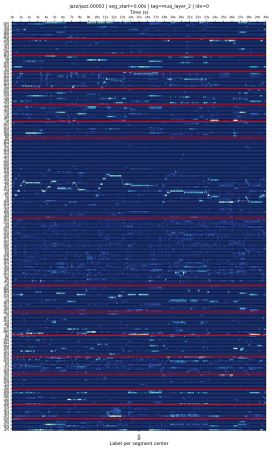
(b) Country.



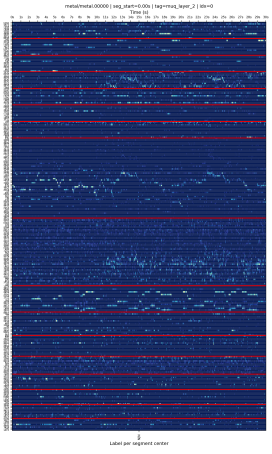
(c) Disco.



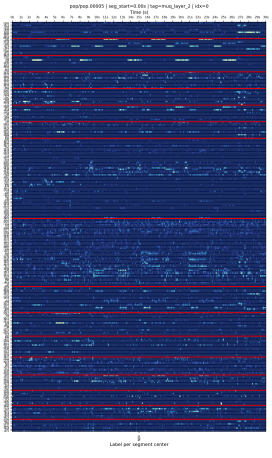
(d) Hiphop.



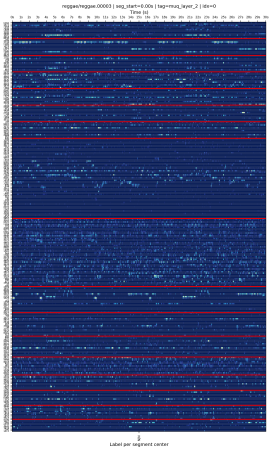
(e) Jazz.



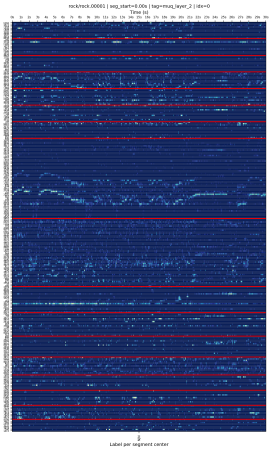
(f) Metal.



(g) Pop.



(h) Reggae.



(i) Rock.

Figure 21: Activations of songs across different genres.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract restates our formulated scope and contributions.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: Page limit; can add during revision.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [No]

Justification: Many unimportant assumptions in the math are intentionally left out in the text, to fit general practices of an AI paper. We internally analyze and discuss those assumptions.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Fully open source for reproduction.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Eventually yes — detailed and faithful instructions will be published on the Github repo. But at the time of writing, those materials are under preparation, with reviewer review as the deadline.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Even if the text missed out some details, the open source code is more than enough to fully characterize the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Cannot afford repeated experiments for each group in our DL setting.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: The compute resources used are low-scale anyways. Our proposed method is highly efficeint.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No foreseeable ill-use risks are associated with our methodology.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Aspects of societal impact of our work are either self-evident (making AI explainable) or elusive (eventual impact of explaining musicology from a technology method).

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: the paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We give proper citations to all external materials.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released source code and weights will be accompanied with documentation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: the core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.