Discovering and Steering Interpretable Concepts in Large Generative Music Models

Nikhil Singh[1*], **Manuel Cherep**[2*], **Pattie Maes**[2]
[1]Dartmouth College, [2]MIT. [*] Equal contribution.

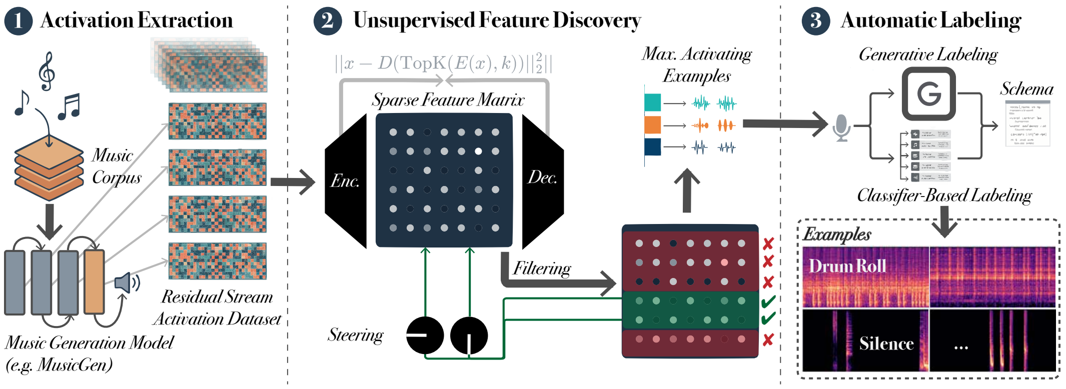

Figure 1: Multi-stage pipeline for discovering and steering interpretable concepts in generative music models. **(1)** Music from a large corpus is passed through a pre-trained generator to extract activations from multiple layers. **(2)** Sparse autoencoders reconstruct activations (also usable for steering), and features are filtered to retain the most viable candidates. **(3)** Retained features are characterized via musical examples and labeled using generative labeling with a multimodal LM and classifier-based labeling with pre-trained models.

Abstract

The fidelity with which neural networks can now generate content such as music presents a scientific opportunity: these systems appear to have learned implicit theories of such content's structure through statistical learning alone. This offers a potentially new lens on theories of human-generated media. When internal representations align with traditional constructs (e.g. chord progressions in music), they show how such categories can emerge from statistical regularities; when they diverge, they expose limits of existing frameworks and patterns we may have overlooked but that nonetheless carry explanatory power. In this paper, focusing on music generators, we introduce a method for discovering interpretable concepts using sparse autoencoders (SAEs), extracting interpretable features from the residual stream of a transformer model. We make this approach scalable and evaluable using automated labeling and validation pipelines. Our results reveal both familiar musical concepts and coherent but uncodified patterns lacking clear counterparts in theory or language. As an extension, we show such concepts can be used to steer model generations. Beyond improving model transparency, our work provides an empirical tool for uncovering organizing principles that have eluded traditional methods of analysis and synthesis.[1]

[1]https://musicdiscovery.media.mit.edu/

1 Introduction

When humans create, we draw upon a rich vocabulary of concepts. Some we can name and explicitly reason about; terms like "symmetry" or "gradient" in the visual arts, for instance. Others shape our media long before we have language for them: consider how the "Hero's Journey" narrative structure infused myths, legends, and epics across cultures for millennia before Joseph Campbell theorized it in the 20th century [1]. This gap between practice and theory is emblematic of a more fundamental pattern: our ability to recognize and use structure often precedes our ability to describe it.

This poses an interesting puzzle for machine learning: how do we bridge the gap between the raw statistical horsepower of generative models and the structured conceptual vocabulary we humans use? While these models demonstrate remarkable capacity to generate coherent content across domains through statistical learning alone, their neural activation patterns may not correspond directly to theoretical constructs. Music provides an ideal domain for investigating this alignment problem due to its combination of well-developed theoretical vocabulary with patterns resistant to verbal description, its statistical regularities alongside cultural diversity, and the absence of large-scale paired training data that facilitates concept discovery in other domains.

This paper introduces a method for doing exactly this (illustrated in Figure 1). Using sparse autoencoders, we extract features from the residual stream activations of a transformer-based music generation model. We emphasize that our work operates at a *proto-theoretical* level. It surfaces latent operational concepts from within a model's internal representations. These concepts may support coherent generations, but they lack the formal structure, completeness, or normative grounding required of a formal musical theory. Our goal is to investigate the distinctions that emerge when generative models are trained to produce music. These latent regularities may eventually serve for developing new formal theories. Please refer to Section A for an extensive review of related work.

2 Methods

2.1 Dataset and Activation Extraction

We use the *MusicSet* [2] dataset, a collection of around 160,000 samples, most of which are $\approx$10 seconds long. MusicSet is a combination of data from MTG-Jamendo [3], MusicCaps [4], and MusicBench [5]. This dataset offers a diverse range of genres, instrumentation, and musical characteristics at a larger scale than many alternative publicly available datasets. We produce activations by feeding the music into two pre-trained MusicGen [6] models: MusicGen-Large (MGL) and MusicGen-Small (MGS). For each model, we extract activation vectors from five layers in the residual stream: the early layer (layer 2), late layer (second-to-last), middle layer, and layers at 25%, 50%, and 75% depth. This corresponds to $\textit{Layer}_{\text{MGS}} \in \{2, 6, 12, 18, 22\}$, and $\textit{Layer}_{\text{MGL}} \in \{2, 12, 24, 36, 46\}$. Activation dimensionality is 1024 for MGS and 2048 for MGL.

2.2 Training the Sparse Autoencoders

We train SAEs to extract interpretable features from the residual stream activations of MusicGen. Each is trained to reconstruct the original activations $\mathbf{x} \in \mathbb{R}^d$ from a learned sparse latent representation $\mathbf{h} \in \mathbb{R}^{EF \cdot d}$, where EF is an expansion factor. The SAE architecture generally consists of an encoder ($\mathbf{h} = \text{ReLU}(\mathbf{W}_e\mathbf{x} + \mathbf{b}_e)$) and a decoder ($\hat{\mathbf{x}} = \mathbf{W}_d\mathbf{h} + \mathbf{b}_d$), both implemented as single linear layers where $\mathbf{W}_e \in \mathbb{R}^{EF \cdot d \times d}$, $\mathbf{W}_d \in \mathbb{R}^{d \times EF \cdot d}$, $\mathbf{b}_e \in \mathbb{R}^{EF \cdot d}$, and $\mathbf{b}_d \in \mathbb{R}^d$ are the learned weights and biases. We apply a k-sparse projection operator $P_k : R^{EF \cdot d} \to R^{EF \cdot d}$ to h, yielding the sparse latent code $z = P_k(h)$. These are typically called k-sparse autoencoders [7, 8]. Where S_k is the set of indices of the top-k values $\in$ h: $z = P_k(h) = h_i$ if $h_i \in S_k(\mathbf{h})$ and 0 otherwise.

We minimize the mean squared reconstruction error, subject to the sparsity constraint. We experiment with expansion factors $EF \in \{4, 32\}$ and sparsity levels $k \in \{32, 100\}$, all of which are values encountered in prior SAE literature [8, 9].

2.3 Feature Mapping and Filtering

SAE latents may correspond to meaningful musical features, but interpreting them is challenging since music tokens are only $\sim$20 ms chunks. Therefore, we compute a per-track summary statistic for each feature based on its activation profile over time. Let f_i denote the i-th learned feature, and let $\alpha_{i,j} = (\alpha_{i,j,1}, \dots, \alpha_{i,j,T_j}) \in \mathbb{R}^{T_j \times d}$ represent its activation time series over the T_j time steps of track j. The *mean activation* of feature f_i in track j is: $\mu_{i,j} = \frac{1}{T_j} \sum_{t=1}^{T_j} \alpha_{i,j,t}$, which corresponds to the average prominence over the duration of the track. We then filter features by their corpus-level activation rate $r_i = \frac{1}{N} \sum_{j=1}^{N} \delta_{i,j}$ where $\delta_{i,j} = \mathbb{I}\left(\max_{1 \le t \le T_j} \alpha_{i,j,t} > \tau\right)$ indicates whether feature f_i activates in track j, and N is the number of validation tracks. We retain only those with $0.01 < r_i < 0.25$ to exclude inactive, overly common, or too rare features. Then, we select the top 10 highest-activating examples. Thresholds are informed by [9].

2.4 Automated Interpretability

Human interpretation of features quickly becomes infeasible due to scale: with expansion factor 32x, a single layer may contain 65,536 features requiring 18+ hours of listening time, multiplied across multiple layers. Since musical concepts present more variedly than linguistic ones and lack strong music-language models for automated interpretation, we employ a multi-step labeling pipeline: (1) querying a multimodal model (Gemini Flash 1.5) with concatenated audio from top-10 max activating examples to produce a set of concept tags, confidence scores for each, and an overarching label and description for the feature; (2) extracting candidate labels using pre-trained Essentia [10] audio classifiers; and (3) using CLAP [11] to measure alignment between labels and activating examples.

2.5 Human Validation

We run a human validation study (IRB approved) where participants are presented with 3 audio examples for each feature randomly subselected from SAEs with significant valid feature counts, alongside proposed labels. They are then asked to rate which label best captures common features and their confidence in that selection, while also providing their own intuitive labels.

2.6 Generation Steering

Steering generations modifies the forward pass by adding scaled feature vectors to the residual stream at the SAE's hook point. For a given SAE feature j, we perform steering by adding the corresponding decoder weight vector $\mathbf{W}_{d,j}$ to the residual stream activations during generation: $\mathbf{x}' = \mathbf{x} + \alpha \cdot \beta \cdot \mathbf{W}_{d,j}$ where $\mathbf{x}$ are the original residual stream activations, $\alpha \in (0, 1)$ is the steering strength, and β is the maximum activation strength for feature j computed over the feature j's max. activating examples.

Experimental Setup. We evaluate steering using a neutral prompt, "Simple melody" (as in [12]) for steering strengths $\alpha \in \{0.0, 1.0\}$. We used *MusicGen-Large* SAEs with exp. factor 32, $k \in \{32, 100\}$, and middle to late layers (24, 36, and 46), based on feature statistics.

Feature Selection and Evaluation. We identify the most steerable features by computing the cosine similarity of CLAP embeddings [11] for each feature using cosine similarity between the feature's top 10 activating examples and (1) the steered example ($\alpha = 1$), and (2) the baseline ($\alpha = 0$).

3 Results

3.1 Statistics of Discovered Features

Before Filtering Before filtering, feature activations show a heavy-tailed distribution: some fire in many tracks, while most activate only rarely (see Figure 4). SAE hyperparameters appear to mediate this balance, i.e. larger k or expansion factor (EF) increase overall recovery but also changes the activation frequency distributions. This imbalance makes raw feature counts misleading and motivates our explicit filtering approach, to isolate features that are both selective and interpretable.

After Filtering Table 1 reports the number of features that survive filtering across models, layers, and SAE settings. For MGL, certain configurations yield thousands of retained features, while MGS rarely produces more than 100. Expansion factor also has a predictable effect: higher EF values increase feature counts, reflecting the greater representational pressure imposed by the wider latent space. Interestingly, we find that this combined well with *larger* k, possibly suggesting that musical clips tend to have many co-occurring activating features (e.g. in different layers and at different time-steps). Early layers (e.g. L2) largely produce more features than later ones, though interpretability does not necessarily track count. The consistent difference between MGL and MGS indicates that scale does more than add parameters: it appears to alter the internal organization of representations in ways that make interpretable structure more extractable.

3.2 Concept Representation

Do later layers encode more interpretable features? Figure 3 shows average CLAP scores between feature audio and their automatically generated labels, stratified by layer. For MGL, deeper layers yield higher scores, indicating that their features are more readily aligned with human-interpretable concepts. This mirrors probing results from [13], which showed that musical properties such as chords and roots are more strongly encoded in deeper transformer layers. Our results generalize that observation: later layers not only encode specific theory-driven constructs more clearly, but also produce features that can be labeled more consistently across diverse concepts.

Does model scale change how features are organized across layers? To test whether larger models produce more layer-specific features, we trained a simple MLP probe to predict the layer of origin from a feature's activation profile. **Accuracy was substantially higher for MGL (**50.29 ± 3.41**) than for MGS (**40.51 ± 3.60**).** This means that features in MGL are more distinctive by layer, whereas those in MGS reflect a less differentiated representational structure. These results extend probing-based findings on synthetic datasets [14]: scale not only increases the number of recoverable features, but also sharpens the division of representational roles across layers.

3.3 Automated Interpretability

CLAP Scores We assessed label quality using CLAP [11] alignment between feature audio and assigned labels. Figure 6 shows the distribution of maximum scores across all SAEs. Essentia tags achieve stronger alignment than Gemini with considerable overlap between the two. Overall, no single labeling strategy dominates. CLAP scores also provide a practical mechanism for filtering: thresholds can be tuned to trade off label quality against feature coverage without exhaustive manual review, though the optimal cutoff may depend on corpus and task.

Human Evaluation To compare methods directly, we evaluated 400 features per pipeline with 80 participants (40 per method, 10 features each). Participants chose the best label, rated confidence, and could supply alternative labels. Confidence was higher for Essentia (3.96/5; 71% $>$4) than for Gemini (3.19/5; 47% $>$4), suggesting that while multimodal LLMs can generate open-ended labels beyond fixed taxonomies, classifier-based tags may be more reliable in practice.

3.4 Steering with Discovered Concepts

Table C.1 summarizes results. Depending on SAE configuration, between 15–35% of tested features showed improved CLAP alignment with their top-activating examples under steering, starting from the prompt "Simple Melody." While our primary goal in this work remains feature discovery [15], a non-trivial fraction of discovered features correspond to causal, manipulable directions in activation space even under the automatic labeling scheme. This shows that interpretable features are not only descriptive, but in many cases actionable for controlling outputs. Figure 7 illustrates such cases, where steering shifts outputs toward the intended feature class while holding prompt and seed fixed.

4 Conclusion

Our results demonstrate the feasibility and potential of using sparse autoencoders to discover interpretable concepts in large music generative models, i.e. using models for *musical discovery* [16]. We have shown that these models learn a rich set of musical patterns, ranging from well-established concepts to emergent regularities that warrant further investigation. Through future work in this area, we believe it is possible to build a comprehensive understanding of how neural networks represent and generate music. Our pipeline offers a scalable path towards this goal.

Acknowledgments

We gratefully acknowledge AWS for providing computational resources through the AWS Research and Engineering Studio (RES) pilot program. We thank Brian McCarthy and Jared Novotny for their valuable support, and Peter Fisher (MIT ORCD) for making this collaboration happen. MC is supported by a fellowship from “la Caixa” Foundation (ID 100010434) with code LCF/BQ/EU23/12010079. We also thank Michael Casey, Anna Huang, and the HAI-Res lab for supportive comments.

References

[1] Joseph Campbell. The hero with a thousand faces. *New World Library*, 1949.

[2] Shaopeng Wei, Manzhen Wei, Haoyu Wang, Yu Zhao, and Gang Kou. Melody-guided music generation. 2024.

[3] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. The mtg-jamendo dataset for automatic music tagging. In *International Conference on Machine Learning*, 2019.

[4] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

[5] Jan Melechovský, Zixun Guo, Deepanway Ghosal, Navonil Majumder, Dorien Herremans, and Soujanya Poria. Mustango: Toward controllable text-to-music generation. In *North American Chapter of the Association for Computational Linguistics*, 2023.

[6] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[7] Alireza Makhzani and Brendan Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

[8] Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

[9] Elana Simon and James Zou. Interplm: Discovering interpretable features in protein language models via sparse autoencoders. *bioRxiv*, pages 2024–11, 2024.

[10] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José Zapata, and Xavier Serra. Essentia: an open-source library for sound and music analysis. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 855–858, 2013.

[11] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[12] Dana Arad, Aaron Mueller, and Yonatan Belinkov. Saes are good for steering–if you select the right features. *arXiv preprint arXiv:2505.20063*, 2025.

[13] Wenye Ma, Xinyue Li, and Gus Xia. Do music llms learn symbolic concepts? a pilot study using probing and intervention. In *Audio Imagination: NeurIPS 2024 Workshop AI-Driven Speech, Music, and Sound Generation*, 2024.

[14] Megan Wei, Michael Freeman, Chris Donahue, and Chen Sun. Do music generation models encode music theory? *arXiv preprint arXiv:2410.00872*, 2024.

[15] Kenny Peng, Rajiv Movva, Jon Kleinberg, Emma Pierson, and Nikhil Garg. Use sparse autoencoders to discover unknown concepts, not to act on known concepts. *arXiv e-prints*, pages arXiv–2506, 2025.

[16] Nikhil Singh, Manaswi Mishra, and Tod Machover. Ai for musical discovery. 2024.

[17] Darrell Conklin. Music generation from statistical models. In *Proceedings of the AISB 2003 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, pages 30–35. Citeseer, 2003.

[18] Nori Jacoby, Naftali Tishby, and Dmitri Tymoczko. An information theoretic approach to chord categorization and functional harmony. *Journal of New Music Research*, 44(3):219–244, 2015.

[19] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. *arXiv preprint arXiv:1809.04281*, 2018.

[20] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, et al. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 12, 2016.

[21] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model. *arXiv preprint arXiv:1612.07837*, 2016.

[22] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023.

[23] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. In *International Conference on Machine Learning*, pages 13916–13932. PMLR, 2023.

[24] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, et al. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. transformer circuits thread, 2024.

[25] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

[26] Stéphane Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.*, 41:3397–3415, 1993.

[27] Wenye Ma and Gus Xia. Exploring the internal mechanisms of music llms: A study of root and quality via probing and intervention techniques. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024.

[28] Marcel A Vélez Vásquez, Charlotte Pouw, John Ashley Burgoyne, and Willem Zuidema. Exploring the inner mechanisms of large generative music models. *ISMIR*, 2024.

[29] Anna Langedijk, Hosein Mohebbi, Gabriele Sarti, Willem Zuidema, and Jaap Jumelet. Decoderlens: Layerwise interpretation of encoder-decoder transformers. *ArXiv*, abs/2310.03686, 2023.

[30] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020.

[31] Sukrut Rao, Sweta Mahajan, Moritz Böhle, and Bernt Schiele. Discover-then-name: Task-agnostic concept bottlenecks via automated concept discovery. In *European Conference on Computer Vision*, pages 444–461. Springer, 2024.

[32] David Temperley. *The cognition of basic musical structures*. 2004.

[33] David Temperley. *Music and probability*. 2007.

[34] Robert O Gjerdingen. Categorization of musical patterns by self-organizing neuronlike networks. *Music Perception*, 7(4):339–369, 1990.

[35] Robert Gjerdingen. *Music in the galant style*. OUP USA, 2007.

A Related Work

A.1 Statistical Learning of Musical Structure

Computational approaches to modeling music span rule-based systems, probabilistic models, and modern neural networks. Early Markov models captured local dependencies like chord transitions but failed at longer-range structure [17], while information-theoretic analyses have revealed statistical bases for harmonic categories [18]. For several years, neural architectures have achieved impressive results in both symbolic and audio modeling [19, 20, 21]. More recent systems often model audio directly with transformers or diffusion approaches [4, 6, 22, 23]. Yet as these models generate increasingly sophisticated music, our understanding of what musical knowledge they acquire lags behind. The progress in generation has coincided with greater opacity of internal structure, motivating our central question: **what do these models actually learn about music?** Our work seeks to make progress on answering this question.

A.2 Interpretability Through Feature Extraction

Interpreting neural network computations has long been a central goal. Recent work shows that LLM representations often encode concepts in surprisingly localized ways [24], echoing classic sparse coding results where mammalian visual cortex was modeled as learning efficient codes for natural images [25]. This broader lesson, namely that sparse latent structure can be computationally and biologically advantageous, has motivated the development of interpretability methods. A recent line trains *sparse autoencoders* (SAEs) [26, 24] on transformer activations, enforcing sparsity to extract interpretable components. Formally, with encoder E and decoder D, one optimizes:

$$\min_{E,D} \mathbb{E}_x[\|x - D(E(x))\|_2^2 + \lambda\|E(x)\|_1] \tag{1}$$

This constraint is thought to encourage discovery of reusable "atomic" concepts rather than memorized reconstructions. Learned features can then be labeled using automated methods (e.g. LLMs applied to maximally activating sequences). Extending such approaches to music generators is non-trivial, given their hierarchical temporal structure and mixed discrete–continuous features. **Our work develops adaptations to apply these principles to the musical domain.**

A.3 Interpreting Music Generation Models

Given a vocabulary of known concepts and example data to match, one strategy toward interpreting music generation models is *probing*. While probing has a rich history in NLP, work has recently begun to apply it to music models. Prior work [14] introduced the *SynTheory* dataset to systematically probe music foundation models on core music theory concepts like tempo, intervals, and chords, revealing that certain layers capture these concepts more strongly than others. Similarly, other work investigates chord and pitch representations in MusicGen and find that musical concepts become more evident in deeper layers, yet may not be linearly separable without additional interventions in the hidden space [27]. Beyond these direct probing and manipulation techniques, recent work [28] uses the *DecoderLens* [29] method to interpret intermediate activations, offering an auditory perspective on what each layer "hears" and how it evolves through the network. Such approaches exemplify an emerging trend: there is increasing interest in understanding how these models do what they do. However, these approaches have primarily focused on probing for known concepts, an important but limited strategy. This raises a key question: what structures might we be missing by focusing only on concepts we already know to look for? To answer this, we propose an unsupervised concept *discovery* pipeline. **In summary, probing asks "do models encode X concept we already know?", while our pipeline asks "what concepts do models encode?" without supervision.**

A.4 Automated Concept Discovery and Evaluation

A central challenge in interpretability is to discover concepts a model has learned organically, rather than imposing them *a priori*. Recent frameworks address this across domains. InterPLM [9] trains sparse autoencoders on a protein language model (ESM-2), uncovering thousands of latent features, many aligning with known biological concepts and others appearing novel. Concept Bottleneck Models (CBMs) [30] instead constrain intermediate layers to discrete, named concepts, though typically with a hand-specified set. Recent variants invert this paradigm, using SAEs to first *discover* latent concepts and then *name* them automatically or semi-automatically [31]. While developed outside of music, these principles are broader: generative music models likely encode diverse concepts without explicit labels. Automated discovery and evaluation of such features could reveal how these models "understand" music, and enable finer control over generated outputs.

B Examples of Concepts

B.1 Examples of Finding Canonical Musical Concepts

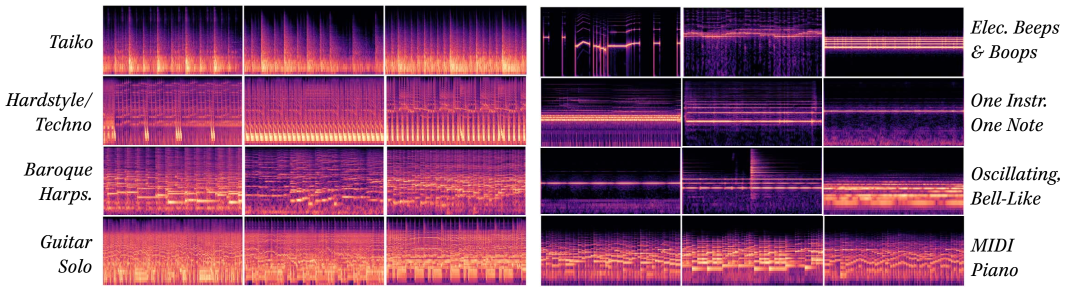

Figure 2: Examples of features discovered using the sparse autoencoders we train. Note: these examples are labeled manually. Spectrograms highlight similarities across examples within a concept.

Retained features often correspond directly to established musical constructs, showing us how these are encoded within the model. Figure 2 (left) shows manually annotated examples from MGL:

1. **Taiko Drums** *(k=32, EF=4)*: activates on resonant Taiko and other low-pitched drums, isolating a timbral family recognizable in theory and practice.
2. **Hardstyle Techno** *(k=32, EF=32)*: consistently aligns with the hardstyle subgenre, capturing a distinct rhythmic and production signature.
3. **Baroque Harpsichord** *(k=32, EF=32)*: isolates the plucked timbre and contrapuntal texture of harpsichord repertoire, as well as similar stylistic cues in Baroque string writing.
4. **Rock Guitar Solos** *(k=32, EF=4)*: activates on electric guitar solos, integrating timbre, ornamentation, and melodic phrasing characteristic of the style.

These cases demonstrate that SAEs can discover categories that existing vocabulary already recognizes as meaningful. The match between latent features and canonical concepts indicates that the activations are not arbitrary artifacts but can and do internally encode distinctions salient to musicians and listeners in ways we can localize and recover from their internal activations.

B.2 Examples of Emergent Musical Regularities

Not all features map neatly to established verbal categories. Some capture patterns that are perceptually coherent but poorly described in existing theory (Figure 2, right):

1. **Electronic Beeps and Boops** *(k=32, EF=4)*: activates on diverse synthetic tones and glitches, a class of sounds central to electronic genres but not theoretically well-defined.
2. **Single Instrument, Single Note** *(k=32, EF=32)*: isolates sustained single-note events across instruments, suggesting that the model may have detectors for basic atomic units of musical *texture*.

3. **Oscillating Bell-like Timbres** ***(k=32, EF=32)***: responds to bell-*like* sounds with beating or oscillations, pointing to sensitivity to fine-grained spectral phenomena.

4. **Romantic Poppy MIDI Piano** ***(k=32, EF=4)***: activates on MIDI piano in pop-ballad contexts, apparently tuned to performance artifacts like rigid quantization and compressed dynamics that co-occur with the poppy, romantic stylistic frame.

These examples surface coherent musical regularities that are not fully described by standard musical terminology likely to be found in prompts or specified in probing experiments. They show that models are not limited to reproducing canonical categories but also carve out distinctions that reflect production practices, timbral subtleties, or emergent stylistic groupings. Identifying such features provides concrete evidence that generative models encode structure beyond what human-led interpretability, such as probing, is likely to anticipate. In turn, analyzing situations in which these features arise may prove useful for building them into coherent theoretical constructs when studied in detail over time by music theorists.

C Additional Results: Statistics of Discovered Features

C.1 Tables

Table 1 reports the number of features that survive filtering across models, layers, and SAE settings. Section C.1 shows the proportion of features per SAE with positive steering improvement.

Table 1: Feature counts across models, layers, and SAE configurations (total 4697). For each model layer, the configuration providing the max. num. of features is **bolded**.

		MusicGen Large					**MusicGen Small**				
Exp. F	*k*	L2	L12	L24	L36	L46	L2	L6	L12	L18	L22
4	32	12	0	4	4	3	0	0	0	0	0
32	32	30	117	149	**135**	71	3	7	**4**	**7**	9
4	100	407	109	147	28	25	6	3	0	0	0
32	100	**2344**	**222**	**412**	131	**177**	**59**	**47**	**4**	4	**17**

Table 2: Proportion of features per SAE with positive steering improvement.

Model	Exp. F	*k*	Layer	Steering Improvement
MGL	32	100	24	96/408 (23.5%)
MGL	32	100	36	46/131 (35.1%)
MGL	32	100	46	27/177 (15.3%)
MGL	32	32	24	44/149 (29.5%)
MGL	32	32	36	39/135 (28.9%)
MGL	32	32	46	16/71 (22.5%)

C.2 Visualizations

Figure 3 shows the average CLAP scores across layers. Figure 4 presents feature activation statistics. For each SAE, we plot the relationship between the average activation strength of each feature (y-axis) and its prevalence, as defined by the fraction of the validation set for which the feature's mean activation is positive (x-axis). Figure 5 visualizes the results from Table 1 for an alternate view, to facilitate overall comparisons. Figure 6 shows the distribution of maximum scores across all SAEs. Figure 7 shows examples of steered features.

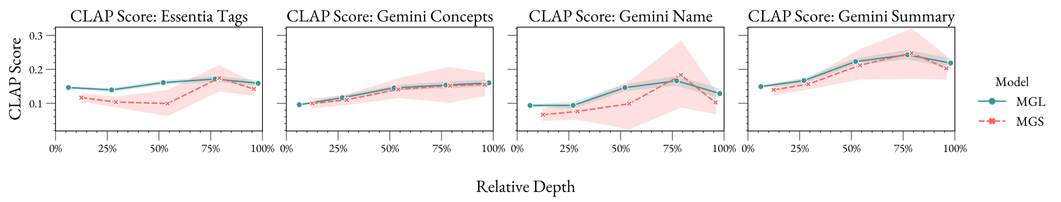

Figure 3: Avg. CLAP [11] score across layers, comparing feature audio to automatic concept labels. For MGL, later layers appear to produce more interpretable features on average.

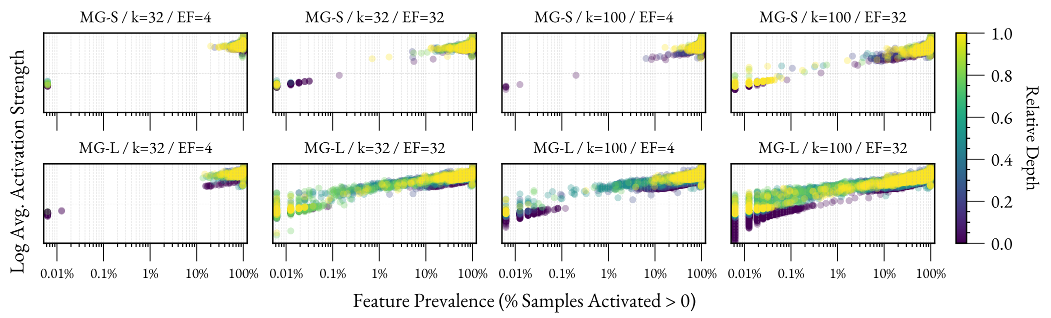

Figure 4: Distribution of learned feature activation characteristics across the MusicSet validation cohort, prior to filtering, for various SAE configurations (log-log plot). The x-axis shows feature prevalence (fraction of validation tracks on which a feature exhibits non-zero mean activation). The y-axis indicates mean activation strength. The observed heavy-tailed distributions, with many features exhibiting either very broad or very sparse activation patterns, suggest the need for a more principled filtering method.

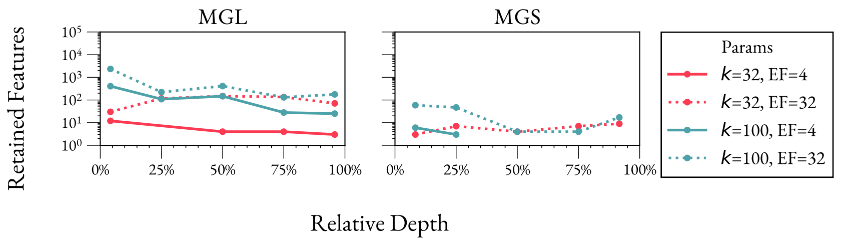

Figure 5: Impact of SAE hyperparameters and target model on yield of viable musical features after filtering. Each bar represents the number of features retained for a specific configuration, varying by MusicGen model size (small or MGS/large or MGL), target layer, k-sparsity, and expansion factor (EF). The substantial variation in feature counts shows how these parameters affect discovery of statistically robust features.

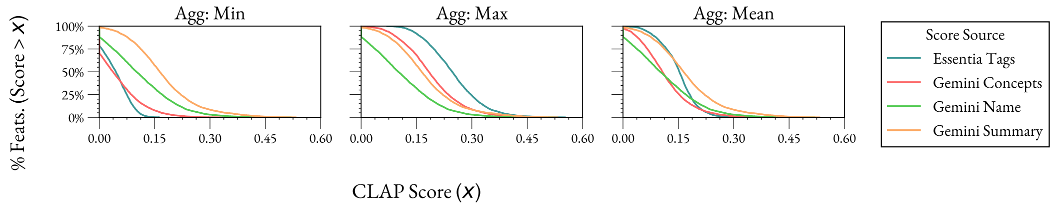

Figure 6: Distribution of max. CLAP scores across all SAEs. Pooling both Gemini- and Essentia-produced labels, we score them using CLAP, showing the trade-off between confidence and coverage at different potential filter levels.

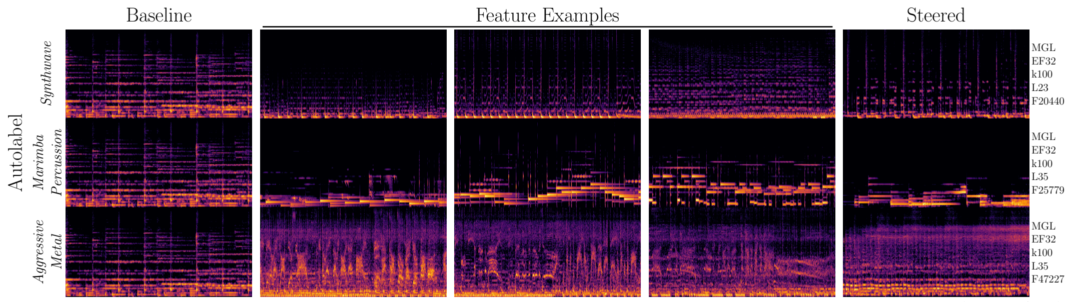

Figure 7: Examples of *steered* features. Note: these examples were labeled *automatically*. **(Baseline)** Generation without steering for "Simple melody." **(Feature Examples)** Top max. activating examples for the steering feature. **(Steered)** Generation steering with the same prompt and seed as the baseline, and maximum strength empirically calculated from the maximum activations. The steering shows close alignment with the feature examples, as seen in the spectrograms.

D Choice of Captioning Model

We compared several versions of Gemini models as a supplementary analysis, specifically *Flash 1.5*, *Flash 2.0*, *Flash 2.5*, and *Pro 2.5*, as automated labeling pipelines. We conducted a preliminary evaluation of label quality using the CLAP score [11].

Figure 8 shows average CLAP scores per MusicGen layer across models. Figure 9 presents the ECDF of maximum CLAP scores over all labeled features. Surprisingly, Gemini Flash 1.5 consistently produces the highest CLAP alignment on the whole, despite being the oldest model tested.

Given these results, although we acknowledge there are significant limitations arising from using CLAP as the evaluation metric, we use Flash 1.5 as the default captioning model for all automatic labels in the paper's primary analyses.

E Prompt-Conditioned Activations

Our primary focus in this work was to analyze the musical representations that emerge *without* text conditioning. These representations are especially important because they capture how the model organizes musical structure in its own representational space, rather than simply responding to verbal instructions. Whereas prompts are already verbal and thus relatively easy to align with human categories, internal musical features may be tacit, difficult to describe, and poorly characterized in existing theory. By targeting this layer of representation, we aim to surface structures that extend beyond what can be easily named, and therefore hold greater potential for discovery.

We see text-conditioned analysis as a valuable future direction, but our findings suggest that it is not tractable under current conditions. First, open datasets with paired text–music examples remain limited. Captions are often sparse, noisy, or stylistically inconsistent, and prompt–audio alignment is insufficient to support reliable, large-scale analysis. This means that conclusions about text-conditioned representations would be confounded by dataset artifacts rather than model behavior. Second, from a methodological standpoint, isolating the contribution of text requires ablation and control techniques that go beyond existing ones. Mechanistic interpretability has not yet converged on reliable tools for partitioning or disentangling multi-modal conditioning, making this a substantial open problem.

Addressing these challenges would require both new datasets with carefully designed prompt–audio correspondence and new methodological advances to interpret conditioning effects in multimodal models. We believe this is a promising research agenda, but it demands significant additional work beyond the scope of this paper. Our contribution here is therefore to establish a foundation by studying the non-textual musical representations directly, which offers both theoretical value, in revealing the categories that emerge independently of language, and practical value by providing a baseline for future multimodal interpretability.

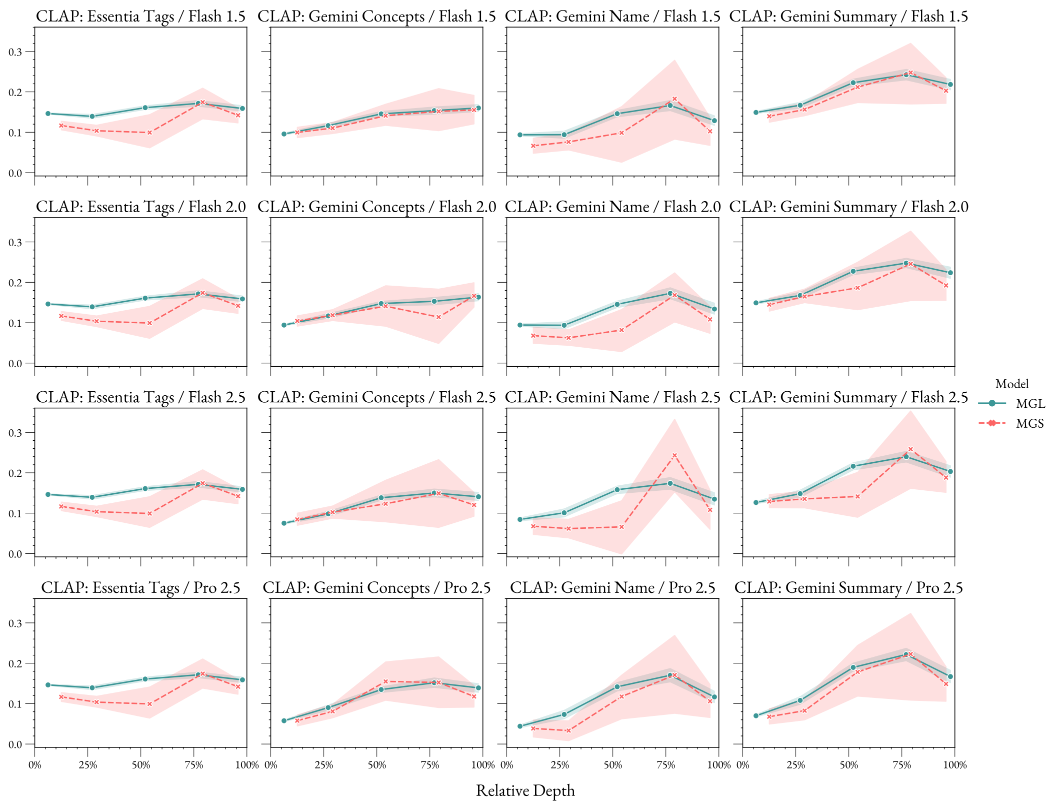

Figure 8: Average CLAP scores by MusicGen layer for each Gemini model used as the captioning pipeline. Gemini Flash 1.5 achieves the highest alignment on average.

F Instructions Given to Participants

Study Details

We are assessing the quality of a novel method for discovering interpretable concepts in generative music models. You will be presented with a series of short sounds with a common pattern or feature. Given a set of options, you will be asked to select the one that best describes all the sounds, as well as your confidence in your prediction.

[FOR EACH SAMPLE]

Select the closest category representing what all these sounds have in common:

[Audio 1] [Audio 2] [Audio 3]

[Option 1]
[Option 2]
[Option 3]
[Option 4]
[Option 5]

How confident are you that this label describes the common feature(s) between these well?

[Completely confident]
[Fairly confident]
[Somewhat confident]

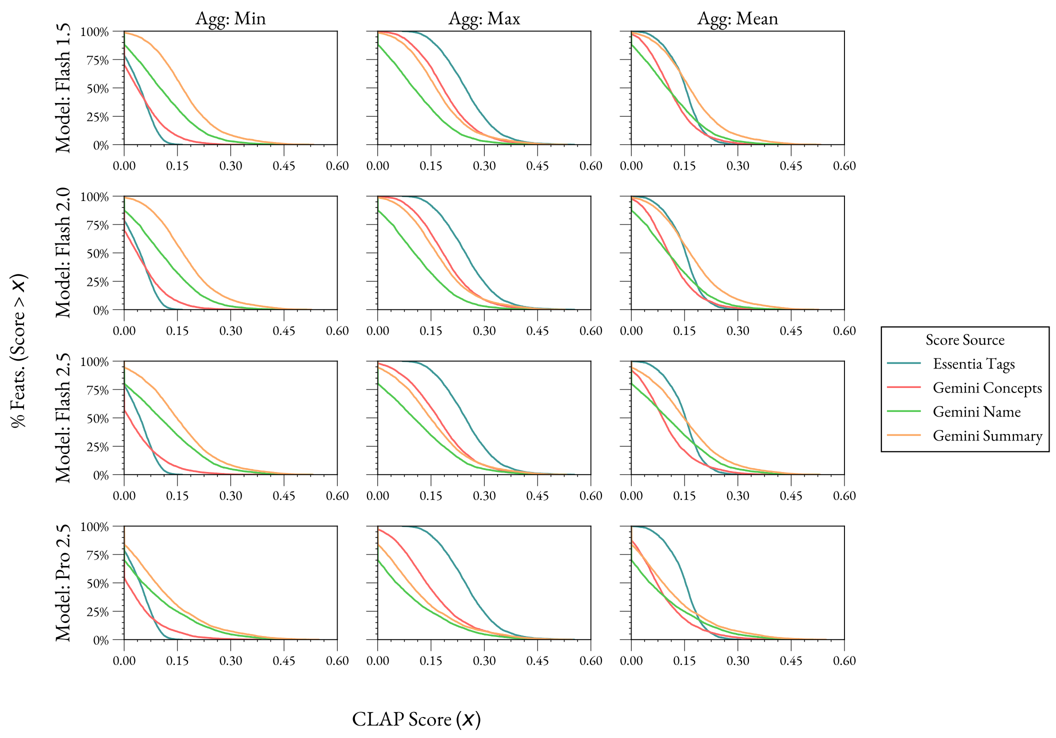

Figure 9: ECDF of maximum CLAP scores across all features for each Gemini variant. Flash 1.5 yields a higher proportion of strong alignments than newer models.

[Slightly confident]
[Not confident at all]

Write down your own label for what these sounds have in common musically

[FREE TEXT]

G Compute Resources

All model training runs in this paper run on a node with 4x NVIDIA L40s GPUs, as well as some of the experiments (e.g. CLAP score computation). Other experiments use a CPU compute node with 128 Intel Xeon Platinum 8375C CPUs @ 2.90GHz each for parallelization.

H LLM Use Disclosure

We used large language models for minor copy editing, including improving grammar and phrasing. The authors reviewed all changes.

I Connections to Music Theory

This work is conceptually related to several core threads we are aware of in music theory, cognitive musicology, and computational modeling of music structure. Here, we clarify how our approach relates to existing work by seeking to uncover "operational" structures (what we refer to as features or concepts) in generative models.

Statistical foundations of music theory [32, 33] argues that listeners develop an understanding of many core elements of Western music (such as meter, harmony, and voice leading) by internalizing probabilistic regularities encountered in musical corpora. From this perspective, foundational concepts within music theory, rather than being seen solely as elements of a formal axiomatic system, can be understood as principles that often reflect this inductively learned, tacit knowledge of musical likelihoods. Temperley's computational cognitive models are themselves compact and idealized representations which help explain why these music-theoretic constructs are perceptually and musically relevant.

Our approach parallels this in an AI setting. Rather than analyzing human musical corpora directly to infer cognitive principles, we seek to "reverse-engineer" generative models *trained* on such data to recover the internal distinctions and abstractions they have formed. The concepts discovered in our pipeline reflect these model abstractions, many of which correspond to statistical regularities observed in music (e.g. genre, texture, instrument timbre, harmonic style).

Operational concepts vs. formal theory Gjerdingen's schema theory [34, 35] discusses the role of learned patterns in shaping musical fluency. His analysis of Galant phrase structure proposes that composers relied on a rich set of prototypical configurations (or *stock musical phrases* in particular), acquired through exposure and practice rather than via formal rule systems. These schemata were typically only formalized retroactively through pedagogical and analytical discourse, as we imagine concepts of the type our pipeline discovers might be in the future. As such, our work focuses on *proto-theoretical regularities*: we extract features that appear to be functionally meaningful to the model regardless of whether they map cleanly to any established theoretical category. Some features do align with known "schemata" but others do not, yet still display internal coherence. In this sense, our pipeline seeks to unearth raw material from which a theory (either of a model's music understanding, or of some aspect of music itself) could potentially be formed.

Meta-theoretical implications This empirical approach may also serve as a testing ground for meta-theoretical questions about music theory. If large-scale generative models learn internal representations that resemble certain theoretical concepts but diverge from others, this may point to systematic gaps or biases in how theory abstracts from musical practice. Conversely, the emergence of coherent but non-canonical features suggests that models may encode distinctions that are musically significant but neglected in prior analyses. In this way, our work does not propose a theory of music, but can be seen as contributing a method for investigating the gap between musical practice and theoretical formalism. We propose that such discovered features features be treated as *hypotheses* about musical structure that may be refined, validated, or rejected through future research.

J Gemini Details

Our prompt to automatically label concepts using Gemini's API, developed through iterative testing:

Prompt for Gemini's Labeling

Listen very carefully to this set of audio clips, which consists of song snippets concatenated in random order.

You need to discover common musical patterns across the whole set, to identify what musical feature is shared across all clips. You will need to listen carefully. For each potential concept you identify, output a name, a confidence score between 0 and 1 (where 1 is highest confidence), and a concise description of the concept.

At a higher level, describe the overall concept shared across the set, give it a suitable name, and provide an overall confidence score (0 to 1).

*Describe the *underlying concepts* not the specific audio snippets (e.g. your description could say "the concept" but not "the audio snippets"). However, try to avoid such verbiage altogether and concisely describe the musical concept's main attributes.*

Include NO FILLER text.

Focus on being specific. Concepts could relate to genre (e.g., hip-hop, salsa, reggaeton, balkan), instruments (e.g., piano, cello, guitar, flute), recording/production techniques (e.g., reverberation, drones, noise, DJ scratching, beatboxing, drum machine, hi-hat patterns, fingerpicking, live recording artifacts, low-pass filtering), or more nuanced musical ideas (e.g., drum solo, chill dance rhythm, serene woodwinds arrangement). These are illustrative examples, NOT a fixed list to choose from.

We use the following model definition to constrain Gemini's response via structured output generation:

Gemini's Response Schema

```python
from pydantic import BaseModel, Field

class Concept(BaseModel):
    """Represents a single identified musical concept."""
    name: str = Field(..., description="Concise name for the musical concept.")
    confidence: float = Field(..., ge=0.0, le=1.0, description="Confidence
    ↪  score (0.0 to 1.0).")
    description: str = Field(..., description="Brief description of the
    ↪  concept.")

class ConceptLabels(BaseModel):
    """Represents the overall analysis result for a set of audio clips."""
    concepts: list[Concept] = Field(..., description="List of specific concepts
    ↪  identified.")
    overall_summary: str = Field(..., description="Overall description of the
    ↪  shared musical concept [no full sentences, concise summary of
    ↪  underlying concept, ignore snippets].")
    overall_name: str = Field(..., description="Concise name for the overall
    ↪  shared concept.")
    overall_confidence: float = Field(..., ge=0.0, le=1.0, description="Overall
    ↪  confidence score (0.0 to 1.0).")
```