LEARNING AUDIO EMBEDDINGS VIA LYRICS ALIGNMENT FOR SCALABLE VERSION IDENTIFICATION 2025

Anonymous Authors
Anonymous Affiliations
anonymous@ismir.net

ABSTRACT

Version Identification aims to recognize distinct renditions of the same underlying musical work, a task central to catalog management, copyright enforcement, and recommendation. While state-of-the-art systems rely on complex audio pipelines, they remain computationally costly, opaque, and difficult to scale. We propose a lightweight and interpretable alternative: *Lyrics-Informed Embeddings* (LIE), audio representations trained to align directly with a lyric-derived semantic space. Our framework leverages advances in automatic speech recognition (Whisper) and multilingual sentence encoders to construct a robust target embedding space from transcribed lyrics. An audio encoder is then trained to project raw audio into this space, optimizing both instance-level alignment and structural consistency. LIE achieves retrieval accuracy on par with, or exceeding, transcription-based and state-of-the-art audio systems, while cutting inference latency by more than $3\times$ relative to transcription pipelines. Our musically grounded framework is lightweight, reproducible, and yields interpretable embeddings that extend beyond version identification to broader music retrieval tasks.

1. INTRODUCTION

Version Identification—also known as Cover Detection—aims to recognize distinct renditions or performances of the same underlying musical composition [1]. Robust VI systems are critical for catalog management, copyright enforcement, cross-platform track linking, and music recommendation.

Historically, VI research has been dominated by audio-based approaches that seek invariance to variations in tempo, instrumentation, pitch or structure. Early systems relied on hand-crafted musical descriptors [2–5], while modern methods employ multimodal deep learning architectures operating on generic audio representations such as the Constant-Q Transform [6–15]. Although effective, these models require extensive training time and computational resources, limiting scalability and reproducibility. Moreover, the learned embedding spaces are often opaque, making it difficult to interpret or transfer them to other music retrieval tasks [16].

We propose a paradigm shift: replacing complex, generic audio pipelines with a lightweight, lyrics-centered framework grounded in a musically meaningful and semantically stable signal. Lyrics are typically preserved across renditions, offering a robust anchor for identifying versions and enabling the learning of interpretable, transferable representations. By operating directly in a lyrics-informed embedding space, we reduce model complexity, improve scalability, and enhance interpretability.

We build on recent advances in ASR and text embedding models to unlock new opportunities for lyrics-based version identification. First, we benchmark pre-trained sentence encoders on clean lyrics to establish an upper bound on retrieval performance. We then incorporate Whisper to transcribe lyrics directly from raw audio and evaluate the robustness of these encoders to transcription noise. Finally, to eliminate the dependency on ASR at inference time, we introduce Lyrics-Informed Embeddings (LIE)—a lightweight audio encoder trained to project raw audio into a lyrics-derived embedding space previously validated for version identification.

2. FRAMEWORK

We introduce an audio representation learning framework for producing Lyrics-Informed Embeddings (LIE)—representations trained to capture lyrics information while eliminating the need for automatic speech recognition at inference. The approach follows a two-stage process. First, a fixed target space is constructed from embeddings of transcribed lyrics. Second, an audio encoder is trained to map raw audio into this space, optimizing its outputs to match the corresponding lyrics embeddings.

2.1 Training Objective

We define the objective guiding the audio encoder to produce embeddings aligned with their corresponding textual representations under cosine similarity. Given a training pair $(x_i, t_i) \in \mathcal{X} \times R^d$, where x_i denotes a 30-second raw audio segment and t_i its target textual embedding, the goal is to learn an audio encoder f_θ that maps x_i to $a_i = f_\theta(x_i) \in R^d$ such that $\cos(a_i, t_i)$ is maximized.

Unlike standard multimodal representation learning—which jointly optimizes audio and text encoders to learn a joint embedding space [17–19]—our approach

projects audio directly into an existing, semantically structured textual space. Since each audio sample is paired with an explicit target embedding, traditional contrastive objectives can introduce noise into the learning signal. More importantly, this setup allows the learning process to exploit the geometric structure of the target space, rather than relying solely on in-batch negatives.

To this end, we adopt a criterion combining instance-level alignment with structure preservation:

$$\mathcal{L}_{\text{total}} = \alpha\,\mathcal{L}_{\cos} + (1-\alpha)\,\mathcal{L}_{\text{struct}},$$

$$\mathcal{L}_{\cos} = \frac{1}{B}\sum_{i=1}^{B}\big(1-\cos(a_i,t_i)\big)$$

$$\mathcal{L}_{\text{struct}} = \frac{1}{B^2}\sum_{i,j=1}^{B}\big(\cos(a_i,a_j)-\cos(t_i,t_j)\big)^2,$$

where B is the batch size and $\alpha \in [0,1]$ an hyperparameter.

The cosine term $\mathcal{L}_{\cos}$ provides an explicit instance-level alignment signal, pulling each audio embedding a_i toward its textual counterpart t_i and directly optimizing the retrieval metric. In contrast, the geometry-preserving term $\mathcal{L}_{\text{struct}}$ enforces global structural consistency by encouraging pairwise similarities in the learned audio space to match those in the fixed textual space, thereby transferring its semantic structure to the learned space.

2.2 Target Space Creation

A key challenge in leveraging lyrics is the limited accessibility of clean, time-aligned transcriptions for large music collections. Such resources are often unavailable at scale and typically require third-party licensing. To overcome this limitation, we construct the target embedding space—serving as the supervision signal for training the audio encoder—directly from audio, thus removing reliance on external lyric datasets. This is achieved through a two-stage pipeline: (1) transcribing vocal segments using a pre-trained automatic speech recognition (ASR) model, and (2) encoding the resulting text into dense vectors with a multilingual sentence encoder already fine-tuned for semantic textual similarity.

As the framework is inherently lyrics-centered, its scope is restricted to non-instrumental tracks. In addition, because ASR models are trained primarily on speech, they tend to hallucinate in non-vocal sections, generating spurious outputs despite the absence of linguistic content. A dedicated preprocessing stage is therefore introduced to filter out tracks with insufficient lyrical content and to extract vocal-only segments prior to transcription, thereby improving both transcription accuracy and the representativeness of the resulting embeddings.

Data Preprocessing. We employ a proprietary deep learning model to estimate a vocalness probability v for each non-overlapping 3-second audio window. The global vocalness score of a track is computed as the mean v over all windows, and recordings with a score below 0.5 are excluded to ensure sufficient lyrical content. Windows with $v \geq 0.5$ are retained as vocal segments and concatenated into contiguous regions, which are then truncated or zero-padded to a fixed length of 30 seconds, meeting the input requirements of the downstream ASR model.

Transcription. We adopt Whisper [20], a multilingual encoder–decoder Transformer widely regarded as state-of-the-art in ASR, noted for its robustness to noise and other real-world acoustic variability—making it a natural choice for the heterogeneous and noisy conditions of music audio. Specifically, we use the `whisper-large-v3-turbo` model, a variant which offers strong transcription accuracy with fast inference.

Text Encoding. We use `Alibaba-NLP/gte-multilingual-base` [21], an encoder-only Transformer fine-tuned for semantic similarity, supporting over 70 languages and producing 768-dimensional sentence embeddings. The model, selected after benchmarking several multilingual text encoders [22–24], consistently outperformed alternatives on both clean and ASR-generated lyrics in the version identification task, and showed stronger robustness to noise introduced by transcription.

2.3 Audio Encoder Architecture

We present the architecture of the proposed Lyrics-Informed Embeddings (LIE) model, which consists of three main components: (1) a frozen Whisper encoder serving as a high-level feature extractor, (2) a learnable attention-based temporal pooling mechanism to aggregate frame-level features, and (3) a multi-layer perceptron (MLP) projection head mapping aggregated audio features into the textual embedding space. An overview is shown in Figure 1. Training configuration and inference time estimation results are provided in Appendix E and C.

Audio Encoder. We adopt the encoder of `openai/whisper-large-v3-turbo`—also used in the target space construction—as our audio backbone. Given an 80-channel log-Mel spectrogram computed by Whisper's feature extractor, the encoder produces a sequence of hidden states $H \in R^{L\times d_w}$, where $d_w = 1280$ is the embedding dimension and $L = 1500$ corresponds to the number of frames for a 30-second input.

Our choice of Whisper is motivated by two key considerations. First, its internal representations are expected to capture phonetic and linguistic information as a consequence of its ASR training objective, which makes them suitable for alignment with textual embeddings. Second, the same Whisper model is used in constructing the target space through transcription, creating an inherent structural compatibility between the audio and text modalities. As a consequence, we keep the encoder frozen throughout training to preserve this alignment and maintain the structure of the latent space learned during Whisper's large-scale training.

Attention-based Temporal Pooling. The Whisper encoder produces a sequence of frame-level hidden states $H = [h_1,\ldots,h_L] \in R^{L\times d_w}$, which must be aggregated into a single fixed-dimensional vector to enable projection into the target text embedding space. Rather than relying on mean pooling—which treats all frames equally—we

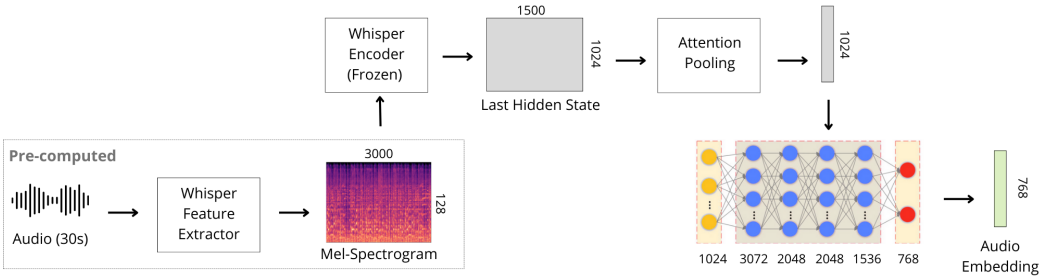

Figure 1. Overview of the LIE framework.

employ a learnable attention-based pooling layer inspired by [25]. A formal description is provided in Appendix A.

Projection Head. The pooled vector $\tilde{h}$ is projected into the 768-dimensional target lyric embedding space through a four-layer MLP with hidden sizes [3072, 2048, 2048, 1536]. Each intermediate layer is followed by LayerNorm and a ReLU activation, while the final layer outputs the lyrics-informed embedding used in our downstream task. This projection head serves as the bridge between modalities, learning the transformation that aligns audio features with their textual counterparts.

3. EMPIRICAL STUDIES

Datasets. We base our experiments on the recently introduced Discogs-VI dataset [26]. For this work, we match Discogs-VI entries to a proprietary catalog to retrieve corresponding `.mp3` audio files using metadata matching, followed by filtering and validation to ensure high-confidence matches. This process yields a subset of 679,692 tracks with associated audio. To train our model, we construct paired data (z, t) consisting of 30-second log-Mel spectrograms precomputed offline using Whisper's feature extractor, and lyric embeddings obtained via the target space construction pipeline (see Section 2.2). We retain only unpadded 30-second segments to accelerate training, resulting in 1.67M audio–text pairs. The dataset is split 80/10/10 into training, validation, and test sets, yielding 1.34M pairs for training.

Evaluation. We evaluate on three benchmarks differing in scale, language coverage, and difficulty. While Covers80 [27] and SHS100K-TEST [28] remain standard benchmarks in the field, Discogs-VI corresponds to the matched subset described in Section 3 restricted to entries with an available YouTube link. All datasets follow the same preprocessing pipeline presented in Appendix B. This results in 116, 167, and 4,623 tracks for Covers80, SHS100K, and Discogs-VI, respectively. For larger-scale evaluation, we additionally construct XL variants of SHS100K and Discogs-VI by removing the clean-lyrics constraint (see Appendix B for reference), resulting in datasets of 1,086 and 121,729 tracks.

In the retrieval setup, embeddings are precomputed for all catalog items $\mathcal{C}$. Given a query $q \in \mathcal{C}$, each candidate $x \in \mathcal{C} \setminus q$ is scored by cosine similarity $\cos(e_q, e_x)$, and results are ranked in descending order. We report the hit rate at rank 1 (HR@1) and the mean average precision at rank 10 (MAP@10).

3.1 Clean and Transcribed Lyrics Benchmarking

We benchmarked a range of multilingual sentence text encoders to evaluate their capacity to transfer their knowledge to the version identification task in a zero-shot setting. For conciseness, we report only the results of the model ultimately selected to define the target embedding space, `gte-multilingual-base`.

Table 1. Text embedding model performance on clean and transcribed lyrics

Dataset	Metric	Clean lyrics	Transcription
Covers80	HR@1	1.000	0.975
	MAP@10	1.000	0.979
SHS100K	HR@1	0.917	0.909
	MAP@10	0.863	0.852
Discogs-VI	HR@1	0.934	0.929
	MAP@10	0.913	0.893

Clean lyrics provide a strong upper bound, with `gte-multilingual-base` achieving perfect accuracy on Covers80 and strong results on SHS100K and Discogs-VI. Equally noteworthy is the robustness to transcription noise. Substituting clean text with Whisper-generated transcriptions leads to only limited degradation in performance, indicating that transcription errors, while unavoidable, do not substantially compromise the discriminative capacity of the textual embeddings.

These findings confirm that (i) lyrics are a highly effective modality for version identification; and (ii) that pretrained multilingual encoders, combined with Whisper-based ASR, provide a robust solution capable of scaling to diverse datasets without task-specific fine-tuning.

3.2 Audio-to–Text Alignment

We assess the alignment between learned audio embeddings (LIE) and their corresponding textual embeddings at both the segment and track levels. First, we compute the cosine similarity for each of the 167,484 audio–text pairs from the test set, where the audio embedding corresponds to the model output for a 30-second segment and the text embedding to its paired target. Second, for tracks containing at least two such segments, we form a global

audio representation by averaging their segment-level embeddings and compare it to the track-level text embedding derived from the full transcription.

Segment-level embeddings yield a mean similarity of 0.8574 (std: 0.0757), whereas aggregated track-level embeddings reach 0.9109 (std: 0.0379). The higher mean and lower variance at the track level show that the model captures segment-level signals and integrates them into stable, global representations. These results provide compelling evidence that our approach achieves tight audio–text alignment, offering cross-modal correspondence through a compact and efficient architecture.

3.3 Downstream Version Identification

Comparison with Transcription Baselines. We compare LIE to transcription-based baselines used to derive target textual embeddings. LIE track-level representations are obtained by averaging embeddings from 30-second vocal segments. For the transcription baseline, we evaluate two settings: (1) global embeddings from full transcriptions, approximating an upper bound and aligning with Section 3.1; (2) averaged embeddings from transcriptions of the same 30-second vocal segments used as LIE inputs, providing a direct basis for comparison.

Table 2. Comparison of LIE and transcription baselines

Dataset	Metric	Transc.	Transc.	LIE
Covers80	HR@1	**0.975**	0.937	0.949
	MAP@10	**0.979**	0.945	0.966
SHS100k-XL	HR@1	**0.954**	0.925	0.935
	MAP@10	**0.910**	0.870	0.875
Discogs-VI-XL	HR@1	**0.856**	0.843	0.853
	MAP@10	0.832	0.817	**0.923**

Across all benchmarks, LIE consistently outperforms transcription-based embeddings extracted from averaged 30-second segments, both in terms of HR@1 and MAP@10. Moreover, for all three datasets, results obtained with LIE embeddings closely approach the upper bound defined by the full-transcription pipeline, with LIE even surpassing it on MAP@10 for Discogs-VI-XL. On this dataset, however, LIE performance drops compared to smaller benchmarks, but this drop is not unique to LIE: the transcription-based model with full lyrics also sees reduced accuracy. Although the scale and diversity of the benchmark likely contribute to these challenges, manual analysis of 200 errors on full transcriptions indicated that roughly 60% arose from data inconsistencies in Discogs-VI.

Comparison with Audio Baselines. To assess the broader effectiveness of LIE, we compare it against several state-of-the-art audio-only systems: ByteCover2 [9], CLEWS [13], CQTNet [14], and DViNet [15], using the official implementations and pretrained checkpoints released by the authors of CLEWS [13]. Full results are reported in Appendix D, while only top-2 audio baselines are presented below.

Table 3. Comparison of LIE with audio baselines

Dataset	Metric	ByteCover2	CLEWS	LIE
Covers80	HR@1	0.865	0.835	**0.949**
	MAP@10	0.877	0.880	**0.966**
SHS100k-XL	HR@1	**0.953**	0.931	0.935
	MAP@10	**0.884**	0.847	0.875
Discogs-VI-XL	HR@1	0.843	0.816	**0.853**
	MAP@10	0.812	0.790	**0.823**

LIE delivers competitive performance across all benchmarks, achieving particularly strong results on Covers80 and Discogs-VI-XL, where it outperforms all audio baselines in HR@1 and MAP@10. On SHS100k-XL, ByteCover2 achieves the top results, with LIE ranking closely behind, but the difference is marginal and partially explained by dataset characteristics: SHS100k contains a notable fraction of parodies—covers that retain the same melody but feature ironic or entirely different lyrics, which are not addressable by lyrics-centered models like LIE.

Overall, these results underscore that LIE generalizes effectively to the version identification task, achieving performance comparable to state-of-the-art audio-based models without relying on textual input or task-specific fine-tuning. Unlike ByteCover2, which benefits from large-scale training and complex architectures, LIE relies on a lightweight architecture, making the model more efficient and easy to reproduce. Yet, its competitiveness across all datasets highlights the effectiveness of a lyrics-informed approach in producing robust and generalizable audio embeddings, opening a promising new direction for version identification.

4. CONCLUSION

This work has introduced Lyrics-Informed Embeddings (LIE), a lightweight and reproducible framework for version identification that departs from the prevailing trend towards increasingly complex, multimodal, and resource-intensive architectures. By grounding audio representations in a lyric-derived semantic space, LIE leverages musically meaningful supervision while remaining lightweight and easy to deploy.

The approach, however, is subject to limitations. The reliance on a vocal detector introduces an additional computational step and constrains the method to tracks with sufficient lyrical content. In addition, the use of a general-purpose text encoder not explicitly optimized for version identification highlights opportunities for refinement. Future research will address these directions by fine-tuning the text encoder to enhance discriminability in the target space, by improving the efficiency of vocal detection, and by integrating LIE within multimodal systems to handle instrumental tracks. Taken together, this contribution positions lyrics-informed representation learning as a promising direction for version identification, offering an interpretable, generalizable, and scalable alternative to complexity-heavy systems in music information retrieval.

5. REFERENCES

[1] F. Yesiler, G. Doras, R. M. Bittner, C. J. Tralie, and J. Serra, "Audio-based musical version identification: Elements and challenges," *IEEE Signal Processing Magazine*, vol. 38, no. 6, p. 115–136, Nov. 2021. [Online]. Available: http://dx.doi.org/10.1109/MSP.2021.3105941

[2] G. Doras and G. Peeters, "Cover detection using dominant melody embeddings," 2019. [Online]. Available: https://arxiv.org/abs/1907.01824

[3] F. Yesiler, J. Serrà, and E. Gómez, "Accurate and scalable version identification using musically-motivated embeddings," 2020. [Online]. Available: https://arxiv.org/abs/1910.12551

[4] G. Doras, F. Yesiler, J. Serrà, E. Gómez, and G. Peeters, "Combining musical features for cover detection," in *International Society for Music Information Retrieval Conference*, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:236096628

[5] F. Yesiler, J. Serrà, and E. Gómez, "Less is more: Faster and better music version identification with embedding distillation," 2020. [Online]. Available: https://arxiv.org/abs/2010.03284

[6] Z. Yu, X. Xu, X. Chen, and D. Yang, "Temporal pyramid pooling convolutional neural network for cover song identification," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, ser. IJCAI'19. AAAI Press, 2019, p. 4846–4852.

[7] X. Du, Z. Yu, B. Zhu, X. Chen, and Z. Ma, "Bytecover: Cover song identification via multi-loss training," 2021. [Online]. Available: https://arxiv.org/abs/2010.14022

[8] S. Hu, B. Zhang, J. Lu, Y. Jiang, W. Wang, L. Kong, W. Zhao, and T. Jiang, "Wideresnet with joint representation learning and data augmentation for cover song identification," in *Interspeech 2022*, 2022, pp. 4187–4191.

[9] X. Du, K. Chen, Z. Wang, B. Zhu, and Z. Ma, "Bytecover2: Towards dimensionality reduction of latent embedding for efficient cover song identification," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 616–620.

[10] X. Du, Z. Wang, X. Liang, H. Liang, B. Zhu, and Z. Ma, "Bytecover3: Accurate cover song identification on short queries," 2023. [Online]. Available: https://arxiv.org/abs/2303.11692

[11] F. Liu, D. Tuo, Y. Xu, and X. Han, "Coverhunter: Cover song identification with refined attention and alignments," in *2023 IEEE International Conference on Multimedia and Expo (ICME)*, 2023, pp. 1080–1085.

[12] X. Du, "X-cover: Better music version identification system by integrating pretrained asr model," in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR 2024)*. San Francisco, California, USA and Online: International Society for Music Information Retrieval Conference (ISMIR), 2024, pp. 70–77.

[13] J. Serrà, R. O. Araz, D. Bogdanov, and Y. Mitsufuji, "Supervised contrastive learning from weakly-labeled audio segments for musical version matching," 2025. [Online]. Available: https://arxiv.org/abs/2502.16936

[14] Z. Yu, X. Xu, X. Chen, and D. Yang, "Learning a representation for cover song identification using convolutional neural network," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 541–545.

[15] R. O. Araz, J. Serrà, X. Serra, Y. Mitsufuji, and D. Bogdanov, "Discogs-vinet-mirex," in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, 2024, technical report submitted as MIREX 2024 entry.

[16] M. Abrassart and G. Doras, "And what if two musical versions don't share melody, harmony, rhythm, or lyrics ?" 2022. [Online]. Available: https://arxiv.org/abs/2210.01256

[17] Q. Huang, A. Jansen, J. Lee, R. Ganti, J. Y. Li, and D. P. W. Ellis, "Mulan: A joint embedding of music audio and natural language," 2022. [Online]. Available: https://arxiv.org/abs/2208.12415

[18] B. Elizalde, S. Deshmukh, M. A. Ismail, and H. Wang, "Clap: Learning audio concepts from natural language supervision," 2022. [Online]. Available: https://arxiv.org/abs/2206.04769

[19] Y. Wu, K. Chen, T. Zhang, Y. Hui, M. Nezhurina, T. Berg-Kirkpatrick, and S. Dubnov, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," 2024. [Online]. Available: https://arxiv.org/abs/2211.06687

[20] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022. [Online]. Available: https://arxiv.org/abs/2212.04356

[21] X. Zhang, Y. Zhang, D. Long, W. Xie, Z. Dai, J. Tang, H. Lin, B. Yang, P. Xie, F. Huang, M. Zhang, W. Li, and M. Zhang, "mgte: Generalized long-context text representation and reranking models for multilingual text retrieval," 2024. [Online]. Available: https://arxiv.org/abs/2407.19669

[22] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei, "Multilingual e5 text embeddings: A technical report," 2024. [Online]. Available: https://arxiv.org/abs/2402.05672

[23] M. Günther, J. Ong, I. Mohr, A. Abdessalem, T. Abel, M. K. Akram, S. Guzman, G. Mastrapas, S. Sturua, B. Wang, M. Werk, N. Wang, and H. Xiao, "Jina embeddings 2: 8192-token general-purpose text embeddings for long documents," 2024. [Online]. Available: https://arxiv.org/abs/2310.19923

[24] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[25] H. Touvron, M. Cord, A. El-Nouby, P. Bojanowski, A. Joulin, G. Synnaeve, J. Verbeek, and H. J'egou, "Augmenting convolutional networks with attention-based aggregation," *arXiv preprint arXiv:2112.13692*, 2021.

[26] R. O. Araz, X. Serra, and D. Bogdanov, "Discogs-vi: A musical version identification dataset based on public editorial metadata," 2024. [Online]. Available: https://arxiv.org/abs/2410.17400

[27] D. P. W. Ellis, "The "covers80" cover song data set," 2007, accessed: 2025-03-27. [Online]. Available: http://labrosa.ee.columbia.edu/projects/coversongs/covers80

[28] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset." in *Ismir*, vol. 2, no. 9, 2011, p. 10.

A. ATTENTION MECHANISM DETAILS

A learnable `[CLS]` token $q_{\text{cls}} \in R^{d_w}$ is prepended to H and acts as the sole query in a single-head self-attention mechanism over the sequence. Rotary positional embeddings (RoPE) are applied to both queries and keys to encode relative position information.

$$Q = q_{\text{cls}} W_Q,\ K = HW_K,\ V = HW_V,$$

$$A = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) \in R^{1\times L}, \quad \tilde{h} = AV \in R^{1\times d_v}.$$

Here, $W_Q, W_K, W_V \in R^{d_w \times d_k}$ are learnable projection matrices. In this formulation, the attention weights A quantify the relevance of each frame-level hidden state in H with respect to the prepended `[CLS]` token. These weights are then used to compute an attention-weighted mean over the value projections V, yielding the updated `[CLS]` representation $\tilde{h}$. The aggregated representation $\tilde{h}$ is then passed through a residual feed-forward block with LayerNorm to produce the final pooled embedding.

B. EVALUATION BENCHMARKS PREPROCESSING

The three evaluated benchmarks are standardized via the following preprocessing pipeline: (i) match tracks to a proprietary catalog via fingerprinting after downloading YouTube audio with provided links, (ii) retain only tracks with clean and valid lyrics in the catalog, with lyric validity checked by comparing them to ASR transcriptions (both encoded with *gte-multilingual-base*, discarding pairs with cosine similarity < 0.6), and (iii) remove tracks with insufficient vocal content (global vocalness score ≤ 0.5, as defined in Section 2.2).

C. TRAINING CONFIGURATION

Model training is performed for 3 epochs with a batch size of 128 on a single NVIDIA RTX A5000 GPU (24 GB VRAM, CUDA 12.2), requiring approximately 33 hours. We adopt the AdamW optimizer, with weight decay set to 0.01 and β coefficients to $(0.9, 0.98)$. The learning rate is fixed at 1×10^{-4} with a linear warmup over the first $10,000$ steps to stabilize early training. Mixed precision (AMP) and torch.compile are enabled for the Whisper encoder to accelerate computation and reduce memory usage, while the rest of the model is trained in standard precision.

D. COMPARISON OF LIE WITH AUDIO BASELINES: ADDITIONAL RESULTS

Table 4. Comparison of LIE with audio baselines

Dataset	**Metric**	CQTNet	DViNet	LIE
Covers80	HR@1	0.848	0.861	0.949
	MAP@10	0.856	0.886	0.966
SHS100k-XL	HR@1	0.900	0.931	0.935
	MAP@10	0.789	0.859	0.875
Discogs-VI-XL	HR@1	0.641	0.751	0.853
	MAP@10	0.568	0.719	0.823

E. INFERENCE TIME

One of the primary motivations behind LIE was to remove the inference bottleneck imposed by the transcription stage in the baseline pipeline used to generate target textual embeddings. To quantify the efficiency gains of LIE, we measured inference time for both pipelines on $2{,}000$ randomly selected tracks from the Discogs-VI benchmark.

In the baseline, Whisper-based vocal transcription alone accounts for 4.398s (std: 3.243) of the 6.072s (std: 3.416) total average inference time, making large-scale deployment impractical. LIE, on the other hand, achieves an average inference time of 1.895s (std: 0.513) per track, corresponding to a $3.2\times$ speed-up over the baseline. When excluding preprocessing—1.661s (std: 0.443) for the baseline and 1.673s (std: 0.415) for LIE—the advantage becomes even more pronounced: the LIE forward pass requires only 0.221s (std: 0.117), compared to 4.410s (std: 1.220) for the baseline's Whisper-dependent stage.