# X-COVER: BETTER MUSIC VERSION IDENTIFICATION SYSTEM BY INTEGRATING PRETRAINED ASR MODEL

xingjian.du97@gmail.com

### ABSTRACT

Methods based on deep learning have emerged as a dominant approach for cover song identification (CSI) literature over the past years, among which ByteCover systems have consistently delivered state-of-the-art performance across major CSI datasets in the field. Despite its steady improvements along previous generations from audio feature dimensionality reduction to short query identification, the system is found to be vulnerable to audios with noise and ambiguous melody when extracting musical information from constant-Q transformation (CQT) spectrograms. Although some recent studies suggest that incorporating lyric-related features can enhance the overall performance of CSI systems, this approach typically requires training a separate automatic lyric recognition (ALR) model to extract lyric-related features from music recordings. In this work, we introduce X-Cover, the latest CSI system that incorporates a pre-trained automatic speech recognition (ASR) module, Whisper, to extract and integrate lyricsrelated features into modelling. Specifically, we jointly fine-tune the ASR block and the previous ByteCover3 system in a parameter-efficient fashion, which largely reduces the cost of using lyric information compared to training a new ALR model from scratch. In addition, a bag of tricks is further applied to the training of this new generation, assisting X-Cover to achieve strong performance across various datasets.

# 1. INTRODUCTION

In the rapidly evolving field of music information retrieval, Cover Song Identification (CSI), which aims to identify different versions of a specific musical composition within a large database, remains a complex and computationally challenging task [1,2]. This problem has received considerable interest for its wide-ranging applications such as intellectual property management and enhancing music recommendation systems [3,4].

With the advancements in deep learning, CSI systems based on neural networks gradually replace traditional models based on handcrafted features [5,6] and become a new paradigm for real-world deployment. Existing methods typically frame CSI as either a classification problem [7–9], a metric learning problem [10], or a combination of both [11,12]. On the other hand, the proliferation of social video platforms like TikTok has also led to a surge in short-form videos, which often contain remixed or covered segments of original compositions and hence involve copyright infringement issues. Unfortunately, as most of the existing works above contain a global pooling layer to directly aggregate the information from all time sections, they are found to suffer from identifying these secondslong short segments.

To address this problem, the latest ByteCover3 system [13] first splits each full audio track into a set of short spectrogram chunks and then uses a neural-network-based extractor (i.e. the ResNet-IBN module introduced in the first ByteCover generation [11]) to encode them into latent embeddings. These low-dimensional embeddings are later sent to calculate a Local Alignment Loss (LAL) that uses the matching of local embeddings to identify short queries against full songs.

Despite the progress in detecting short cover songs, the ByteCover3 system is still found to be vulnerable to nonmusical information in real-world scenarios. For instance, musical segments in short videos are frequently overlaid with ambient noise, speech, or poorly composed usergenerated melodies. The presence of these non-musical elements that mask or distort the musical elements can mislead the feature extraction phase of ByteCover3, leading to inaccurate or ambiguous representations of the audio content. This misrepresentation can degrade the system's ability to correctly match the audio sample against its database of known songs, reducing both the accuracy and reliability of the system in operational settings.

To enhance the robustness of existing CSI systems, an intuitive approach is to incorporate more discriminative information into model training. Recent research [14,15] has demonstrated that lyric-related features are less susceptible to being masked by unrelated noise sources and can serve

<sup>©</sup> Y. Du, M. Liu, P. Zou, X. Liang, Z. Wang, H. Liang, and B. Zhu. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). Attribution: X. Du, M. Liu, P. Zou, X. Liang, Z. Wang, H. Liang, and B. Zhu, "X-Cover: Better Music Version Identification system by integrating pretrained ASR model", in *Proc. of the 25th Int. Society for Music Information Retrieval Conf.*, San Francisco, United States, 2024.

as auxiliary inputs to bolster CSI systems. For example, one recent work [14] utilizes latent embeddings from an automatic lyrics recognition model (ALR) as lyric-related features for CSI. However, this approach requires training an extra ALR module from the beginning and employs Dali [16] as its training dataset, which implies that it only supports the recognition of English lyrics.

In this paper, we extend the current ByteCover3 system by leveraging a pre-trained Automatic Speech Recognition (ASR) model to jointly model both lyric and musical features for short-query identification, which eliminates the need for training an additional ALR model from scratch. Specifically, we use Whisper [17] as the pre-trained ASR module for its inherent scalability and robust recognition capabilities across multiple languages. However, the training is still quite challenging in terms of GPU memory consumption and inference time given the large parameter size of Whisper and its autoregressive nature at the text decoding stage. To counter these issues, we employ a prefixtuning fashion that adapts the output of Whisper to CSI training. A trainable prefix latent is added before each text decoder block in Whisper to reprogram the model to extract features specific to CSI task without extensive retraining of the entire model. Finally, the fusion model is trained using the local alignment loss (LAL) scheme introduced in ByteCover3, and together with a bag of new techniques, we further improve the current ByteCover system to be more efficient and accurate in CSI tasks with controllable training GPU memory and reasonable inference time.

# 2. PRELIMINARIES

This paper builds on ByteCover series [11–13] training framework and model structure. The primary motivation of ByteCover series is to develop a highly accurate, robust and efficient cover song detection system for real-world industrial-level tasks with various query types and large music corpora, beyond typical laboratory settings.

ByteCover1 [11] introduced a streamlined framework designed to train a neural network that extracts versionrelated embeddings from the CQT spectrogram of input audio recordings. This model utilized Instance Batch Normalization (IBN) layer [18] within its ResNet architecture, which enhances the model's capability to learn invariant features while preserving discrimination and is critical for handling diverse musical styles. Additionally, a Generalized Mean (GeM) pooling layer was employed to compress local features into a global feature, optimizing the model's training objectives. Furthermore, ByteCover1 adopted a multi-loss training paradigm that combined classification loss and triplet loss, fostering a more robust representation and improved accuracy.

For improved throughput, the authors of Byte-Cover2 [12] identified an anisotropy in the embedding distribution of ByteCover1, which led to inefficient utilization of the embedding dimension size. To address this issue, ByteCover2 introduced a Principal Component Analysis - Fully Connected (PCA-FC) layer. The weights of this layer are informed by the transformation matrix derived from a PCA analysis of the original ByteCover1 embeddings. This strategic adjustment effectively alleviated the anisotropy problem, enabling ByteCover2 to match the performance of ByteCover1 while only requiring oneeighth of the dimension size. Consequently, this reduction drastically accelerated the retrieval of query embeddings and linearly decreased the storage costs of the embedding database relative to the magnitude of the dimension size reduction.

With the emergence of short video content, more queries in CSI systems appear to be short audio clips. However, the authors of ByteCover3 [13] observed that state-of-the-art CSI methods performed suboptimally on these short queries, where the accuracy of previous deep learning CSI models [10-12, 19] degraded significantly as the duration of query audio decreased. This issue was linked to the global pooling modules employed in previous works that often neglected local features, which complicates the task of matching segments of songs to complete tracks. Traditional audio matching algorithms, such as the "shingling" method referenced in [20], slice inputs into segments and extract features separately to preserve local details. Unfortunately, this straightforward strategy struggled when applied to deep learning-based CSI methods, which are inherently data-driven rather than handcrafted. Consequently, there is a misalignment between training and inference objectives: while softmax and triplet losses focus on matching pairs of embedding vectors, the model deals with sequences of local embeddings at the inference phase.

To bridge this gap, ByteCover3 introduced a novel training paradigm known as Local-Aware Losses (LAL) for metric learning on sequence data. This approach extends softmax and triplet losses into a more general form that directly optimizes the metrics between two sequences of embeddings. Specifically, ByteCover3 employs the MaxMean Measure to assess the similarity between two sequences of vectors, consequently ensuring better alignment of the training and inference targets and enhancing performance. This measure calculates the similarity between each segment in the query audio and the most similar segment in the candidate song, averaging these similarities to produce a final similarity score. This method is computationally efficient and differentiable. With the use of LAL and the MaxMean measure, ByteCover3 demonstrated significant improvements in retrieval capabilities with 30-second queries.

The subsequent method section, Section 3, describes the details of X-Cover and is organized as follows:

- Subsection 3.1 introduces the overall framework of X-Cover, which retains a high-level similarity to ByteCover3.
- Subsection 3.2 discusses the incorporation of the pretrained ASR model, Whisper [17], to enhance the robustness of the CSI system. It also describes a non-autoregressive decoding method designed to accelerate Whisper's decoding. Compared to previous methods that relied solely on cover song data,



Figure 1. The input CQT spectrogram and Mel spectrogram are split into n equal-sized chunks with overlap in temporal dimension. Subsequently, the ResNet-IBN extractor derived from [13] generates an embedding  $\mathbf{X}_m$  that contains N local features corresponding to N input chunks with the CQT spectrogram. The optimized Whisper variant generates an embedding  $\mathbf{X}_l$  that contains N local features corresponding to N input chunks with the CQT spectrogram. The optimized Whisper variant generates an embedding  $\mathbf{X}_l$  that contains N local features corresponding to N input chunks with the log Mel spectrogram. Then, we optimize the model with a multi-loss objective that consists of a classification loss  $L_{\text{lac}}$  and a triplet loss  $L_{\text{lat}}$  using the *MaxMean* measure, where  $\mathbf{X}_p$  and  $\mathbf{X}_n$  represent the positive sample and negative sample in triplet loss.

X-Cover achieved superior performance without the necessity of pretraining on additional lyric data. This simplification and streamlining of the training process significantly enhance the efficiency of model preparation.

• Subsection 3.3 introduces a bag of tricks to improve the performance of the CSI model, covering aspects such as data augmentation, model architecture, and loss function design details.

### 3. X-Cover

In this section, we delineate the architecture and training paradigms of X-Cover, emphasizing its novelties: efficient adaption of Whisper and a bag of tricks for improving ByteCover3 in Cover Song Identification (CSI) tasks. The overall architecture is depicted in Fig. 1.

### 3.1 Overall Framework of X-Cover

The architecture of X-Cover is an evolutionary extension of ByteCover3, delineated in Section 3. X-Cover retains the multi-objective learning paradigm and the ResNet-Based Feature Extractor from ByteCover3. To incorporate lyric-related features, X-Cover introduces a pretrained ASR model, Whisper, alongside the existing ResNet-IBN architecture [11].

In ByteCover3, local features are extracted by initially resampling the audio to 22,050 Hz and partitioning it into N overlapping segments of 20 seconds each, with a 10second hop. These segments are subsequently transformed into CQT spectrograms, serving as the input to the ResNet-IBN model. The model outputs a 4-D embedding, which undergoes GeM pooling to yield a compact final local embedding  $\mathbf{X}_m \in \mathbb{R}^{N \times C_m}$ , comprising N local features.

To facilitate the integration of Whisper as an additional feature extractor, mel spectrograms are extracted from the input audio. To temporally align the features from both branches, the same chunking strategy as in ByteCover3 is employed. Post chunking, the specialized Whisper model for CSI utilizes these mel spectrograms to extract lyricrelated embeddings  $\mathbf{X}_l \in \mathbb{R}^{N \times C_l}$ . Upon obtaining the two sets of latent embeddings,  $\mathbf{X}_m$  and  $\mathbf{X}_l$ , a straightforward feature-dimensional concatenation is performed to generate the fused embedding  $\mathbf{X}_f \in \mathbb{R}^{N \times (C_m + C_l)}$ , for the simplicity, we redefine it as  $\mathbf{X} \in \mathbb{R}^{N \times C}$  where  $C = C_f = C_m + C_l$ . This is feasible due to the temporal alignment and identical lengths of  $\mathbf{X}_m$  and  $\mathbf{X}_l$ . X-Cover leverages the Local Alignment Loss (LAL) methodology, originally proposed in ByteCover3, to enhance local segments matching capabilities. The LAL comprises a classification loss  $\mathcal{L}_{lac}$  and a triplet loss  $\mathcal{L}_{lat}$ , defined as follows:

$$logit_k = MaxMean(\mathbf{X}, \mathbf{W}_k), \tag{1}$$

$$\mathcal{L}_{\text{lac}} = \text{CE}(\sigma(\{\text{logit}_k\}_{k=1}^{K}), y),$$
(2)

$$\mathcal{L}_{\text{lat}} = [\text{MaxMean}(\mathbf{X}_n, \mathbf{X}) - \text{MaxMean}(\mathbf{X}_p, \mathbf{X})]_+,$$
(3)

where  $CE(\cdot, \cdot)$  is the cross entropy,  $\sigma(\cdot)$  is the softmax function,  $\mathbf{X}_n, \mathbf{X}_p \in \mathbb{R}^{N \times C}$  is the fused embedding of the negative sample and postive sample while calculating triplet loss.  $\mathbf{W} \in \mathbb{R}^{K \times L \times C}$  is a trainable weight matrix in the linear layer before softmax, and  $\mathbf{W}_k \in \mathbb{R}^{L \times C}$  denotes the proxy representation for class k. L is a hyperparameter which is set to 9 in ByteCover3 and X-Cover. These loss functions serve as a robust optimization objective, fortifying the model's performance and adaptability for emerging challenges in music information retrieval. The subsequent subsections will expound upon these enhancements and their contributions to the overarching efficacy of X-Cover.

### 3.2 Efficient Adaption of Whisper for CSI

Whisper [17] serves as a state-of-the-art ASR framework, exhibiting robust scalability through its Transformer-based encoder-decoder architecture. Its efficacy in ASR tasks has been corroborated by numerous studies [21, 22]. The encoder processes normalized spectrograms via an initial stem, consisting of two convolutional layers, before routing the output through multiple Transformer blocks employing pre-activation residual connections.

On the decoder end, learned positional embeddings are integrated with tied input-output token representations to generate the final transcript. To maintain architectural coherence, the encoder and decoder are structured to have an identical number of Transformer blocks and the same width.

In the Whisper decoder, an autoregressive approach is adopted, similar to the GPT series of language models. The probability each token is sequentially determined based on the preceding context. Nonetheless, extensive parameter count of Whisper renders it challenging for downstream applications. Conventional fine-tuning strategies are computationally expensive, leading us to adopt prefixtuning [23], a more efficient alternative that utilizes a smaller set of trainable parameters.

Motivated by these advances, we introduce an optimized Whisper variant for lyric-based feature extraction from audio recordings. The detailed structure of it is shown in Figure 2. The audio encoder ingests chunked log mel spectrograms  $\mathbf{S}_{mel} \in \mathbb{R}^{N \times F \times T}$  and comprises blocks with self-attention and MLP layers. Residual connections are employed in both layers, culminating in audio features  $\mathbf{X}_{ae} \in \mathbb{R}^{N \times C_{ae}}$ .

For every segment, the text decoder process the corresponding audio feature  $\mathbf{X}_{\mathrm{ae}}(i) \in \mathbb{R}^{C_{\mathrm{ae}}}$  independently for transcribing. The text decoder starts with four initial tokens  $\mathbf{t}_{init}~\in~\mathbb{R}^4,$  which serve specific purposes such as indicating the start of prediction, speech presence, task specification, and timestamp prediction. These tokens pass through an embedding layer to yield embedded tokens  $\mathbf{E}_{\text{init}} \in \mathbb{R}^{4 \times C_l}$ , defined during the pretrain phase. A trainable suffix latent  $\mathbf{E}_e \in \mathbb{R}^{L_e \times C_l}$  is appended to these initial tokens. Additionally, before each text decoder block, a trainable prefix latent  $\mathbf{E}_{p_i} \in \mathbb{R}^{L_p imes C_l}$  is added, forming the input for the first text decoder block as  $\mathbf{E}_{in_1} = [\mathbf{E}_{p_1}; \mathbf{E}_{init}; \mathbf{E}_e] \in \mathbb{R}^{(L_p+4+L_e) \times C_l}$ . The prefix embeddings are learned to overwrite the instruction carried by init tokens, to reprogram Whisper to extract features that assit in the CSI task. Compared with the audio encoder block, the text decoder block incorporates an additional cross-attention layer. The decoder block consists of self-attention, cross-attention, and MLP layers, sequentially connected. The self-attention layer accepts the out-



**Figure 2**. The detailed structure of modified Whisper Model.

put of the preceding text decoder block, while the crossattention layer uses both the audio encoder output  $\mathbf{X}_{ae}(i)$ as its key and value and the output of self-attention layer as its query. The MLP layer subsequently processes the cross-attention output. After processing through the first text decoder block and subsequent truncation, we obtain  $\mathbf{E}_{\text{out}_1} \in \mathbb{R}^{(4+L_e) \times C_l}$ . The same process is repeated for subsequent text decoder blocks, each time adding a new set of trainable prefix latents. Since the first  $L_p$  output embeddings of each text decoder block are truncated, the input dimensions of each decoder block remain consistent even after adding a new set of prefix latents. The outputs of each text decoder block are stored to form an lyric-based feature group  $\mathbf{G} = [\mathbf{E}_{\text{out}_1}, \mathbf{E}_{\text{out}_2}, ..., \mathbf{E}_{\text{out}_B}] \in \mathbb{R}^{((4+L_e) \times B) \times C_l}.$ Finally, We select the last 70% to 80% of the audio-lyric feature group and the mean across the second dimension is taken to get the audio-lyric feature  $\mathbf{X}_{l}(i) \in \mathbb{R}^{C_{l}}$ . The medium version of the whisper we adopted contains 24 text decoder blocks, and the hidden size is 1024.

In terms of computational efficiency, the introduction of prefix latents enables the Whisper model to generate multiple embeddings in a single forward pass, thereby obviating the need for autoregressive operations during the decoding phase. This can be regarded as a Non-Autoregressive (NAR) method for efficient inference [24]. Despite this, certain studies [24] indicate that NAR performance is generally inferior to that of Autoregressive (AR) methods. To explore the performance upper bound of Whisper in CSI tasks, we also propose an AR-based feature extraction method. Importantly, in this AR approach, there are no modifications required to the Whisper model, and its

# parameters are kept frozen throughout the process. The Whisper model transcribes each audio segment and collects the final hidden states from each inference pass. These states are then aggregated using a mean operation to form the lyric-based feature $X_l(i)$ . Since the shape of the embeddings outputted by both NAR and AR methods is consistent, they can be used interchangeably. In section 4, we refer to the strategy using efficient Whisper adaptation as "*E*" and the original Whisper-based strategy as "*AR*". Lastly, we introduce a trainable linear projection layer subsequent to the Whisper output to map the embedding dimensions $C_l$ to 512. For clarity, the symbol $C_l$ continues to denote the dimensions of these projected embeddings.

# 3.3 A Bag of Tricks for Improving ByteCover3

We enhance our model's performance by incorporating three techniques: Sparse Softmax, Non-local Operations, and Grid Distortion. Due to the crowdsourced nature of the labels in the SHS100k dataset [9], mislabeling of version categories is inevitable. Additionally, the SHS100k dataset contains 8,858 cliques of music, indicating a high number of classes to distinguish when training models using softmax loss. Sun et al. [25] demonstrated that an increase in the number of classes raises the risk of overfitting with softmax loss, particularly when incorrect labels are present, as it may further degrade model performance by fitting these erroneous labels. Therefore, in X-Cover, we have opted to use Sparse Softmax [25] loss instead of the original softmax loss. Sparse Softmax loss retains only the top-K logits and the logits for the ground truth class during probability computation, effectively reducing the number of classes to distinguish to K + 1. However, a smaller K value during the initial phase of training might impair the model's ability to fit. Through tuning, we found that fine-tuning with K = 1024 after training with the original softmax loss achieves optimal results.

As demonstrated in [26], capturing global spectrogram information is crucial for understanding the complex compositions in music, characterized by varied spectral characteristics over time. However, the limited receptive field size of conventional CNNs restricts their ability to capture long-range dependencies across different parts of the spectrogram effectively. To address this limitation, we integrate Non-Local modules [27] into the ResNet-IBN architecture. These modules utilize the strength of Non-Local operations to compute interactions directly between any two positions in the input data, irrespective of their physical distance. This feature is particularly advantageous for analyzing music tracks, where distant sections may share thematic but transformed material, a characteristic common in cover songs.

Finally, Grid Distortion is borrowed from computer vision field, involves random scaling transformations in both the frequency and time dimensions of the spectrogram to simulate time-stretching and pitch shift.

# 4. EXPERIMENTS

We evaluated X-Cover using two publicly available datasets: SHS100K, which consists of 8,858 cover groups and 108,523 individual recordings [9], and Covers80, featuring 160 recordings that include two covers of each of the 80 songs [5]. The training and test division of SHS100K adheres to previous work [8,11-13], while Covers80 serves exclusively for testing. We convert all audio to CQT and Mel spectrograms before training. For CQT, we set the bins per octave to 12 and use a Hann window during extraction with a hop size of 512. All audio is resampled to 22,050 Hz before CQT conversion. We then downsample the CQT temporally by averaging over 100 adjacent frames to enhance computational efficiency and reduce latency. For Mel spectrograms, we follow the configuration in Whisper [17], resampling audio to 16,000 Hz and computing an 80-channel log-magnitude Mel spectrogram with 25-millisecond windows and a 10-millisecond stride. X-Cover's training phase uses weights from ByteCover3 for initialization. For the Whisper branch, we adopt configurations from the Whisper-Medium setup [17], and its initial weights are also sourced from pre-trained models.

Model	$\# Dims. \downarrow \qquad mAP \uparrow$		MR1↓		
	Covers80				
Me+Ha+Ly [14]	1536	0.993	1.02		
ByteCover3 [13]	512	0.927	3.32		
X-Cover-E	2560	0.992	1.04		
X-Cover-AR	2560 <b>1.000</b>		1.00		
	SHS100K-TEST				
MOVE [10]	512	0.519	154.5		
Me+Ha+Ly [14]	1536	0.794	39.3		
ByteCover3 [13]	512	0.824	37.0		
Whisper-E	512	0.437	150		
Whisper-AR	512	0.708	145.4		
ByteCover3.5	2048	0.857	22.7		
X-Cover-E	2560	0.889	14.9		
X-Cover-AR + PCA-FC [12]	512	0.924	14.7		
X-Cover-AR	2560	0.924	14.9		

Table 1. Performance on different datasets.

## 4.1 Comparison on Performance and Efficiency

As shown in Table 1, we benchmark the performance of various models, including our X-Cover, on two datasets: Covers80 and SHS100K-TEST. Metrics reported include the number of dimensions (Dims.), mean Average Precision (mAP), and Mean Rank 1 (MR1). For comparative analysis, we include MOVE [10], Me+Ha+Ly [14], and ByteCover3 [13]. Our X-Cover incorporate Whisper models trained with LAL loss, with and without efficient adaptation. By integrating these Whisper-based models with ByteCover3.5, we produce two hybrid solutions: X-Cover-*E* and X-Cover-*AR*. Notably ByteCover3.5 is an improved version over ByteCover3 which incorporates a series of enhancements described in Section 3.3. Addition-

Proceedings of the 25th ISMIR Conference, San Francisco, USA and Online, Nov 10-14, 2024

	ByteCover3.5	Whisper-E	Whisper-AR	X-Cover- $E$	X-Cover-AR
Feature Extraction(ms)	21	402	10467	452	10522

Table 2. The average inference time per audio

ally, the Me+Ha+Ly model [14] included in the comparison is not a strict replication of the original work which is a composite system comprised with a melody extraction model, the ALR model, and the MOVE [10]. Due to limited information in the original paper and the smaller size of ALR model compared to Whisper, our version of Me+Ha+Ly uses Basic-Pitch [28] and our Whisper-AR, ensuring a more equitable comparison. We will release the implementation publicly. All models are trained with the Adam Optimizer and a batch size of 128. Table 2 presents the average inference time for X-Cover variants on SHS100K-TEST using an NVIDIA A100 GPU.



Figure 3. Length of Queries vs. Performance.

Figure 3 displays the mAP results of X-Cover-*E* for different query lengths on SHS100K-TEST, using Re-MOVE [19], ByteCover2 [12] and ByteCover3 [13] as compared methods. As illustrated in the figure, our X-Cover-*E* model achieves the best mAPs for all query lengths except for the 6-second length. This clearly indicates the effectiveness of X-Cover-*E*. The poorer performance at the 6-second scenario is likely due to the shortness of the queries, which may not contain enough meaningful lyric information, thus not highlighting the strengths of X-Cover-*E*.

Overall, X-Cover-AR consistently outperforms other models on both datasets. Remarkably, X-Cover-AR, X-Cover-E and Me+Ha+Ly achieve nearly 100% accuracy on Covers80. Both methods employ Whisper-AR, making this high accuracy expected given Covers80 which is dataset with a limited size and predominantly consisting of recordings with vocal components. Therefore, we believe SHS100K-TEST is a more robust test of model performance. On this dataset, both X-Cover-AR and X-Cover-E achieve state-of-the-art performance. The performance gap between X-Cover-E and X-Cover-AR can be attributed to the differences in Whisper-AR and Whisper-E. The comparable performance and significantly faster speed of X-Cover-*E* validate our efficient adaptation. Surprisingly, Whisper-*AR*, cloned from an ASR model except for the final linear layer, shows comparable performance to other SoTA methods on SHS100K, highlighting the potential of large-scale pretrained ASR models in CSI tasks. However, the second-level inference time for individual samples in X-Cover-*AR* poses a challenge for its practical deployment in real-life scenarios. Finally, we employ the PCA-FC dimensionality reduction module from ByteCover2 to compress X-Cover-*AR* features from 2560 to 512 dimensions, finding negligible performance loss. This suggests that the performance gains in X-Cover variants are not due to increased feature size.

### 5. CONCLUSION

This paper has enhanced the robustness and efficiency of the ByteCover3 system in CSI by integrating lyric-related features using a pre-trained Automatic Speech Recognition model, Whisper. This integration addresses the issue of non-musical elements that distort musical characteristics essential for accurate CSI, without the need for training an additional lyrics recognition module from scratch.

The use of Whisper, adapted via prefix-tuning, significantly reduces the computational demands typically associated with large-scale ASR systems, thereby improving efficiency in both training and inference stages. Our results demonstrate improved accuracy and reliability of CSI, particularly in handling short queries against full songs.

In conclusion, our approach contributes to the development of more robust, efficient, and scalable CSI systems, enhancing both intellectual property management and music recommendation systems, especially in social video platforms. Future work will aim to further optimize these methods and explore additional features to increase resilience against noise in practical applications.

### 6. REFERENCES

- M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features." in *ISMIR*, vol. 2005. Citeseer, 2005, p. 6.
- [2] F. Yesiler, G. Doras, R. M. Bittner, C. J. Tralie, and J. Serrà, "Audio-based musical version identification: Elements and challenges," *IEEE Signal Processing Magazine*, vol. 38, no. 6, pp. 115–136, 2021.
- [3] M. F. Yesiler *et al.*, "Data-driven musical version identification: accuracy, scalability and bias perspectives," 2022.

- [4] J. Serra, "Identification of versions of the same musical composition by processing audio descriptions," *Department of Information and Communication Technologies*, 2011.
- [5] D. P. Ellis and G. E. Poliner, "Identifying cover songs with chroma features and dynamic programming beat tracking," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007.
- [6] J. Serra, E. Gómez, P. Herrera, and X. Serra, "Chroma binary similarity and local alignment applied to cover song identification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 6, pp. 1138–1151, 2008.
- [7] 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*, 2019, pp. 4846–4852.
- [8] Z. Yu, X. Xu, X. Chen, and D. Yang, "Learning a representation for cover song identification using convolutional neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), 2020, pp. 541–545.
- [9] X. Xu, X. Chen, and D. Yang, "Key-invariant convolutional neural network toward efficient cover song identification," in *IEEE International Conference on Multimedia and Expo*, 2018, pp. 1–6.
- [10] F. Yesiler, J. Serrà, and E. Gómez, "Accurate and scalable version identification using musically-motivated embeddings," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 21– 25.
- [11] X. Du, Z. Yu, B. Zhu, X. Chen, and Z. Ma, "Bytecover: Cover song identification via multi-loss training," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*). IEEE, 2021, pp. 551–555.
- [12] 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).* IEEE, 2022, pp. 616–620.
- [13] X. Du, Z. Wang, X. Liang, H. Liang, B. Zhu, and Z. Ma, "Bytecover3: Accurate cover song identification on short queries," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2023, pp. 1–5.
- [14] M. Abrassart and G. Doras, "And what if two musical versions don't share melody, harmony, rhythm, or lyrics?" in *ISMIR 2022*, 2022.

- [15] A. Vaglio, R. Hennequin, M. Moussallam, and G. Richard, "The words remain the same: Cover detection with lyrics transcription," in 22nd International Society for Music Information Retrieval Conference IS-MIR 2021, 2021.
- [16] G. Meseguer-Brocal, A. Cohen-Hadria, and G. Peeters, "Dali: A large dataset of synchronized audio, lyrics and notes, automatically created using teacherstudent machine learning paradigm," *arXiv preprint arXiv:1906.10606*, 2019.
- [17] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.
- [18] X. Pan, P. Luo, J. Shi, and X. Tang, "Two at once: Enhancing learning and generalization capacities via ibn-net," in *Proceedings of the european conference on computer vision (ECCV)*, 2018, pp. 464–479.
- [19] F. Yesiler, J. Serrà, and E. Gómez, "Less is more: Faster and better music version identification with embedding distillation," in *Proc. of the Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, 2020.
- [20] F. Zalkow, J. Brandner, and M. Müller, "Efficient retrieval of music recordings using graph-based index structures," *Signals*, vol. 2, no. 2, pp. 336–352, 2021.
- [21] K. Li, Z. Liu, T. He, H. Huang, F. Peng, D. Povey, and S. Khudanpur, "An empirical study of transformerbased neural language model adaptation," in *ICASSP* 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020.
- [22] Q. Yue, W. Shi, Y. He, J. Chu, Z. Han, and X. Han, "An improved speech recognition system based on transformer language model," in 2021 International Conference on Control, Automation and Information Sciences (ICCAIS), 2021.
- [23] X. Liu, K. Ji, Y. Fu, W. L. Tam, Z. Du, Z. Yang, and J. Tang, "P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks," *arXiv preprint arXiv:2110.07602*, 2021.
- [24] Y. Xiao, L. Wu, J. Guo, J. Li, M. Zhang, T. Qin, and T.y. Liu, "A survey on non-autoregressive generation for neural machine translation and beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
- [25] S. Sun, Z. Zhang, B. Huang, P. Lei, J. Su, S. Pan, and J. Cao, "Sparse-softmax: A simpler and faster alternative softmax transformation," *arXiv preprint arXiv:* 2112.12433, 2021.
- [26] X. Du, B. Zhu, Q. Kong, and Z. Ma, "Singing melody extraction from polyphonic music based on spectral

correlation modeling," in *ICASSP 2021-2021 IEEE In*ternational Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021, pp. 241–245.

- [27] X. Wang, R. Girshick, A. Gupta, and K. He, "Nonlocal neural networks," *CVPR*, 2018.
- [28] R. M. Bittner, J. J. Bosch, D. Rubinstein, G. Meseguer-Brocal, and S. Ewert, "A lightweight instrumentagnostic model for polyphonic note transcription and multipitch estimation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Singapore, 2022.