# Contrastive Lyrics Alignment with a Timestamp-Informed Loss

Timon Kick
*ETH Zurich*
tikick@ethz.ch

Florian Grötschla
*ETH Zurich*
fgroetschla@ethz.ch

Luca A. Lanzendörfer
*ETH Zurich*
lanzendoerfer@ethz.ch

Roger Wattenhofer
*ETH Zurich*
wattenhofer@ethz.ch

***Abstract*—Recent multimodal methods for lyrics alignment have relied on large datasets. Our approach introduces a box loss that directly incorporates timestamp information into the loss function, enabling precise alignment and competitive results even with limited training data. We also address the noise present in the public DALI dataset, conducting a thorough cleaning process to improve the quality of training data. Finally, we propose JamendoLyrics++, a substantial extension of the common JamendoLyrics evaluation dataset, offering improved genre diversity for better evaluation of lyrics alignment systems.**

***Index Terms*—lyrics alignment, audio signal processing, open-source code, open-source dataset**

## I. Introduction

Lyrics alignment is the task of synchronizing the lyrics of a song with its audio. This process can be executed at various levels of granularity, such as lines, words, or even characters, depending on the required temporal precision. Lyrics alignment has various applications, including karaoke systems, lyrics-based music retrieval, and enhancing user experience in music streaming services [1].

Early approaches to lyrics alignment faced significant challenges due to the lack of polyphonic training datasets [2]. These methods often adapted automatic speech recognition (ASR) systems to the solo singing domain [3]–[5], using the DAMP a cappella singing dataset [6]. However, singing is generally more complex than speech, exhibiting a wider pitch and temporal range. Furthermore, the domain mismatch between solo singing and polyphonic audio resulted in poor performance on the latter.

Results in lyrics alignment improved substantially with the release of the DALI v1 dataset [7], providing time-aligned lyrics annotations for approximately 5k polyphonic songs, as well as the use of larger internal datasets. Gupta et al. investigated music-informed audio features [8] and genre-informed phone and silence models [9]. Demirel et al. explored low-resource lyrics alignment using anchor word selection followed by anchor segmentation [10]. Huang et al. proposed a multitask learning approach [11] using pitch timestamps provided in DALI v2 [12].

These and other methods [13], [14] have been trained with a Connectionist Temporal Classification (CTC) loss [15]. The recent work by Durand et al. is particularly novel for its use of a multimodal contrastive learning approach [16]. They achieved state-of-the-art performance using a large internal dataset of approximately 88k songs. That same year, Kang et al. presented a similar multimodal method [17]. Their DALI-trained model was not able to achieve the same level of performance; however, with a large internal dataset of approximately 67k songs and ensembling techniques, they achieved competitive results. This raises the question of whether all multimodal methods require large amounts of data.

In this work, we build on the contrastive framework proposed by [16], and demonstrate that it is possible to achieve competitive results using a fraction of the original training data. We propose improvements by incorporating timestamp information directly into the contrastive loss, which, to the best of our knowledge, is the first instance where such information is used within the loss rather than only for generating training samples. This addition improves the model's performance, making it competitive when trained only on DALI. This not only shows that multimodal methods can excel with limited data but also ensures a fair comparison with other state-of-the-art methods trained on DALI [9], [11]. Additionally, we conduct a detailed analysis of the failure cases of our model, discovering and partially cleaning noisy samples in the DALI dataset. Furthermore, we improve on JamendoLyrics [18], a commonly used test set, by proposing JamendoLyrics++, which contains four times more manually annotated samples. The original JamendoLyrics dataset is limited by its small size, consisting of only 20 English songs. JamendoLyrics++ offers a larger, more comprehensive, and genre-diverse collection, enabling more robust comparisons and more accurate generalization estimates of model performance across different musical genres.

Our contributions can be summarized as follows:

- We propose a novel loss for contrastive lyrics alignment that incorporates timestamp information. We open-source our code and checkpoints for open science and reproducibility.[1]
- We analyze DALI v2 and identify noisy samples, providing labels to facilitate dataset cleaning.
- We propose JamendoLyrics++,[2] an extension of JamendoLyrics with four times more data and high genre diversity.

[1]https://github.com/tikick/LyricsAlignment
[2]https://github.com/tikick/jamendolyricspp

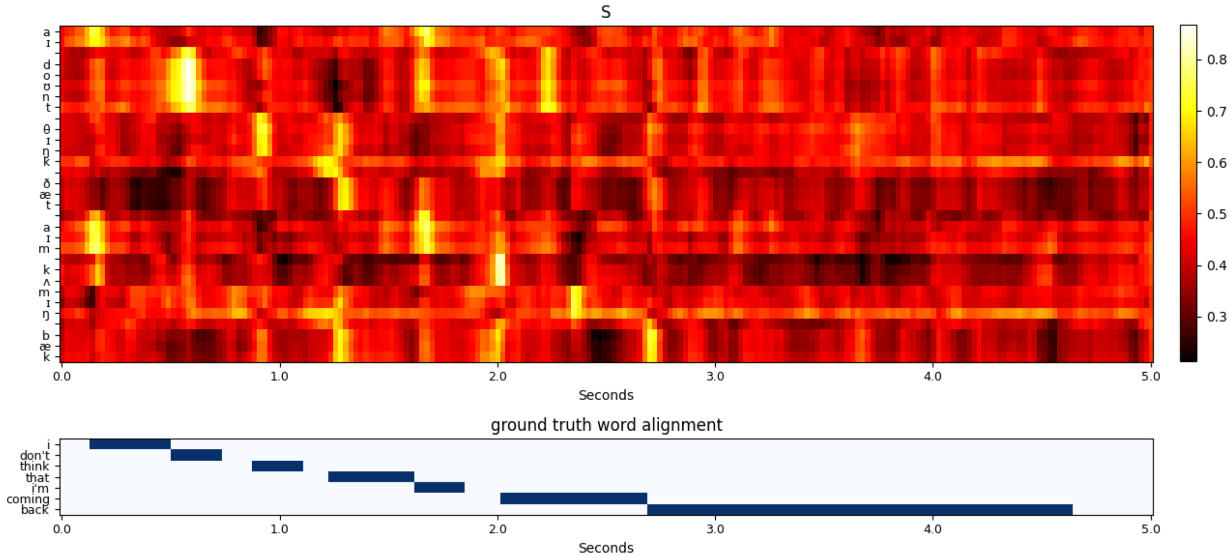


Fig. 1. The similarity matrix above, and the ground truth word alignment below. Observe the vertical bright stripes at the start of each word.

## II. Methods

We base our approach on the similarity model by [16], proposing improvements and experimental modifications. In the following we present the different components of the contrastive learning framework. Specifically, we describe the text and audio encoders, the similarity matching and contrastive learning procedure for training, and the alignment decoding. For an overview figure and more details refer to the original paper. We conclude with a few comments.

**Audio Encoder.** The audio encoder $f_a$ is designed to detect the phonetic content of the singing voice in the audio. It processes input spectrograms $\mathbf{X} \in \mathbb{R}^{T \times D}$ with a duration of 5 seconds, where $T$ is the number of spectrogram frames and $D$ is the number of frequency bins. The encoder is a residual network comprising 10 residual convolutional blocks (RCBs), each containing 2 repetitions of group normalization, ReLU activation, and a 2D convolutional layer with a $3 \times 3$ kernel and 64 features. The output is a 1D convolution layer applied on each time bin with $E = 64$ filters, resulting in an embedding matrix $\mathbf{A} \in \mathbb{R}^{T \times E}$.

**Text Encoder.** The text encoder $f_l$ estimates how the singing voice could sound for any given lyrics symbol. To account for the pronunciation dependence on neighboring symbols, the encoder processes the subsequence $s_{n-C}, \ldots, s_n, \ldots, s_{n+C}$ for each symbol $s_n$ in the lyrics, which could be characters, phonemes, or other text representations. These symbols are passed through a trainable embedding layer, and a simple dense network with one hidden layer and ReLU activation, yielding an $E$-dim. embedding for each symbol. A given $N$-symbol lyrics sequence is thus mapped to an embedding matrix $\mathbf{L} \in \mathbb{R}^{N \times E}$. Both the text and audio encoder embeddings are $l_2$ normalized to enable cosine similarity comparisons.

**Similarity Matching and Training.** For training, a contrastive learning approach is employed. Positive examples $s^+$ are taken from the lyrics corresponding to a given audio segment, while negative examples $s^-$ are sampled from the distribution $p_s$ over symbols obtained from all lyrics in the dataset that do not appear in the audio segment. The similarity between the text and audio embeddings is maximized for positive pairs and minimized for negative pairs with the following objective

$$L = \mathbb{E}_{(\mathbf{X},s^+)\sim p_d} \left[ (m(\mathbf{X}, s^+) - 1)^2 + \mathbb{E}_{s^- \sim p_s} m(\mathbf{X}, s^-)^2 \right], \tag{1}$$

where $p_d$ is the distribution over audio segments and symbols sampled from the corresponding lyrics sequence, and $m(\mathbf{X}, s) = \max_{t \in [1,T]} f_l(s) \cdot f_a(\mathbf{X})_t^\top$ is the maximum similarity of symbol $s$ over the entire audio segment $\mathbf{X}$.

**Alignment Decoding.** Post-training, the alignment is performed on a normalized similarity matrix $\mathbf{S} = \frac{1}{2}(\mathbf{A} \cdot \mathbf{L}^\top + 1)$, ensuring that $\mathbf{S} \in [0,1]^{T \times N}$. A similarity matrix example is shown in Fig. 1. The alignment is decoded using a modified Dynamic Time Warping (DTW) algorithm [19], which excludes horizontal score accumulation and vertical steps. That is, the algorithm finds a monotonic path that maximizes the cumulative similarity score across the diagonal steps. This decoding algorithm is applied on the log-transformed similarity matrix. The authors thus interpret the similarity matrix $\mathbf{S}$ as a probability matrix and decode the most likely path in the log space.

To address start- and end-of-line words misalignment, an estimated line position is used to constrain the alignment process.

**Frames Concentration.** Inspecting the similarity matrix $\mathbf{S}$ (see Fig. 1) reveals that the model concentrates all syllable/word information into the first few corresponding frames, which is suboptimal for predicting token alignment and word ends. Since current evaluation metrics only use word starts,

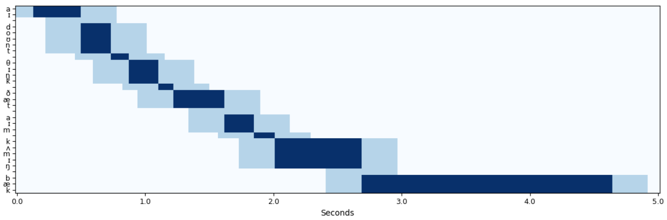


Fig. 2. The box loss. The maximum similarity of a lyrics token is taken withing the corresponding box and not the entire audio segment. To account for noisy timestamps, a slack hyperparameter widens the box (in light blue).

this weakness remains hidden.

**Multi-Loss.** We also experiment with a multi-loss approach. Most lyrics alignment models are trained with a CTC loss, while [16] use a contrastive loss. To explore potential synergies, we added a linear layer to the audio encoder to obtain a posteriorgram, i.e., a frame-wise distribution over symbols, in addition to the frame embeddings. We train the model using a linear combination of contrastive and CTC loss. We experimented with combining the contrastive and CTC loss in various proportions, obtaining comparable results to the contrastive loss alone. This suggests that the two losses are collinear rather than complementary. When placing high weight on the CTC loss, we observed a slight performance decrease, consistent with the results of the CTC-only model reported by [16]. As the multi-loss underperformed models that were only trained with the contrastive loss, we continued our experiments using only the contrastive loss.

### A. Box Loss

We propose a novel box loss that incorporates timestamp information directly into the contrastive loss. To our knowledge, this is the first instance where timestamp information is used within the loss function itself rather than only for generating training samples. Specifically, we take the maximum over the frames where the token appears according to the timestamps, rather than over the entire audio segment as in Eq. 1, see Fig. 2. To account for potentially noisy timestamps in the training dataset, we introduce a slack hyperparameter to widen the box. That is, we redefine the $m$ function for positive symbols $s$ as follows

$$m(\mathbf{X}, s) = max_{t\in[t^s_{start}-\zeta, t^s_{end}+\zeta]} f_l(s) \cdot f_a(\mathbf{X})^\top_t ,$$

where $t^s_{start}, t^s_{end}$ are the start and end frames of symbol $s$, and $\zeta$ is the slack hyperparameter.

In addition, we explore a variant of the box loss, termed negative box loss, which eliminates the need for negative sampling and instead designates tokens appearing only once within a segment as negatives outside their defined box. Formally, the objective is

$$L = \mathbb{E}_{(\mathbf{X},s^+)\sim p_d} \left[ (m(\mathbf{X}, s^+) - 1)^2 + m^-(\mathbf{X}, s^+)^2 \right] ,$$

where $m$ is defined as above and

$$m^-(\mathbf{X}, s) = max_{t\in[1,t^s_{start}-\zeta)\cup(t^s_{end}+\zeta,T]} f_l(s) \cdot f_a(\mathbf{X})^\top_t$$

if $s$ is unique within $\mathbf{X}$ and 0 otherwise.

## III. Experiments

### A. Evaluation Dataset and Metrics

For evaluation, we use the common JamendoLyrics dataset, which contains 20 songs annotated at the word level. We compute two standard evaluation metrics: the Average Absolute Error

$$AAE = \frac{\sum_{w=1}^{W} |t^w_{pred} - t^w_{gt}|}{W},$$

and the Percentage of Correct Onsets with a tolerance window of 0.3 seconds

$$PCO = \frac{\sum_{w=1}^{W} \mathbf{1}\{|t^w_{pred} - t^w_{gt}| < 0.3\}}{W},$$

where $W$ is the number of words in a song, and $t^w_{pred}, t^w_{gt}$ are the predicted and ground truth start time of the $w$-th word. These metrics are averaged over all JamendoLyrics songs.

### B. Training Dataset

The framework by [16], along with all experimental modifications, is trained on the DALI dataset. The first DALI version comprises 5,358 English songs with word-level lyrics annotations. DALI v2 extends this to 7,756 songs, including other languages. For our experiments, we use the English subset of DALI v2 with available audio, which amounted to 4,899 songs. Analogous to [16], we reserve 2% of the training dataset for validation. A 5-second sliding window with a hop size of 2.5 seconds is applied to generate the samples. The target lyrics for an audio segment are the words fully contained within the window. We use a text-to-phone converter [20] to obtain IPA characters as text representation.

### C. DALI Cleaning

Early experiments suggested substantial noise in the DALI dataset, as the best box slack was quite large (1.5 seconds), and the JamendoLyrics and validation metrics were significantly different. To investigate this, we used a model trained on an internal dataset (to avoid bias) to compute the PCO score for each DALI song. We inspected songs with a PCO below 80% (around 1k songs) for patterns in the deviation between ground truth and predicted timestamps, issues with lyrics, and other anomalies. We documented our findings in a CSV file available on our GitHub repository.

Our analysis revealed that approximately 10% of the DALI songs have issues. About 100 songs have mismatches between lyrics and audio, such as wrong lyrics (interestingly, a significant number of these entries contain the Tetris lyrics by the Brentalfloss YouTube channel), English lyrics with non-English singing, or audio without singing (karaoke). Another 100 songs display issues only towards the end, such as additional lyrics paragraphs. Moreover, we identified various types of timestamp offsets among more than 200 songs: a constant global offset, a linearly increasing or decreasing offset, and different local offsets. Most offsets are quite small, not exceeding 1 second, but some are as large as 8 seconds or more.

TABLE I
PERFORMANCE COMPARISON

| Evaluation Data | JamendoLyrics | | | | JamendoLyrics++ | | | |
|---|---|---|---|---|---|---|---|---|
| Training Data | DALI | | DALI Clean | | DALI | | DALI Clean | |
| Metric | PCO | AAE | PCO | AAE | PCO | AAE | PCO | AAE |
| Contrastive | 92% | 0.25 | 93% | 0.24 | 95% | 0.28 | 94% | 0.22 |
| Box | 93% | 0.24 | 94% | 0.20 | 95% | 0.22 | 95% | 0.20 |
| Negative Box | 90% | 0.41 | 90% | 0.47 | 91% | 0.54 | 91% | 0.53 |

In an attempt to clean DALI, we correct timestamps with constant offsets and remove all other noisy songs. This not only ensures that the data we train on is cleaner, but also that the measured validation performance is more precise. Note that some songs, such as those with additional lyrics at the end, do not pose a problem during training (as the audio is "missing" and no training samples are created), but do pose a problem during validation (all words might need to be predicted earlier to make the additional lyrics fit).

**Training.** We train our models for 16 epochs and choose the checkpoint that achieves the highest PCO score on the validation subset.

## IV. RESULTS

### A. Box Loss and DALI Cleaning

Table I presents the performance comparison of our models using different losses and datasets. An inspection of our models' predictions revealed that most timestamps were slightly delayed. We thus shifted all predictions forward by 0.1 seconds. Compared to the contrastive loss baseline, the box loss provides a noticeable improvement in both evaluated metrics. This, however, is not the case for the negative box loss. Cleaning the training data also contributed to improved performance, with the exception of the negative box loss model, where performance remained unchanged. This highlights the potential benefit of reducing noise in datasets such as DALI. We encourage other researchers working on lyrics alignment to consider the noise in DALI.

We examined the failure cases of our models. Most challenging songs from both DALI and JamendoLyrics contain repeated syllables (e.g. "la la la", "who oh oh") or repeated words and lines. We suspect these are challenging songs for many lyrics alignment systems. Missing a single repetition can shift the alignment of subsequent repetitions; although most lyrics and audio may still match overall, this misalignment negatively impacts both the PCO and AAE metrics.

We also conducted experiments using the Mel-Roformer source-separated vocals instead of mixed audio [21]. Contrary to expectations, this significantly worsened performance.

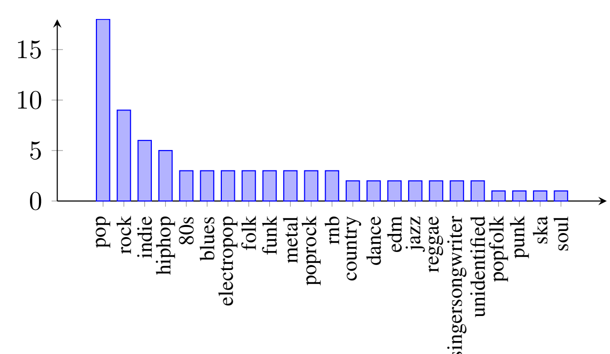


Fig. 3. Distribution of genres in JamendoLyrics++. We observe that our dataset covers a wide range of genres with focus on the popular and broad genres pop and rock.

TABLE II
COMPARISON WITH STATE-OF-THE-ART MODELS

| | PCO | AAE | Training Dataset |
|---|---|---|---|
| DSE [16] | 93% | **0.16** | Internal 88k dataset |
| GYL [9] | **94%** | 0.22 | DALI |
| HBE [11] | **94%** | 0.23 | DALI |
| Ours | **94%** | 0.20 | DALI |

### B. Comparison with the State-of-the-Art

Table II compares our best-performing model against state-of-the-art models from the literature on the JamendoLyrics dataset.

## V. JAMENDOLYRICS++

Finally, we introduce JamendoLyrics++, an 80-song dataset that serves as a substantial extension to the original JamendoLyrics dataset. This new dataset is a carefully curated subset of songs from the Jamendo platform, selected from those that provide accompanying lyrics. Combined with the original JamendoLyrics dataset of 20 English songs, this yields a total of 100 songs for lyrics alignment evaluation. As depicted in Fig. 3, JamendoLyrics++ covers over 20 musical styles, with a focus on the popular and very broad genres pop and rock [22], which supports more comprehensive benchmarking of lyrics alignment models across different genres.

We processed the provided lyrics to ensure high-quality data. In particular, we added missing repetitions of choruses or individual words and lines, removed any tags, and made small corrections when the lyrics deviated from what was actually sung. We employed a two-phase process to create the timestamps. First, we used one of our models to generate initial, noisy timestamps. Then, to ensure precise annotations, we manually corrected and refined the timestamps with a graphical interface. Results are reported in Table I and align with those in JamendoLyrics. An exception is the contrastive loss trained on DALI, with the same PCO score as the box loss. This is likely due to the initial JamendoLyrics++ timestamps coming from this model, thus introducing a bias.

## VI. CONCLUSIONS

We have introduced a timestamp-informed box loss for contrastive lyrics alignment, demonstrating its effectiveness in achieving competitive results with reduced training data. Our approach also highlights the importance of dataset quality, as shown through our DALI dataset cleaning efforts. Finally, the release of the JamendoLyrics++ dataset offers a more robust benchmark for future research.

## REFERENCES


[1] Hiromasa Fujihara and Masataka Goto, "Lyrics-to-audio alignment and its application," in *Multimodal Music Processing*, vol. 3 of *Dagstuhl Follow-Ups*, pp. 23–36. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Germany, 2012.

[2] Annamaria Mesaros and Tuomas Virtanen, "Automatic alignment of music audio and lyrics," in *Proceedings of the 11th Int. Conference on Digital Audio Effects (DAFx-08)*, 2008.

[3] Chitralekha Gupta, Rong Tong, Haizhou Li, and Ye Wang, "Semi-supervised lyrics and solo-singing alignment," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2018, pp. 600–607.

[4] Bidisha Sharma, Chitralekha Gupta, Haizhou Li, and Ye Wang, "Automatic lyrics-to-audio alignment on polyphonic music using singing-adapted acoustic models," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 396–400, IEEE.

[5] Anna Kruspe, "Lyrics alignment using hmms, posteriorgram-based dtw, and phoneme-based levenshtein alignment," .

[6] Smule, "Digital archive mobile performances (damp)," https://ccrma.stanford.edu/damp/, Accessed: 2024-08-05.

[7] Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Gaël Peeters, "Dali: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[8] Chitralekha Gupta, Emre Yılmaz, and Haizhou Li, "Acoustic modeling for automatic lyrics-to-audio alignment," in *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 2040–2044, ISCA.

[9] Chitralekha Gupta, Emre Yılmaz, and Haizhou Li, "Automatic lyrics alignment and transcription in polyphonic music: Does background music help?," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 496–500, IEEE.

[10] Emir Demirel, Sven Ahlbäck, and Simon Dixon, "Low resource audio-to-lyrics alignment from polyphonic music recordings," 2021.

[11] Jiawen Huang, Emmanouil Benetos, and Sebastian Ewert, "Improving lyrics alignment through joint pitch detection," 2022.

[12] Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Gaël Peeters, "Creating dali, a large dataset of synchronized audio, lyrics, and notes," *Transactions of the International Society for Music Information Retrieval*, vol. 3, no. 1, pp. 55–67, 2020.

[13] Daniel Stoller, Simon Durand, and Sebastian Ewert, "End-to-end lyrics alignment for polyphonic music using an audio-to-character recognition model," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, IEEE.

[14] Andrea Vaglio, Romain Hennequin, Manuel Moussallam, Gaël Richard, and Florence d'Alché Buc, "Multilingual lyrics-to-audio alignment," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[15] Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*. 2006, vol. 148, pp. 369–376, ACM.

[16] Simon Durand, Daniel Stoller, and Sebastian Ewert, "Contrastive learning-based audio to lyrics alignment for multiple languages," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. June 2023, IEEE.

[17] Minsung Kang, Soochul Park, and Keunwoo Choi, "Hclas-x: Hierarchical and cascaded lyrics alignment system using multimodal cross-correlation," 2023.

[18] Emir Demirel, Daniel Stoller, and Simon Durand, "JamendoLyrics Multi-Lang – an evaluation dataset for multi-language lyrics research," https://github.com/f90/jamendolyrics/, 2023, Accessed: 2024-09-06.

[19] Meinard Müller, "Dynamic time warping," in *Information Retrieval for Music and Motion*. Springer, Berlin, Heidelberg, 2007.

[20] Mathieu Bernard and Hadrien Titeux, "Phonemizer: Text to phones transcription for multiple languages in python," *Journal of Open Source Software*, vol. 6, no. 68, pp. 3958, 2021.

[21] Ju-Chiang Wang, Wei-Tsung Lu, and Minz Won, "Mel-band roformer for music source separation," *arXiv preprint arXiv:2310.01809*, 2023.

[22] YouGov, "What are the most popular music genres around the world?," https://business.yougov.com/content/48874-what-are-the-most-popular-music-genres-around-the-world, Accessed: 2024-09-09.