



Lyrics segmentation via bimodal text–audio representation

Michael Fell, Yaroslav Nechaev, Gabriel Meseguer-Brocal, Elena Cabrio,
Fabien Gandon, Geoffroy Peeters

► To cite this version:

Michael Fell, Yaroslav Nechaev, Gabriel Meseguer-Brocal, Elena Cabrio, Fabien Gandon, et al.. Lyrics segmentation via bimodal text–audio representation. *Natural Language Engineering*, 2022, 28 (3), pp.317 - 336. 10.1017/S1351324921000024 . hal-03295581

HAL Id: hal-03295581

<https://hal.science/hal-03295581>

Submitted on 14 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Lyrics Segmentation via Bimodal Text-audio Representation^{†‡}

Michael Fell^{1E}, Yaroslav Nechaev², Gabriel Meseguer-Brocal³,
Elena Cabrio¹, Fabien Gandon¹ and Geoffroy Peeters⁴

¹*Université Côte d’Azur, CNRS, Inria, I3S, France*

²*Amazon, Cambridge, MA, USA*

³*Ircam Lab, CNRS, Sorbonne Université, France*

⁴*LTCI, Télécom Paris, Institut Polytechnique de Paris, France*

^E*Corresponding author. Email: mic.fell@gmail.com*

(Received 14 October 2021)

Abstract

Song lyrics contain repeated patterns that have been proven to facilitate automated lyrics segmentation, with the final goal of detecting the building blocks (e.g. chorus, verse) of a song text. Our contribution in this article is two-fold. First, we introduce a convolutional neural network-based model that learns to segment the lyrics based on their repetitive text structure. We experiment with novel features to reveal different kinds of repetitions in the lyrics, for instance based on phonetical and syntactical properties. Second, using a novel corpus where the song text is synchronized to the audio of the song, we show that the text and audio modalities capture complementary structure of the lyrics and that combining both is beneficial for lyrics segmentation performance. For the purely text-based lyrics segmentation on a dataset of 103k lyrics, we achieve an f-score of 67.4%, improving on the state of the art (59.2% f-score). On the synchronized text-audio dataset of 4.8k songs, we show that the additional audio features improve segmentation performance to 75.3% f-score, significantly outperforming the purely text-based approaches.

1 Introduction

Understanding the structure of song lyrics (e.g. intro, verse, chorus) is an important task for music content analysis (Cheng *et al.*, 2009; Watanabe *et al.*, 2016) since it allows to split a song into semantically meaningful segments enabling a description of each section rather than a global description of the whole song. The importance of this task arises also in Music Information Retrieval, where music structure detection is a research area aiming at automatically estimating the temporal structure of a

[†] This article is an extended version of (Fell *et al.*, 2018).

[‡] This work is partly funded by the French Research National Agency (ANR) under the WASABI project (contract ANR-16-CE23-0017-01).

music track by analyzing the characteristics of its audio signal over time. Given that lyrics contain rich information about the semantic structure of a song, relying on textual features could help in overcoming the existing difficulties associated with large acoustic variation in music. However, so far only a few works have addressed the task lyrics-wise (Fell *et al.*, 2018; Mahedero *et al.*, 2005; Watanabe *et al.*, 2016; Baratè *et al.*, 2013). Carrying out structure detection by means of an automated system is therefore a challenging but useful task, that would allow to enrich song lyrics with improved structural clues that can be used for instance by search engines handling real-world large song collections. A step forward, a complete music search engine should support search criteria exploiting both the audio and the textual dimensions of a song.

Structure detection consists of two steps: a text segmentation stage that divides lyrics into segments, and a semantic labelling stage that labels each segment with a structure type (e.g. intro, verse, chorus). Given the variability in the set of structure types provided in the literature according to different genres (Tagg, 1982; Brackett, 1995), rare attempts have been made to achieve the second step, i.e. semantic labelling. While addressing the first step is the core contribution of this paper, we leave the task of semantic labelling for future work.

In (Fell *et al.*, 2018) we proposed a first neural approach for lyrics segmentation that was relying on purely textual features. However, with this approach we fail to capture the structure of the song in case there is no clear structure in the lyrics - when sentences are never repeated or in the opposite case when they are always repeated. In such cases however, the structure may arise from the acoustic/audio content of the song, often from the melody representation. This paper aims at extending the approach proposed in (Fell *et al.*, 2018) by complementing the textual analysis with acoustic aspects. We perform lyrics segmentation on a synchronized text-audio representation of a song to benefit from both textual and audio features.

In this direction, this work focuses on the following research question: *given the text and audio of a song, can we learn to detect the lines delimiting segments in the song text?* This question is broken down into two sub questions: *1) given solely the song text, can we learn to detect the lines delimiting segments in the song?* and *2) do audio features - in addition to the text - boost the model performance on the lyrics segmentation task?*

To address these questions, this article contains the following contributions. Contributions 1a) and 1b) have been previously published in (Fell *et al.*, 2018), while 2) is a novel contribution.

- 1a) We introduce a convolutional neural network-based model that *i)* efficiently exploits the Self-Similarity Matrix representations (SSM) used in the state-of-the-art (Watanabe *et al.*, 2016), and *ii)* can utilize traditional features alongside the SSMs (see Section 2 until 2.2).
- 1b) We experiment with novel features that aim at revealing different properties of a song text, such as its phonetics and syntax. We evaluate this **unimodal** (purely text-based) approach on two standard datasets of English lyrics, the

Music Lyrics Database and the WASABI corpus (see Section 3.1). We show that our proposed method can effectively detect the boundaries of music segments outperforming the state of the art, and is portable across collections of song lyrics of heterogeneous musical genre (see Sections 3.2-3.4).

- 2) We experiment with a **bimodal** lyrics representation (see Section 2.3) that incorporates audio features into our model. For this, we use a novel bimodal corpus (DALI, see Section 4.1) in which each song text is time-aligned to its associated audio. Our bimodal lyrics segmentation performs significantly better than the unimodal approach. We investigate which text and audio features are the most relevant to detect lyrics segments and show that the text and audio modalities complement each other. We perform an ablation test to find out to what extent our method relies on the alignment quality of the lyrics-audio segment representations (see Sections 4.2-4.4).

To better understand the rationale underlying the proposed approach, consider the segmentation of the Pop song depicted in Figure 1. The left side shows the lyrics and its segmentation into its structural parts: the horizontal green lines indicate the segment borders between the different lyrics segments. We can summarize the segmentation as follows: Verse₁-Verse₂-Bridge₁-Chorus₁-Verse₃-Bridge₂-Chorus₂-Chorus₃-Chorus₄-Outro. The middle of Figure 1 shows the repetitive structure of the lyrics. The exact nature of this structure representation is introduced later and is not needed to understand this introductory example. The crucial point is that the segment borders in the song text (green lines) coincide with highlighted rectangles in the chorus (the C_i) of the lyrics structure (middle). We find that in the verses (the V_i) and bridges (the B_i) highlighted rectangles are only found in the melody structure¹ (right). The reason is that these verses have different lyrics, but share the same melody (analogous for the bridges). While the repetitive structure of the lyrics is an effective representation for lyrics segmentation, we believe that an enriched segment representation that also takes into account the audio of a song can improve segmentation models. While previous approaches relied on purely textual features for lyrics segmentation, showing the discussed limitations, we propose to perform lyrics segmentation on a synchronized text-audio representation of a song to benefit from both textual and audio features.

Earlier in this section, we presented our research questions and motivation, along with a motivational example. In the remainder of the paper, in Section 2 we define the task of classifying lines as segment borders, the classification methods we selected for the task, and the bimodal text-audio representation. In Section 3 we describe our lyrics segmentation experiments using text lines as input. Then, Section 4 describes our lyrics segmentation experiments using multimodal (unimodal or bimodal) lyrics lines, containing text or audio information or both as input. We follow up with a shared error analysis for both experiments in Section 5. In Section 6 we position our work in the current state of the art, and in Section 7

¹ Technically, what we show is a part of the audio structure, based on chroma features. We describe these features and their connection to the melody in more detail in Section 2.3. For the purpose of a simpler presentation, we often call this the melody structure.

we conclude with future research directions to provide more metadata to music information retrieval systems.

2 Modelling Segments in Song Lyrics

Detecting the structure of a song text is a non-trivial task that requires diverse knowledge and consists of two steps: text segmentation followed by segment labelling. In this work we focus on the task of segmenting the lyrics. This first step is fundamental to segment labelling when segment borders are not known. Even when segment borders are “indicated” by line breaks in lyrics available online, those line breaks have usually been annotated by users and neither are they necessarily identical to those intended by the songwriter, nor do users in general agree on where to put them. Thus, a method to automatically segment unsegmented song texts is needed to automate that first step. Many heuristics can be imagined to find the segment borders. In our example, separating the lyrics into segments of a constant length of four lines (Figure 1) gives the correct segmentation. However, in another example, the segments can be of different length. This is to say that enumerating heuristic rules is an open-ended task.

We follow (Watanabe *et al.*, 2016) by casting the lyrics segmentation task as binary classification. Let $L = \{a_1, a_2, \dots, a_n\}$ be the lyrics of a song composed of n **text lines** and $seg : L \rightarrow \mathbb{B}$ be a function that returns for each line $a_i \in L$ if it is the end of a segment. Here, $\mathbb{B} = \{0, 1\}$ is the Boolean domain. The task is to learn a classifier that approximates seg . At the learning stage, the ground truth segment borders are observed as double line breaks in the lyrics. At the testing stage, we hide the segment borders and the classifier has to predict them.

As lyrics are texts that accompany music, their text lines do not exist in isolation. Instead, each text line is naturally associated to a segment of audio. We define a **bimodal lyrics line** $a_i = (l_i, s_i)$ as a pair containing both the i -th text line l_i , and its associated **audio segment** s_i . In the case we only use the text lines, we model this as unimodal lyrics lines, i.e. $a_i = (l_i)$.²

In order to infer the lyrics structure, we rely on our Convolutional Neural Network-based model that we introduced in (Fell *et al.*, 2018). Our model architecture is detailed in Section 2.2. It detects segment boundaries by leveraging the repeated patterns in a song text that are conveyed by the Self-Similarity Matrices.

² This definition can be straightforwardly extended to more modalities, a_i then becomes a tuple containing time-synchronized information.

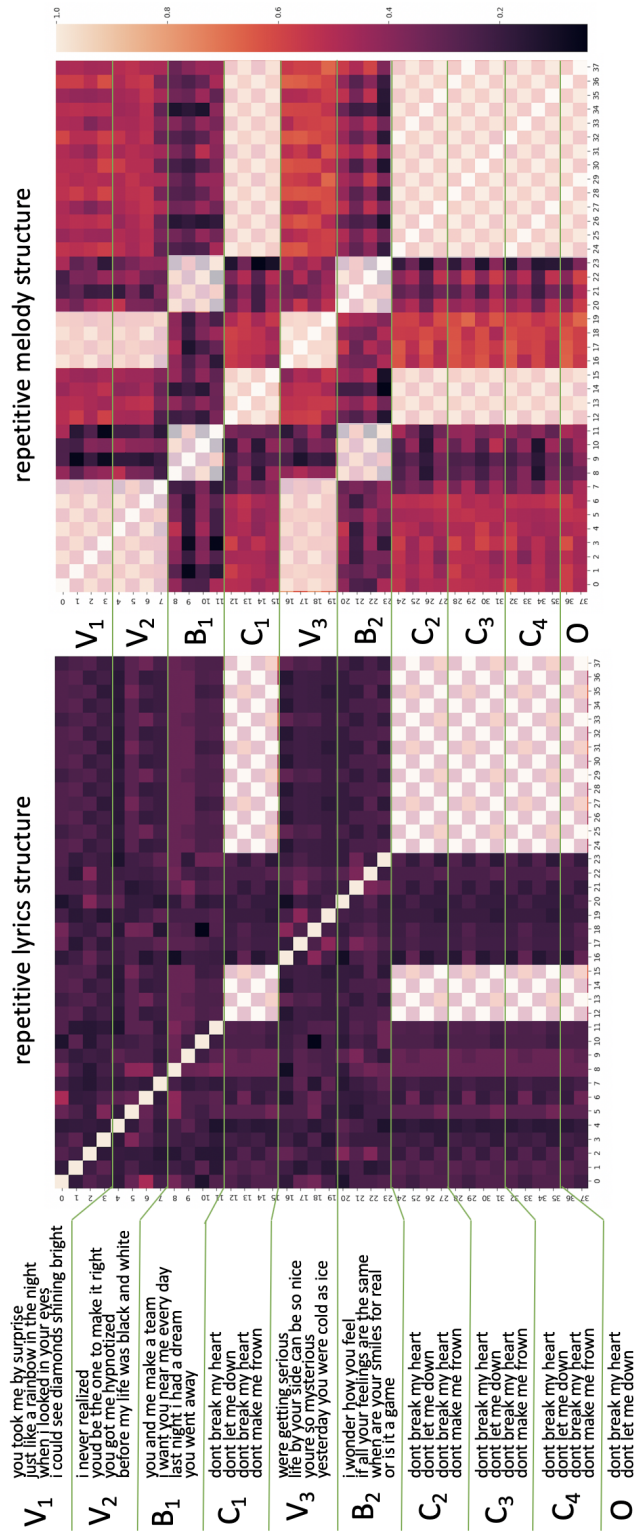


Fig. 1: Lyrics (left) of a Pop song, the repetitive structure of the lyrics (middle), and the repetitive structure of the song melody (right). Lyrics segment borders (green lines) coincide with highlighted rectangles in lyrics structure and melody structure. (“Don’t Break My Heart” by Den Harrow)

2.1 Self-Similarity Matrices

We produce Self-Similarity Matrices (SSMs) based on bimodal lyrics lines $a_i = (l_i, s_i)$ in order to capture repeated patterns in the text line l_i as well as its associated audio segment s_i . SSMs have been previously used in the literature to estimate the structure of music (Foote, 2000; Cohen-Hadria & Peeters, 2017) and lyrics (Watanabe *et al.*, 2016; Fell *et al.*, 2018). Given a song consisting of lyrics lines $\{a_1, a_2, \dots, a_n\}$, a Self-Similarity Matrix $SSM_M \in \mathbb{R}^{n \times n}$ is constructed, where each element is set by computing a similarity measure between the two corresponding elements $(SSM_M)_{ij} = \text{sim}_M(x_i, x_j)$. We choose x_i, x_j to be elements from the same modality, i.e. they are either both text lines (l_i) or both audio segments (s_i) associated to text lines. sim_M is a similarity measures that compares two elements of the same modality to each other. In the unimodal case, we compute SSMs from only one modality: either text lines l_i or audio segments s_i .

As a result, SSMs constructed from a text-based similarity highlight distinct patterns of the text, revealing the underlying structure. Analogously, SSMs constructed from an audio-based similarity highlight distinct patterns of the audio. In our motivational example, the textual SSM encodes how similar the text lines are on a character level (see Figure 1, middle) while the audio SSM encodes how similar the associated melodies are to each other (see Figure 1, right). In our experiments, we work with text-based similarities (see Section 3.2) as well as audio-based similarities (see Section 4.2). While in our motivational example we manually overlay the different SSMs, to find that some structural elements are only unveiled by the melody - and not by the text - in our neural architecture, we overlay different SSMs by stacking them into a single time-aligned tensor with c channels, as described in the following.

There are two common patterns that were investigated in the literature: diagonals and rectangles. Diagonals parallel to the main diagonal indicate sequences that repeat and are typically found in a chorus. Rectangles, on the other hand, indicate sequences in which all the lines are highly similar to one another. Both of these patterns were found to be indicators of segment borders.

2.2 Convolutional Neural Network-based Model

Lyrics segments manifest themselves in the form of distinct patterns in the SSM. In order to detect these patterns efficiently, we introduce the Convolutional Neural Network (CNN) architecture which is illustrated in Figure 2. The model predicts for each lyrics line if it is segment ending. For each of the n lines of a song text the model receives patches (see Figure 2, step A) extracted from SSMs $\in \mathbb{R}^{n \times n}$ and centered around the line: $\text{input}_i = \{P_i^1, P_i^2, \dots, P_i^c\} \in \mathbb{R}^{2w \times n \times c}$, where c is the number of SSMs or number of channels and w is the window size. To ensure the model captures the segment-indicating patterns regardless of their location and relative size, the input patches go through two convolutional layers (see Figure 2, step B) (Goodfellow *et al.*, 2016), using filter sizes of $(w+1) \times (w+1)$ and $1 \times w$, respectively. By applying max pooling after both convolutions each feature is downsampled to a scalar. After

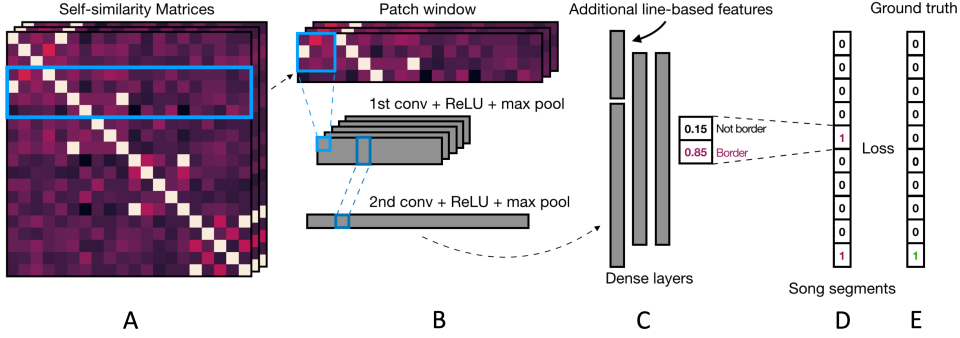


Fig. 2: Convolutional Neural Network-based model inferring lyrics segmentation.

the convolutions, the resulting feature vector is concatenated with the line-based features (see Figure 2, step C) and goes through a series of densely connected layers. Finally, the *softmax* is applied to produce probabilities for each class (border/not border) (see Figure 2, step D). The model is trained with supervision using binary cross-entropy loss between predicted and ground truth segment border labels (see Figure 2, step E). Note that while the patch extraction is a local process, the SSM representation captures global relationships, namely the similarity of a line to all other lines in the lyrics.

2.3 Bimodal Lyrics Lines

To perform lyrics segmentation on a bimodal text-audio representation of a song to benefit from both textual and audio features, we use a corpus where the annotated lyrics ground truth (segment borders) is synchronized with the audio. This bimodal dataset is described in Section 4.1. We focus solely on the audio extracts that have singing voice, as only they are associated to the lyrics. For that let t_i be the time interval of the (singing event of) text line l_i in our synchronized text-audio corpus. Then, a bimodal lyrics line $a_i = (l_i, s_i)$ consists of both a text line l_i (the text line during t_i) and its associated audio segment s_i (the audio segment during t_i). As a result, we have the same number of text lines and audio segments. While the text lines l_i can be used directly to produce SSMs, the complexity of the raw audio signal prevents it from being used as direct input of our system. Instead, it is common to extract features from the audio that highlight some aspects of the signal that are correlated with the different musical dimensions. Therefore, we describe each audio segment s_i as set of different time vectors. Each frame of a vector contains information of a precise and small time interval. The size of each audio frame depends on the configuration of each audio feature. Specifically, we use a sample rate of 22kHz to extract from each time frame two sets of features using `librosa.feature` (McFee *et al.*, 2015). We call an audio segment s_i *featurized* by a feature f if f is applied to all frames of s_i . For our bimodal segment representation we featurize each s_i with one of the following features:

- **Mel-frequency cepstral coefficients** ($mfcc \in \mathbb{R}^{14}$): these coefficients

(Davis & Mermelstein, 1980) emphasize parts of the signal that are related with our understanding of the musical timbre. The mfcc describe the overall shape of a spectral envelope of a signal as a set of features. We extract 15 coefficients and discard the first component as it only conveys a constant offset.

- **Chroma feature ($chr \in \mathbb{R}^{12}$):** this feature (Fujishima, 1999) describes the harmonic information of each frame by computing the “presence” of the twelve different notes. We compute a 12-element feature vector where each feature corresponds with each pitch class in western music where one octave is divided into 12 equal-tempered pitches.

3 Experiments with Unimodal Text-based Representations

This section describes our lyrics segmentation experiments using text lines as input. First, we describe the datasets used (Section 3.1). We then define the different similarity measures used to construct the self-similarity matrices (see Section 3.2). In Section 3.3 we describe the models and configurations that we have investigated. Finally, we present and discuss the obtained results (Section 3.4).

3.1 Datasets: MLDB and WASABI

Song texts are available widely across the Web in the form of user-generated content. Unfortunately for research purposes, there is no comprehensive publicly available online resource that would allow a more standardized evaluation of research results. This is mostly attributable to copyright limitations and has been criticized before in (Mayer & Rauber, 2011). Research therefore is usually undertaken on corpora that were created using standard web-crawling techniques by the respective researchers. Due to the user-generated nature of song texts on the Web, such crawled data is potentially noisy and heterogeneous, e.g. the way in which line repetitions are annotated can range from verbatim duplication to something like *Chorus (4x)* to indicate repeating the chorus four times.

In the following we describe the lyrics corpora we used in our experiments. First, MLDB and WASABI are purely textual corpora. Complementarily, DALI is a corpus that contains bimodal lyrics representations in which text and audio are synchronized.

The Music Lyrics Database (MLDB) V.1.2.7³ is a proprietary lyrics corpus of popular songs of diverse genres. We use this corpus in the same configuration as used before by the state of the art in order to facilitate a comparison with their work. Consequently, we only consider English song texts that have five or more segments and we use the same training, development and test indices, which is a 60%-20%-20% split. In total we have 103k song texts with at least 5 segments. 92% of the remaining song texts count between 6 and 12 segments.

³ <http://www.odditysoftware.com/page-datasales1.htm>

The WASABI corpus⁴ (Meseguer-Brocal *et al.*, 2017), is a larger corpus of song texts, consisting of 744k English song texts with at least 5 segments, and for each song it provides the following information: its lyrics⁵, the synchronized lyrics when available⁶, DBpedia abstracts and categories the song belongs to, genre, label, writer, release date, awards, producers, artist and/or band members, the stereo audio track from Deezer, when available, the unmixed audio tracks of the song, its ISRC, bpm, and duration.

3.2 Similarity Measures

In the following, we define the text-based similarities used to compute the SSMs. Given the text lines of the lyrics, we compute three line-based text similarity measures, based on either their characters, their phonetics or their syntax.

- **String similarity (sim_{str}):** a normalized Levenshtein string edit similarity between the characters of two lines of text (Levenshtein, 1966). This has been widely used - e.g. (Watanabe *et al.*, 2016; Fell *et al.*, 2018).
- **Phonetic similarity (sim_{phon}):** a simplified phonetic representation of the lines computed using the “Double Metaphone Search Algorithm” (Philips, 2000). When applied to “i love you very much” and “i’l off you vary match” it returns the same result: “ALFFRMX”. This algorithm was developed to capture the similarity of similar sounding words even with possibly very dissimilar orthography. We translate the text lines into this “phonetic language” and then compute sim_{str} between them.
- **Lexico-syntactical similarity (sim_{lsyn}):** this measure was initially proposed in (Fell, 2014) to capture both the lexical similarity between text lines as well as the syntactical similarity. Consider the two text lines “Look into my eyes” and “I look into your eyes”: there is a similarity on the lexical level, as similar words are used. Also, the lines are similar on the syntactical level, as they share a similar word order. We estimate the lexical similarity sim_{lex} between two lines as their relative word bigram overlap, and in analogy, we estimate their syntactical similarity sim_{syn} via relative POS tag bigram overlap. Finally, we define the lexico-syntactical similarity sim_{lsyn} as a weighted sum of sim_{lex} and sim_{syn} . The details of the computation are described in the Appendix - Section B.

3.3 Models and Configurations

We represent song texts via text lines and experiment on the MLDB and WASABI datasets. We compare to the state of the art (Watanabe *et al.*, 2016) and successfully reproduce their best features to validate their approach. Two groups of features are

⁴ <https://wasabi.i3s.unice.fr/>

⁵ Extracted from <http://lyrics.wikia.com/>

⁶ From <http://usdb.animux.de>

used in the replication: repeated pattern features (RPF) extracted from SSMs and n-grams extracted from text lines. The RPF basically act as hand-crafted image filters that aim to detect the edges and the insides of diagonals and rectangles in the SSM.

Then, our own models are neural networks as described in Section 2.2, that use as features SSMs and two line-based features: the line length and n-grams. For the line length, we extracted the character count from each line, a simple proxy of the orthographic shape of the song text. Intuitively, segments that belong together tend to have similar shapes. Similarly to (Watanabe *et al.*, 2016)’s term features we extracted those n-grams from each line that are most indicative for segment borders: using the tf-idf weighting scheme, we extracted n-grams that are typically found left or right from the segment border, varied n-gram lengths and also included indicative part-of-speech tag n-grams. This resulted in 240 term features in total. The most indicative words at the start of a segment were: {ok, lately, okay, yo, excuse, dear, well, hey}. As segment-initial phrases we found: {Been a long, I’ve been, There’s a, Won’t you, Na na na, Hey, hey}. Typical words ending a segment were: {..., ..., !, ., yeah, ohh, woah. c’mom, wonderland}. And as segment-final phrases we found as most indicative: {yeah!, come on!, love you., !!!, to you., with you., check it out, at all., let’s go, ...}

In this experiment we consider only SSMs made from text-based similarities; we note this in the model name as CNN_{text} . We further name a CNN model by the set of SSMs that it uses as features. For example, the model $\text{CNN}_{\text{text}}\{\text{str}\}$ uses as only feature the SSM made from string similarity sim_{str} , while the model $\text{CNN}_{\text{text}}\{\text{str}, \text{phon}, \text{lsyn}\}$ uses three SSMs in parallel (as different input channels), one from each similarity.

For convolutional layers we empirically set $w_{\text{size}} = 2$ and the amount of features extracted after each convolution to 128. Dense layers have 512 hidden units. We have also tuned the learning rate (negative degrees of 10), the dropout probability with increments of 0.1. The batch size was selected from the beginning to be 256 to better saturate our GPU. The CNN models were implemented using Tensorflow.

For comparison, we implement two baselines. The random baseline guesses for each line independently if it is a segment border (with a probability of 50%) or not. The line length baseline uses as only feature the line length in characters and is trained using a logistic regression classifier.

In order to favor comparative analysis, the first experiments are run against the MLDB data set (see Section 3.1) used by the state-of-the-art method (Watanabe *et al.*, 2016). To test the system portability to bigger and more heterogeneous data sources, we further experimented our method on the WASABI corpus (see Section 3.1). In order to test the influence of genre on classification performance, we aligned MLDB to WASABI as the latter provides genre information. Song texts that had the exact same title and artist names (ignoring case) in both data sets were aligned. This rather strict filter resulted in an amount of 58567 (57%) song texts with genre information in MLDB. Table 2 shows the distribution of the genres in MLDB song texts. We then tested our method on each genre separately, to test our hypothesis that classification is harder for some genres in which almost no repeated

patterns can be detected (as Rap songs). To the best of our knowledge, previous work did not report on genre-specific results.

In this work we did not normalize the lyrics in order to rigorously compare our results to (Watanabe *et al.*, 2016). We estimate the proportion of lyrics containing words that indicate the text structure (*Chorus*, *Intro*, *Refrain*, ...), to be marginal (0.1-0.5%) in the MLDB corpus. When applying our methods for lyrics segmentation to lyrics found online, an appropriate normalization method should be applied as a pre-processing step. For details on such a normalization procedure we refer the reader to (Fell, 2014), Section 2.1.

Evaluation metrics are Precision (P), Recall (R), and f-score (F_1). Significance is tested with a permutation test (Ojala & Garriga, 2010), and the p -value is reported.

3.4 Results and Discussion

Table 1 shows the results of our experiments with text lines on the MLDB dataset. We start by measuring the performance of our replication of (Watanabe *et al.*, 2016)’s approach. This reimplementation exhibits 56.3% F_1 , similar to the results reported in the original paper (57.7%). The divergence could be attributed to a different choice of hyperparameters and feature extraction code. Much weaker baselines were explored as well. The random baseline resulted in 18.6% F_1 , while the usage of simple line-based features, such as the line length (character count), improves this to 25.4%.

The best CNN-based model, $\text{CNN}_{\text{text}}\{\text{str}, \text{phon}, \text{lsyn}\} + \text{n-grams}$, outperforms all our baselines reaching 67.4% F_1 , 8.2pp better than the results reported in (Watanabe *et al.*, 2016). We perform a permutation test (Ojala & Garriga, 2010) of this model against all other models. In every case, the performance difference is statistically significant ($p < .05$).

Subsequent feature analysis revealed that the model $\text{CNN}_{\text{text}}\{\text{str}\}$ is by far the most effective. The $\text{CNN}_{\text{text}}\{\text{lsyn}\}$ model exhibits much lower performance, despite using a much more complex feature. We believe the lexico-syntactical similarity is much noisier as it relies on n-grams and PoS tags, and thus propagates error from the tokenizers and PoS taggers. The $\text{CNN}_{\text{text}}\{\text{phon}\}$ exhibits a small but measurable performance decrease from $\text{CNN}_{\text{text}}\{\text{str}\}$, possibly due to phonetic features capturing similar regularities, while also depending on the quality of preprocessing tools and the rule-based phonetic algorithm being relevant for our song-based dataset. The $\text{CNN}_{\text{text}}\{\text{str}, \text{phon}, \text{lsyn}\}$ model that combines the different textual SSMs yields a performance comparable to $\text{CNN}_{\text{text}}\{\text{str}\}$.

In addition, we test the performance of several line-based features on our dataset. Most notably, the n-grams feature provides a significant performance improvement producing the best model. Note that adding the line length feature to any CNN_{text} model does not increase performance.

To show the portability of our method to bigger and more heterogeneous datasets, we ran the CNN model on the WASABI dataset (as described in Section 3.1), obtaining results that are very close to the ones obtained for the MLDB dataset: pre-

<i>Model</i>	<i>Features</i>	<i>P</i>	<i>R</i>	<i>F₁</i>
Random baseline	n/a	18.6	18.6	18.6
Line length baseline	text line length	16.7	52.8	25.4
Handcrafted filters	RPF (our replication)	48.2	67.8	56.3
	RPF (Watanabe <i>et al.</i> , 2016)	56.1	59.4	57.7
	RPF + n-grams	57.4	61.2	59.2
CNN _{text}	{str}	70.4	63.0	66.5
	{phon}	75.9	55.6	64.2
	{lsyn}	74.8	50.0	59.9
	{str, phon, lsyn}	74.1	60.5	66.6
	{str, phon, lsyn} + n-grams	72.1	63.3	67.4

Table 1: Results with text lines on MLDB dataset in terms of Precision (P), Recall (R) and F_1 in %.

cision: 67.4% for precision, 67.3% recall, and 67.4% f-score using the CNN_{text}{str} model.

Results differ significantly based on genre. We split the MLDB dataset with genre annotations into training and test, trained on all genres, and tested on each genre separately. In Table 2 we report the performances of the CNN_{text}{str} on lyrics of different genres. Songs belonging to genres such as Country, Rock or Pop, contain recurrent structures with repeating patterns, which are more easily detectable by the CNN_{text} algorithm. Therefore, they show significantly better performance. On the other hand, the performance on genres such as Hip Hop or Rap, is much worse.

4 Experiments with Multimodal Text-audio Representations

This section describes our lyrics segmentation experiments using multimodal (unimodal or bimodal) lyrics lines, containing text or audio information or both as input. We follow the same structure as in our experiments with text lines, describing the dataset used (see Section 4.1), defining the similarity measures used to construct the self-similarity matrices (see Section 4.2), describing the models and configurations that we have investigated (see Section 4.3), and finally, presenting and discussing the obtained results (see Section 4.4).

<i>Genre</i>	<i>Lyrics[#]</i>	<i>P</i>	<i>R</i>	<i>F₁</i>
Rock	6011	73.8	57.7	64.8
Hip Hop	5493	71.7	43.6	<u>54.2</u>
Pop	4764	73.1	61.5	66.6
RnB	4565	71.8	60.3	65.6
Alternative Rock	4325	76.8	60.9	67.9
Country	3780	74.5	66.4	70.2
Hard Rock	2286	76.2	61.4	67.7
Pop Rock	2150	73.3	59.6	65.8
Indie Rock	1568	80.6	55.5	65.6
Heavy Metal	1364	79.1	52.1	63.0
Southern Hip Hop	940	73.6	34.8	<u>47.0</u>
Punk Rock	939	80.7	63.2	70.9
Alternative Metal	872	77.3	61.3	68.5
Pop Punk	739	77.3	68.7	72.7
Soul	603	70.9	57.0	63.0
Gangsta Rap	435	73.6	35.2	<u>47.7</u>

Table 2: Results with text lines. $\text{CNN}_{\text{text}}\{\text{str}\}$ model performances across musical genres in the MLDB dataset in terms of Precision (P), Recall (R) and F_1 in %. Underlined are the performances on genres with less repetitive text. Genres with highly repetitive structure are in bold.

4.1 Dataset: DALI

The DALI corpus⁷ (Meseguer-Brocal *et al.*, 2018) contains synchronized lyrics-audio representations on different levels of granularity: syllables, words, lines and segments. Depending on the song, the alignment quality between text segments and audio segments is higher or lower. In the Appendix (see Section A), we explain how we estimate this segment alignment quality $Qual$.

Then, in order to test the impact of $Qual$ on the performance of our lyrics segmentation algorithm, we partition the DALI corpus into parts with different $Qual$. Initially, DALI consists of 5358 lyrics that are synchronized to their audio track. Like in previous publications (Watanabe *et al.*, 2016; Fell *et al.*, 2018), we ensure that all song texts contain at least 5 segments. This constraint reduces the number of tracks used by us to 4784. We partition the 4784 tracks based on their $Qual$ into high (Q^+), med (Q^0), and low (Q^-) alignment quality datasets. Table 3 gives an overview over the resulting dataset partitions. The Q^+ dataset consists of 50842 lines and 7985 segment borders and has the following language distribution: 72% English, 11% German, 4% French, 3% Spanish, 3% Dutch, 7% other languages.

⁷ <https://github.com/gabolsgabs/DALI>

Corpus name	Alignment quality	Song count
Q^+	high (90-100%)	1048
Q^0	med (52-90%)	1868
Q^-	low (0-52%)	1868
full dataset	-	4784

Table 3: The DALI dataset partitioned by alignment quality

4.2 Similarity Measures

In this experiment, we add to the common choice of text-based similarity measures also audio-based similarities - the crucial ingredient that makes our approach multimodal. In the following, we define the text-based and audio-based similarities that we use to compute the SSMs.

Text similarity: For our model, we produce SSMs based on the String similarity measure as introduced in Section 3.2. The measure is applied on the textual component l_i of the multimodal lines a_i .

Audio similarities: We have previously defined the process of extracting audio features, as well as the concrete audio features (see Section 2.3). When extracting features from audio segments of different lengths, we obtain feature vectors of different lengths. There are several alternatives to measure the similarity between two audio sequences (e.g. mfcc sequences) of possibly different lengths, among which Dynamic Time Warping T_d is the most popular one in the Music Information Retrieval community. Given bimodal lyrics lines a_u, a_v , we compare two audio segments s_u and s_v that are featurized by a particular audio feature (mfcc or chroma) using T_d :

$$T_d(i, j) = d(s_u(i), s_v(j)) + \min \left\{ \begin{array}{l} T_d(i-1, j), \\ T_d(i-1, j-1), \\ T_d(i, j-1) \end{array} \right\}$$

T_d must be parametrized by an inner distance d to measure the distance between the frame i of s_u and the frame j of s_v . Depending on the particular audio feature s_u and s_v are featurized with, we employ a different inner distance as defined below. Let m be the length of the vector s_u and n be the length of s_v . Then, we compute the minimal distance between the two audio sequences as $T_d(m, n)$ and normalize this by the length r of the shortest alignment path between s_u and s_v to obtain values in $[0, 1]$ that are comparable to each other. We finally apply $\lambda x \cdot (1 - x)$ to turn the distance T_d into a similarity measure S_d :

$$S_d(s_u, s_v) = 1 - T_d(m, n) \cdot r^{-1}$$

Given bimodal lyrics lines a_i , we now define similarity measures between audio

segments s_i that are featurized by a particular audio feature presented previously (mfcc, chr) based on our similarity measure S_d :

- **MFCC similarity (sim_{mfcc}):** S_d between two audio segments featurized by the mfcc feature. As inner distance we use the cosine distance: $d(x, y) = x \cdot y \cdot (\|x\| \cdot \|y\|)^{-1}$
- **Chroma similarity (sim_{chr}):** S_d between two audio segments featurized by the chroma feature. As inner distance we use the cosine distance.

4.3 Models and Configurations

Here, the song texts are represented via bimodal lyrics lines, incorporating both text and audio information, and experimentation is performed on the DALI corpus. In order to test our hypotheses which text and audio features are most relevant to detect segment boundaries, and whether the text and audio modalities complement each other, we compare different types of models: baselines, text-based models, audio-based models, and finally bimodal models that use both text and audio features. We provide the following baselines: the random baseline guesses for each line independently if it is a segment border (with a probability of 50%) or not. The line length baselines use as feature only the line length in characters (text-based model) or milliseconds (audio-based model) or both, respectively. These baselines are trained using a logistic regression classifier.

Finally, the last baseline models the segmentation task as sequence tagging by tagging each text line as segment-ending or not ending. This model uses an RNN and the lyrics line is here modelled as the average word vector of all words in the line. The RNN uses GRU cells with 50 hidden states and 300 dimensional word vectors (Pennington *et al.*, 2014).

All other models are CNNs using the architecture described previously and use as features SSMs made from different textual or audio similarities as described in Section 4.2. The CNN-based models that use purely textual features (str) are named CNN_{text} , while the CNN-based models using purely audio features (mfcc, chr) are named $\text{CNN}_{\text{audio}}$. Lastly, the CNN_{mult} models are multimodal in the sense that they use combinations of textual and audio features. We name a CNN model by its modality (text, audio, mult) as well as by the set of SSMs that it uses as features. For example, the model $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}\}$ uses as textual feature the SSM made from string similarity sim_{str} and as audio feature the SSM made from mfcc similarity sim_{mfcc} .

As dataset we use the Q^+ part of the DALI dataset (see Section 4.1). We split the data randomly into training and test sets using the following scheme: considering that the DALI dataset is relatively small, we average over two different 5-fold cross-validations. We prefer this sampling strategy for our small dataset over a more common 10-fold cross-validation as it avoids the test set becoming too small.

4.4 Results and Discussion

The results of our experiments with multimodal lyrics lines on the DALI dataset are depicted in Table 4. The random baseline and the different line length baselines reach a performance of 15.5%-33.5% F_1 . Interestingly, the audio-based line length (33.5% F_1) is more indicative of the lyrics segmentation than the text-based line length (25.0% F_1).⁸ Finally, the word-based RNN sequence tagger performs better (41.6% F_1) than the simple baselines, but is vastly inferior to the CNN-based models. Given this finding, we did not try the sequence tagger with additional audio features.

The model $\text{CNN}_{\text{text}}\{\text{str}\}$ performs with 70.8% F_1 similarly to the $\text{CNN}_{\text{text}}\{\text{str}\}$ model from the first experiment (66.5% F_1). The models use the exact same SSM_{str} feature and hyperparameters, but another lyrics corpus (DALI instead of MLDB). We believe that as DALI was assembled from karaoke singing instances, it likely contains more repetitive song texts that are easier to segment using the employed method. Note that the DALI dataset is too small to allow a genre-wise comparison as we did in the previous experiment using the MLDB dataset.

The $\text{CNN}_{\text{audio}}$ models perform similarly well than the CNN_{text} models. $\text{CNN}_{\text{audio}}\{\text{mfcc}\}$ reaches 65.3% F_1 , while $\text{CNN}_{\text{audio}}\{\text{chr}\}$ results in 63.9% F_1 . The model $\text{CNN}_{\text{audio}}\{\text{mfcc}, \text{chr}\}$ performs with 70.4% F_1 significantly ($p < .001$) better than the models that use only one of the features. As the mfcc feature models timbre and instrumentation, whilst the chroma feature models melody and harmony, they provide complementary information to the $\text{CNN}_{\text{audio}}$ model which increases its performance.

Most importantly, the CNN_{mult} models combining text- with audio-based features constantly outperform the CNN_{text} and $\text{CNN}_{\text{audio}}$ models. $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}\}$ and $\text{CNN}_{\text{mult}}\{\text{str}, \text{chr}\}$ achieve a performance of 73.8% F_1 and 74.5% F_1 , respectively - this is significantly ($p < .001$) higher compared to the 70.8% (70.4%) F_1 of the best CNN_{text} ($\text{CNN}_{\text{audio}}$) model. Finally, the overall best performing model is a combination of the best CNN_{text} and $\text{CNN}_{\text{audio}}$ models and delivers 75.3% F_1 . $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}, \text{chr}\}$ is the only model to significantly ($p < .05$) outperform all other models in all three evaluation metrics: precision, recall, and F_1 . Note, that all CNN_{mult} models outperform all CNN_{text} and $\text{CNN}_{\text{audio}}$ models significantly ($p < .001$) in recall.

We perform an ablation test on the alignment quality. For this, we train CNN-based models with those feature sets that performed best on the Q^+ part of DALI. For each modality (text, audio, mult), i.e. $\text{CNN}_{\text{text}}\{\text{str}\}$, $\text{CNN}_{\text{audio}}\{\text{mfcc}, \text{chr}\}$, and $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}, \text{chr}\}$, we train a model for each feature set on each partition of DALI (Q^+ , Q^0 , Q^-). We always test our models on the same alignment quality they were trained on. The alignment quality ablation results are depicted in Table 5. We find that independent of the modality (text, audio, mult.), all models perform significantly ($p < .001$) better with higher alignment quality. The effect of

⁸ Note that adding line length features to any CNN-based model does not increase performance.

<i>Model</i>	<i>Features</i>	<i>P</i>	<i>R</i>	<i>F₁</i>
Random baseline	n/a	15.7	15.7	15.7
Line length baselines	text length	16.6	51.8	25.0
	audio length	22.7	63.8	33.5
	text length + audio length	22.6	63.0	33.2
RNN sequence tagging	word vectors	39.9	43.6	41.6
CNN _{text}	{str}	78.7	64.2	70.8
CNN _{audio}	{mfcc}	79.3	55.9	65.3
	{chr}	76.8	54.7	63.9
	{mfcc, chr}	79.2	63.8	70.4
CNN _{mult}	{str, mfcc}	80.6	69.0	73.8
	{str, chr}	82.5	69.0	74.5
	{str, mfcc, chr}	82.7	70.3	75.3

Table 4: Results with multimodal lyrics lines on the Q^+ dataset in terms of Precision (P), Recall (R) and F_1 in %. Note that the CNN_{text}{str} model is the same configuration as in Table 2, but trained on different dataset.

modality on segmentation performance (F_1) is as follows: on all datasets we find CNN_{mult}{str, mfcc, chr} to significantly ($p < .001$) outperform both CNN_{text}{str} and CNN_{audio}{mfcc, chr}. Further, CNN_{text}{str} significantly ($p < .001$) outperforms CNN_{audio}{mfcc, chr} on the Q^0 and Q^- dataset, whereas this does not hold on the Q^+ dataset ($p \geq .05$).

5 Error Analysis

An SSM for a Rap song is depicted in Figure 3. As texts in this genre are less repetitive, the SSM-based features are less reliable to determine a song’s structure. Moreover, when returning to the introductory example in Figure 1, we observe that verses (the V_i) and bridges (the B_i) are not detectable when looking at the text representation only (see Figure 1, middle). The reason is that these verses have different lyrics. However, as these parts share the same melody, highlighted rectangles are visible in the melody structure.

Indeed, we found our bimodal segmentation model to produce significantly ($p < .001$) better segmentations (75.3% F_1) compared to the purely text-based (70.8% F_1) and audio-based models (70.4% F_1). The increase in F_1 stems from both increased precision and recall. The model increase in precision is observed as

<i>Dataset</i>	<i>Model</i>	<i>Features</i>	<i>P</i>	<i>R</i>	<i>F₁</i>
Q^+	CNN_{text}	{str}	78.7	64.2	70.8
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	79.2	63.8	70.4
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	82.7	70.3	75.3
Q^0	CNN_{text}	{str}	73.6	54.5	62.8
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	74.9	48.9	59.5
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	75.8	59.4	66.5
Q^-	CNN_{text}	{str}	67.5	30.9	41.9
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	66.1	24.7	36.1
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	68.0	35.8	46.7

Table 5: Results with multimodal lyrics lines for the alignment quality ablation test on the datasets Q^+ , Q^0 , Q^- in terms of Precision (P), Recall (R) and F_1 in %.

CNN_{mult} often produces less false positive segment borders, i.e. the model delivers less noisy results. We observe an increase in recall in two ways: first, CNN_{mult} sometimes detects a combination of the borders detected by CNN_{text} and $\text{CNN}_{\text{audio}}$. Secondly, there are cases where CNN_{mult} detects borders that are not recalled in either of CNN_{text} or $\text{CNN}_{\text{audio}}$.

Segmentation algorithms that are based on exploiting patterns in an SSM, share a common limitation: non-repeated segments are hard to detect as they do not show up in the SSM. Note, that such segments are still occasionally detected indirectly when they are surrounded by repeated segments. Furthermore, a consecutively repeated pattern such as C_2 - C_3 - C_4 in Figure 1 is not easily segmentable as it could potentially also form one ($C_2C_3C_4$) or two (C_2 - C_3C_4 or C_2C_3 - C_4) segments. Another problem is that of inconsistent classification inside of a song: sometimes, patterns in the SSM that look the same to the human eye are classified differently. Note, however that on the pixel level there is a difference, as the inference in the used CNN is deterministic. This is a phenomenon similar to adversarial examples in image classification (same intension, but different extension).

We now analyze the predictions of our different models for the example song given in Figure 1. We compare the predictions of the following three different models: the text-based model $\text{CNN}_{\text{text}}\{\text{str}\}$ (visualized in Figure 1 as the left SSM called “repetitive lyrics structure”), the audio-based model $\text{CNN}_{\text{audio}}\{\text{chr}\}$ (visualized in Figure 1 as the right SSM called “repetitive melody structure”), and the bimodal model $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}, \text{chr}\}$. Starting with the first chorus, C_1 , we find it to be segmented correctly by both $\text{CNN}_{\text{text}}\{\text{str}\}$ and $\text{CNN}_{\text{audio}}\{\text{chr}\}$. As previously discussed, consecutively repeated patterns are hard to segment and our text-based model indeed fails to correctly segment the repeated chorus (C_2 - C_3 - C_4). The audio-based model $\text{CNN}_{\text{audio}}\{\text{chr}\}$ overcomes this limitation and segments the repeated

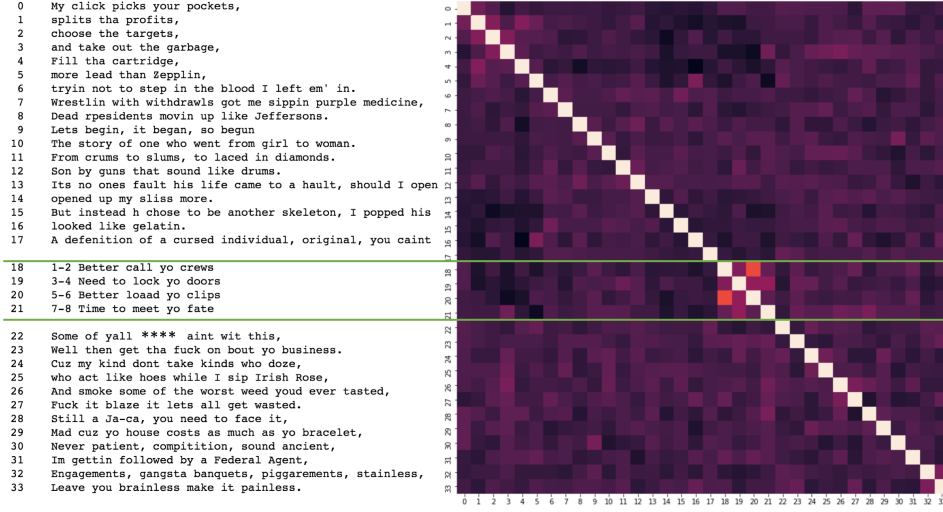


Fig. 3: Example SSM computed from textual similarity sim_{str} . As common for Rap song texts, there is no chorus (diagonal stripe parallel to main diagonal). However, there is a highly repetitive musical state from line 18 to 21 indicated by the corresponding rectangle in the SSM spanning from (18,18) to (21,21). (“Meet Your Fate” by Southpark Mexican, MLDB-ID: 125521)

chorus correctly. Finally, we find that in this example both the text-based and the audio-based models fail to segment the verses (the V_i) and bridges (the B_i) correctly. The $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}, \text{chr}\}$ model manages to detect the bridges and verses in our example.

Note that adding more features to a model does not always increase its ability to detect segment borders. While in some examples, the $\text{CNN}_{\text{mult}}\{\text{str}, \text{mfcc}, \text{chr}\}$ model detects segment borders that were not detected in any of the models $\text{CNN}_{\text{text}}\{\text{str}\}$ or $\text{CNN}_{\text{audio}}\{\text{mfcc}, \text{chr}\}$, there are also examples where the bimodal model does not detect a border that is detected by both the text-based and the audio-based models.

6 Related Work

Besides the work of (Watanabe *et al.*, 2016) that we have discussed in detail in Section 2, only a few papers in the literature have focused on the automated detection of the structure of lyrics. (Mahedero *et al.*, 2005) report experiments on the use of standard NLP tools for the analysis of music lyrics. Among the tasks they address, for structure extraction they focus on lyrics having a clearly recognizable structure (which is not always the case) divided into segments. Such segments are weighted following the results given by descriptors used (as full length text, relative position of a segment in the song, segment similarity), and then tagged with a label describing them (e.g. chorus, verses). They test the segmentation algorithm on a

small dataset of 30 lyrics, 6 for each language (English, French, German, Spanish and Italian), which had previously been manually segmented.

More recently, (Baratè *et al.*, 2013) describe a semantics-driven approach to the automatic segmentation of song lyrics, and mainly focus on pop/rock music. Their goal is not to label a set of lines in a given way (e.g. verse, chorus), but rather identifying recurrent as well as non-recurrent groups of lines. They propose a rule-based method to estimate such structure labels of segmented lyrics, while in our approach we apply machine learning methods to unsegmented lyrics.

(Cheng *et al.*, 2009) propose a new method for enhancing the accuracy of audio segmentation. They derive the semantic structure of songs by lyrics processing to improve the structure labeling of the estimated audio segments. With the goal of identifying repeated musical parts in music audio signals to estimate music structure boundaries (lyrics are not considered), (Cohen-Hadria & Peeters, 2017) propose to feed Convolutional Neural Networks with the square-sub-matrices centered on the main diagonals of several SSMs, each one representing a different audio descriptor, building their work on (Foote, 2000).

For a different task than ours, (Mihalcea & Strapparava, 2012) use a corpus of 100 lyrics synchronized to an audio representation with information on musical key and note progression to detect emotion. Their classification results using both modalities, textual and audio features, are significantly improved compared to a single modality.

7 Conclusion

In this article, we have addressed the task of lyrics segmentation on synchronized text-audio representations of songs. For the songs in the corpus DALI where the lyrics are aligned to the audio, we have derived a measure of alignment quality specific to our task of lyrics segmentation. Then, we have shown that exploiting both textual and audio-based features lead the employed Convolutional Neural Network-based model to significantly outperform the state-of-the-art system for lyrics segmentation that relies on purely text-based features. Moreover, we have shown that the advantage of a bimodal segment representation pertains even in the case where the alignment is noisy. This indicates that a lyrics segmentation model can be improved in most situations by enriching the segment representation by another modality (such as audio).

As for future work, the problem of inconsistent classification inside of a song (SSM patterns look almost identically, but classifications differ) may be tackled by clustering the SSM patterns in such a way that very similar looking SSM patterns end up in the same cluster. This can be seen as a preprocessing denoising step of the SSMs where details that are irrelevant to our task are deleted, without losing relevant information. Furthermore, the problem that the bimodal model sometimes fails to detect a segment border, even if the submodels correctly detected that border may be tackled by implementing a late fusion approach (Snoek *et al.*, 2005) where the prediction of the bimodal model is conditioned on the predictions of both the text-based and the audio-based submodels. In alternative to our CNN-based approach,

other neural architectures such as RNNs and Transformers (Vaswani *et al.*, 2017) can be applied to the lyrics segmentation problem. While our initial experiments in framing the lyrics segmentation task as sequence tagging (see Section 4.3) did not yield results competitive to our CNNs, we believe that experimentation with more recent sentence embeddings, such as those derived from pretrained language models (Devlin *et al.*, 2018), can be beneficial. Finally, we would like to experiment with further modalities, for instance with subtitled music videos where text, audio, and video are all synchronized to each other.

A Measuring the Segment Alignment Quality in DALI

The DALI corpus (Meseguer-Brocal *et al.*, 2018) contains synchronized lyrics-audio representations on different levels of granularity: syllables, words, lines and segments. It was created by joining two datasets: (1) a corpus for karaoke singing (AMX) which contains alignments between lyrics and audio on the syllable level and (2) a subset of WASABI lyrics that belong to the same songs as the lyrics in AMX. Note that corresponding lyrics in WASABI can differ from those in AMX to some extent. Also, in AMX there is no annotation of segments. DALI provides estimated segments for AMX lyrics, projected from the ground truth segments from WASABI. For example, Figure 4 shows on the left side the lyrics lines as given in AMX. The right side shows the lyrics lines given in WASABI as well as the ground truth lyrics segments. The left side shows the estimated lyrics segments in AMX. Note how the lyrics in WASABI have one segment more, as the segment W_3 has no counterpart in AMX.

Based on the requirements for our task, we derive a measure to assess how well the estimated AMX segments correspond / align to the groundtruth WASABI segments. Since we will use the WASABI segments as ground truth labels for supervised learning, we need to make sure, the AMX lines (and hence audio information) actually belongs to the aligned segment. As only for the AMX lyrics segments we have aligned audio features and we want to consistently use audio features in our segment representations, we make sure that every AMX segment has a counterpart WASABI segment (see Figure 4, $A_0 \sim W_0$, $A_1 \sim W_1$, $A_2 \sim W_2$, $A_3 \sim W_4$). On the other hand, we allow WASABI segments to have no corresponding AMX segments (see Figure 4, W_3). We further do not impose constraints on the order of appearance of segments in AMX segmentations vs. WASABI segmentations, to allow for possible rearrangements in the order of corresponding segments. With these considerations, we formulate a measure of alignment quality that is tailored to our task of bimodal lyrics segmentation. Let A, W be segmentations, where $A = A_0 A_1 \dots A_n$ and the A_i are AMX segments and $W = W_0 W_1 \dots W_m$ with WASABI lyrics segments W_i . Then the alignment quality between the segmentations A, W is composed from the similarities of the best-matching segments. Using string similarity sim_{str} as defined in Section 3.2, we define the alignment quality $Qual$ as follows:

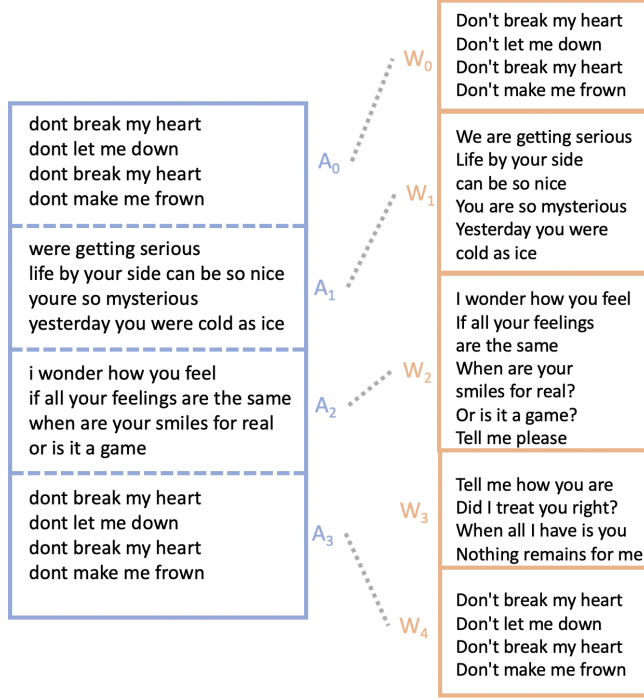


Fig. 4: Lyrics lines and estimated lyrics segments in AMX (left). Lyrics lines and ground truth lyrics segments in WASABI (right) for the song (“Don’t Break My Heart” by Den Harrow)

$$\begin{aligned} Qual(A, W) &= Qual(A_0 A_1 \dots A_n, W_0 W_1 \dots W_m) \\ &= \min_{0 \leq i \leq n} \{ \max_{0 \leq j \leq m} \{ \text{sim}_{\text{str}}(A_i, W_j) \} \} \end{aligned}$$

B Lexico-syntactical similarity

Formally, given two text lines x, y , let $\text{bigrams}(x)$ be the set of bigrams in line x . Following (Fell, 2014) lexical similarity sim_{lex} between lines x, y is then defined as:

$$\text{sim}_{\text{lex}}(x, y) = \frac{|\text{bigrams}(x) \cap \text{bigrams}(y)|}{\max\{|\text{bigrams}(x)|, |\text{bigrams}(y)|\}}$$

To define the syntactical similarity sim_{syn} , we apply a POS tagger to those word bigrams that do not overlap. Formally, the non-overlapped bigrams are $\hat{x} = \text{bigrams}(x) \setminus (\text{bigrams}(x) \cap \text{bigrams}(y))$ and $\hat{y} = \text{bigrams}(y) \setminus (\text{bigrams}(x) \cap \text{bigrams}(y))$. We then apply element-wise a function *postag* to the non-overlapped

bigrams in \hat{x}, \hat{y} to obtain POS tagged bigrams. Syntactical similarity sim_{syn} is thus given by:

$$\text{sim}_{\text{syn}}(x, y) = \left(\frac{|\text{postag}(\hat{x}) \cap \text{postag}(\hat{y})|}{\max\{|\text{postag}(\hat{x})|, |\text{postag}(\hat{y})|\}} \right)^2$$

Note that the whole term is squared to heuristically account for the simple fact that there are usually many more words than POS tags and so syntactical similarities are inherently larger than lexical ones since the overlap is normalized by a smaller number of overall POS tags in consideration.

We define sim_{lsyn} as a weighted sum of sim_{lex} and sim_{syn} :

$$\text{sim}_{\text{lsyn}}(x, y) = \alpha_{\text{lex}} \cdot \text{sim}_{\text{lex}}(x, y) + (1 - \alpha_{\text{lex}}) \cdot \text{sim}_{\text{syn}}(x, y)$$

We heuristically set $\alpha_{\text{lex}} = \text{sim}_{\text{lex}}$. The idea for this weighting is that when x and y have similar wordings, they likely have high similarity, so the wording should be more important if it is more similar. On the other hand, if the wording is more dissimilar, the structural similarity should be more important for figuring out a *lexico-syntactical* similarity between two lines. Hence, sim_{lsyn} can be written as:

$$\text{sim}_{\text{lsyn}}(x, y) = \text{sim}_{\text{lex}}^2(x, y) + (1 - \text{sim}_{\text{lex}}) \cdot \text{sim}_{\text{syn}}(x, y)$$

We close with an example to illustrate the computation of sim_{lsyn} :⁹

x = “The man sleeps deeply.”

$\Rightarrow \text{bigrams}(x) = \{\text{the man, } \mathbf{man\ sleep}, \text{sleep deep}\}$

y = “A man slept.”

$\Rightarrow \text{bigrams}(y) = \{\text{a man, } \mathbf{man\ sleep}\}$

$\Rightarrow \text{sim}_{\text{lex}}(x, y) = \frac{1}{3}$

$\hat{x} = \{\text{the man, sleep deep}\}$

$\hat{y} = \{\text{a man}\}$

$\text{postag}(\hat{x}) = \{\mathbf{DET\ NOUN}, \text{VERB\ ADVERB}\}$

$\text{postag}(\hat{y}) = \{\mathbf{DET\ NOUN}\}$

$\Rightarrow \text{sim}_{\text{syn}}(x, y) = \left(\frac{1}{2}\right)^2 = \frac{1}{4}$

$\Rightarrow \text{sim}_{\text{lsyn}}(x, y) = \left(\frac{1}{3}\right)^2 + \left(1 - \frac{1}{3}\right) \cdot \frac{1}{4} = \frac{1}{9} + \frac{1}{6} \approx 0.28$

⁹ We assume stemming in this example.

References

- Baratè, A., Ludovico, L. A., & Santucci, E. 2013 (Dec). A Semantics-Driven Approach to Lyrics Segmentation. *Pages 73–79 of: 2013 8th International Workshop on Semantic and Social Media Adaptation and Personalization.*
- Brackett, D. 1995. *Interpreting Popular Music.* Cambridge University Press.
- Cheng, H. T., Yang, Y. H., Lin, Y. C., & Chen, H. H. 2009 (May). Multimodal structure segmentation and analysis of music using audio and textual information. *Pages 1677–1680 of: 2009 IEEE International Symposium on Circuits and Systems.*
- Cohen-Hadria, Alice, & Peeters, Geoffroy. 2017 (June). Music Structure Boundaries Estimation Using Multiple Self-Similarity Matrices as Input Depth of Convolutional Neural Networks. *In: AES International Conference Semantic Audio 2017.*
- Davis, Steven B., & Mermelstein, Paul. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *ACOUSTICS, SPEECH AND SIGNAL PROCESSING, IEEE TRANSACTIONS ON*, 357–366.
- Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, & Toutanova, Kristina. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*
- Fell, Michael. 2014. *Lyrics classification.* M.Phil. thesis, Saarland University, Germany.
- Fell, Michael, Nechaev, Yaroslav, Cabrio, Elena, & Gandon, Fabien. 2018. Lyrics Segmentation: Textual Macrostructure Detection using Convolutions. *Pages 2044–2054 of: Proceedings of the 27th International Conference on Computational Linguistics.*
- Foote, Jonathan. 2000. Automatic audio segmentation using a measure of audio novelty. *Pages 452–455 of: Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, vol. 1. IEEE.
- Fujishima, Takuya. 1999. Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music. *In: ICMC.* Michigan Publishing.
- Goodfellow, Ian, Bengio, Yoshua, & Courville, Aaron. 2016. *Deep Learning.* MIT Press. <http://www.deeplearningbook.org>.
- Levenshtein, Vladimir I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Pages 707–710 of: Soviet physics doklady*, vol. 10.
- Mahedero, Jose P. G., Martínez, Álvaro, Cano, Pedro, Koppenberger, Markus, & Gouyon, Fabien. 2005. Natural Language Processing of Lyrics. *Pages 475–478 of: Proceedings of the 13th Annual ACM International Conference on Multimedia.* MULTIMEDIA '05. New York, NY, USA: ACM.
- Mayer, Rudolf, & Rauber, Andreas. 2011. Musical genre classification by ensembles of audio and lyrics features. *Pages 675–680 of: Proceedings of the 12th International Conference on Music Information Retrieval.*
- McFee, Brian, Raffel, Colin, Liang, Dawen, Ellis, Daniel PW, McVicar, Matt, Battenberg, Eric, & Nieto, Oriol. 2015. librosa: Audio and music signal analysis in python. *Pages 18–25 of: Proceedings of the 14th python in science conference*, vol. 8.
- Meseguer-Brocal, Gabriel, Peeters, Geoffroy, Pellerin, Guillaume, Buffa, Michel, Cabrio, Elena, Faron Zucker, Catherine, Giboin, Alain, Mirbel, Isabelle, Hennequin, Romain, Moussallam, Manuel, Piccoli, Francesco, & Fillon, Thomas. 2017 (Aug.). WASABI: a Two Million Song Database Project with Audio and Cultural Metadata plus WebAudio enhanced Client Applications. *In: Web Audio Conference 2017 – Collaborative Audio #WAC2017.* Queen Mary University of London, London, United Kingdom.
- Meseguer-Brocal, Gabriel, Cohen-Hadria, Alice, & Peeters, Geoffroy. 2018. DALI: a large Dataset of synchronized Audio, Lyrics and notes, automatically created using teacher-student machine learning paradigm. *In: ISMIR Paris, France.*
- Mihalcea, Rada, & Strapparava, Carlo. 2012. Lyrics, music, and emotions. *Pages 590–599 of: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.* Association for Computational Linguistics.

- Ojala, Markus, & Garriga, Gemma C. 2010. Permutation Tests for Studying Classifier Performance. *J. Mach. Learn. Res.*, **11**(Aug.), 1833–1863.
- Pennington, Jeffrey, Socher, Richard, & Manning, Christopher. 2014. Glove: Global vectors for word representation. *Pages 295–313 of: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.
- Philips, Lawrence. 2000. The Double Metaphone Search Algorithm. *C/C++ Users Journal*, **18**(06), 38–43.
- Snoek, Cees G. M., Worring, Marcel, & Smeulders, Arnold W. M. 2005. Early Versus Late Fusion in Semantic Video Analysis. *Pages 399–402 of: Proceedings of the 13th Annual ACM International Conference on Multimedia*. MULTIMEDIA '05. New York, NY, USA: ACM.
- Tagg, Philip. 1982. Analysing popular music: theory, method and practice. *Popular Music*, **2**, 37–67.
- Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, & Polosukhin, Illia. 2017. Attention is All you Need. *Pages 5998–6008 of: Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., & Garnett, R. (eds), Advances in Neural Information Processing Systems 30*. Curran Associates, Inc.
- Watanabe, Kento, Matsubayashi, Yuichiroh, Orita, Naho, Okazaki, Naoaki, Inui, Kentaro, Fukayama, Satoru, Nakano, Tomoyasu, Smith, Jordan, & Goto, Masataka. 2016. Modeling Discourse Segments in Lyrics Using Repeated Patterns. *Pages 1959–1969 of: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*.