Music Boundary Detection using Convolutional Neural Networks: A Comparative Analysis of Combined Input Features

Carlos Hernandez-Olivan, Jose R. Beltran, David Diaz-Guerra *

Department of Electronic Engineering and Communications, University of Zaragoza, 50018, Zaragoza (Spain)

Received 26 October 2020 | Accepted 9 September 2021 | Published 25 October 2021

Abstract

The analysis of the structure of musical pieces is a task that remains a challenge for Artificial Intelligence, especially in the field of Deep Learning. It requires prior identification of the structural boundaries of the music pieces, whose structural boundary analysis has recently been studied with unsupervised methods and supervised neural networks trained with human annotations. The supervised neural networks that have been used in previous studies are Convolutional Neural Networks (CNN) that use Mel-Scaled Log-magnitude Spectograms features (MLS), Self-Similarity Matrices (SSM) or Self-Similarity Lag Matrices (SSLM) as inputs. In previously published studies, pre-processing is done in different ways using different distance metrics, and different audio features are used for computing the inputs, so a generalised pre-processing method for calculating model inputs is missing. The objective of this work is to establish a general method to pre-process these inputs by comparing the results obtained by taking the inputs calculated from different pooling strategies, distance metrics and audio characteristics, also taking into account the computing time to obtain them. We also establish the most effective combination of inputs to be delivered to the CNN to provide the most efficient way to extract the boundaries of the structure of the music pieces. With an adequate combination of input matrices and pooling strategies, we obtain an accuracy F_1 of 0.411 that outperforms a current work done under the same conditions (same public available dataset for training and testing).

Keywords

Deep Learning, CNNs, Music Analysis, Music Information Retrieval (MIR), Music Structure, Self-Similarity Matrix (SSM).

DOI: 10.9781/ijimai.2021.10.005

I. Introduction

Music Information Retrieval (MIR[1]) is the interdisciplinary science for retrieving information from music. MIR is a field of research that faces different tasks in automatic music analysis, such as pitch tracking, chord estimation, score alignment or music structure detection. One of the most active communities and references in MIR is the Music Information Retrieval Evaluation eXchange (MIREX[2]). This is the community that every year holds the International Society for Music Information Retrieval Conference (ISMIR). Algorithms are submitted to be tested in MIREX's datasets within the different MIR tasks. Most of the previous results analyzed and compared in this work have been presented in different MIREX campaigns.

The automatic structural analysis or Music Structure Analysis (MSA) of music is a very complex challenge that has been studied in recent years [1], but it has not yet been solved with an adequate accuracy that surpasses the analysis performed by musicians or specialists. This kind of analysis is only a part of the musical analysis, which involves musical aspects like harmony, timbre and tempo, and segmentation principles like repetition, homogeneity and novelty [2]. This automatic music analysis can be faced starting from music representations such as the score of the piece, the MIDI file of the piece, or the raw audio file.

In music, *form* refers to the structure of a musical piece, which consists of dividing the musical pieces into small units, starting with the motifs, then the phrases, and finally the sections that express a musical idea. *Boundary detection* is the first step that has to be done in musical form analysis and must be done before the naming of the different segments depending on the similarity between them. This last step is named *Labelling* or *Clustering*. This task, translated to the most common genre in MIREX datasets, the pop genre, would be the detection and extraction of the chorus, verse, or introduction of the corresponding song. Detecting the boundaries of music pieces consists on identifying the transitions where these parts begin and end, a task that professional musicians do almost automatically by listening a piece of music. This detection of the boundaries in a musical piece is based on the *Audio Onset Detection* task, which is the first step for several higher-level music analysis tasks such as beat detection, tempo estimation, and transcription.

[1] https://musicinformationretrieval.com/index.html

[2] https://www.music-ir.org/mirex/wiki/MIREX_HOME

* Corresponding author.

E-mail addresses: carloshero@unizar.es (Carlos Hernandez-Olivan), jrbelbla@unizar.es (Jose R. Beltran), ddga@unizar.es (David Diaz-Guerra).

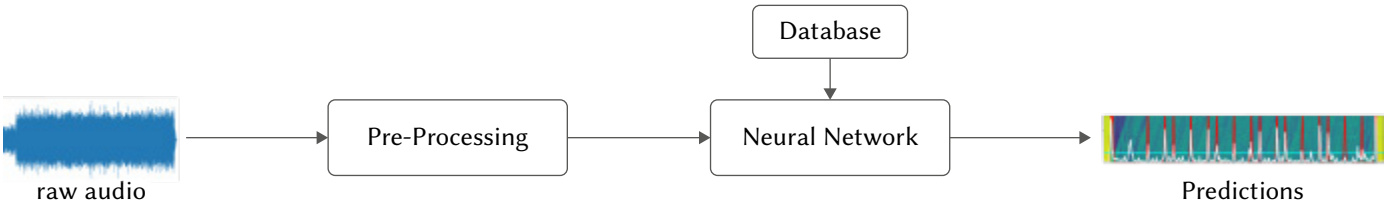


Fig. 1. General scheme of supervised neural networks.

This problem can be accomplished with different techniques that have in common the need of pre-processing the audio files in order to extract the desired audio features and then apply unsupervised or supervised methods. There are several studies where this pre-processing step is made in different ways, so there is not yet a generalized input pre-processing method. The currently *end-to-end* best-performing methods use CNNs trained with human annotations. The inputs to the CNN are MelScaled Log-magnitude Spectograms (MLSs) [3], Self-similarity Lag-Matrices (SSLMs) in combination with the MLSs [4], and also combining these matrices with chromas [5].

One of the limitations of these methods is that the analysis and results obtained depend largely on the database annotator since there can be inconsistencies between different annotators when analyzing the same piece. Therefore, these methods are limited to the quality of the labels given by the annotators and they cannot outperform them.

This paper deals with the issue of structure detection in music pieces. In particular, we study the comparison of different methods of boundary detection between the musical sections by means of Convolutional Neural Networks. The paper is structured as follows: Section II presents an overview of the related work and previous studies in which this work is based on. The Self-Similarity Matrices and the used datasets are also presented. In Section III, the pre-processing method of the matrices that will be used as inputs of the neural network (NN) is explained. Section IV introduces the database used for training, validating and testing, and the labelling process. Section V shows the NN structure and the thresholding and peak-picking strategies and section VI describes the metrics used to test the model and exposes the results of the experiments and their comparison with previous studies. Finally, section VII presents the discussion and section VIII discusses proposals for future lines of work. All code used in this paper, including the pre-trained models of every case of study in this work, is made publicly available[3] and further results are shown in the website[4].

II. Related Work

Several studies have been done in the field of structure recognition in music since Foote introduced the self-similarity matrix (SSM) in 1999 [6] and later, in 2003, he derived from it the selfsimilarity lag matrix (SSLM) [7]. Before the introduction of the SSMs and SSLMs, the studies were based on processing spectrograms [8], but in recent years it has been demonstrated that SSMs and SSLMs calculated from audio features in combination with spectrograms provide better results. We describe some previous works of both unsupervised and supervised methods which belongs to the MIREX's task: Music Structure Segmentation.

A. Unsupervised Methods

The main idea of most of the unsupervised methods is to extract the musical structure of the music pieces but not necessarily the boundaries between the structure sections.

According to Paulus et al. [9], these methods can be summarized in three approaches based on: novelty, homogeneity and repetition. These approaches are computed with unsupervised Machine Learning algorithms such as genetic algorithms (*fitness functions*), Hidden Markov Models (HMM), *K-means*, Linear Discriminant Analysis (NDA), Decision Stump or Checkerboard-like kernels.

The **Novelty-based** approach consists on the detection of the transitions between contrasting parts [1]. This approach is well-performed using checkerboard-like kernel methods which were introduced by Foote in 2000 [10]. These methods have evolved during the years and it has been found that multipletemporal-scale kernels, as those of Kaiser and Peeters in 2013 [11], outperformed the results of previous works by proposing a fusion of the novelty and repetition approaches.

The **Homogeneity-based** approach is based on the identification of sections that are consistent with respect to their musical properties [1]. These methods use Hidden Markov Models, like Logan and Chu [12], Aucouturier and Sandler [13] and Levy and Schandler [14] or combinations of SSMs like Traile and McFee [15], and McFee and Bello [16].

The **Repetition-based** approach refers to finding recurring patterns. These methods apply a clustering algorithm to the SSMs or SSLMs. They are more applicable for labeling the structural parts of music pieces rather than precise segmentation which is required by boundary detection. Lu et al. in 2004 [17], Paulus and Klapuri in 2006 [18], Turnbull et al. [19], McFee and Ellis [20], and McCallum [21] are examples of this method.

To conclude, we can affirm that unsupervised algorithms are very efficient performing the labelling (clustering) part, but not the boundaries detection task, which is better performed by supervised neural networks which came up in 2014 and are described in section B.

B. Supervised Neural Networks

Supervised neural networks learn from input representations given the ground truth, which are the label annotations of the targets (Fig. 1).

Previous studies of boundary detection used Mel-Scaled Log-magnitude Spectograms (MLS) as the inputs of CNNs [3]. This method was based on *Audio Onset Detection* task [22], which consists on finding the starting points of every musically relevant event in an audio signal, specifically the beginning of a music note. This task can be interpreted as a computer vision problem, like edge detection, but applied to spectrograms instead of images with different textures.

Later on, in 2015, Grill and Schlüter improved their previous work by adding SSLMs, which yielded to better results [4], and the addition of SSLMs with different lag factors to the input of the CNN [5], outperforming this method and reaching the best result to date.

In Tables I and II we show a recap of the results of almost all of the previous works that have been done in boundary detection using both unsupervised and supervised neural networks. Results and algorithms nomenclature in Table I have been extracted from MIREX's campaigns of different years. It must be said that the results obtained

[3] https://github.com/carlosholivan/MusicBoundariesCNN

[4] https://carlosholivan.github.io/publications/2021-boundaries/2021-boundaries.html

TABLE I. Results of Boundary Detection of Previous Studies for "Full Structure" and "Segmentation" Tasks. Only the Best-performing Algorithm in Terms of F-measure of Each Year for A 0.5s Time-window Tolerance Is Shown. The F-measure Is Shown for Different Databases (See Sec.D)

Year[5]	Autors [Ref.]	Algorithm	Input	Method	F-measure (F_1) for Testing Databases			
					MIREX09	RCW-A	RCW-B	SALAMI
Unsupervised Methods								
2009	Paulus & Klapuri [24]	PK	MFCCs, chromas	*Fitness function*	0.27	-	-	-
2010	Mauch et al. [25]	MND1	MFCCs, Discrete Cepstrum	*HMM*	0.325	0.359	-	-
2011	Sargent et al. [26]	SB-VRS1	Chords estimation	*Viterbi*	0.231	0.324	-	-
2012	Kaiser et al. [27]	KSP2	SSM	*Novelty measure*	0.280	0.366	0.289	0.286
2013	McFee & Ellis [20]	MP2	MLS	*Fisher's Linear Discriminant*	0.281	0.355	0.278	0.317
2014	Nieto & Bello [28]	NB1	MFCCs + chromas	*Checkerboard-like kernel*	0.289	0.352	0.269	0.299
2015	Cannam et al. [29]	CC1	Timbre-type histograms	*HMM*	0.197	0.224	0.203	0.213
2016	Nieto [30]	ON2	Constant-Q Transform Spectrogram	*Linear Discriminant Analysis*	0.259	0.381	0.255	0.299
2017	Cannam et al. [29]	CC1	Timbre-type histograms	*HMM*	0.201	0.228	0.192	0.212
Supervised Neural Networks								
2014	Schlüter et al. [31]	SUG1	MLS	*CNN*	0.434	0.546	0.438	0.529
2015	Grill & Schlüter [32]	GS1	MLS + SSLMs	*CNN*	0.523	0.697	0.506	0.541

TABLE II. Results of Previous Works in Boundary Detection Task for 0.5s Time-window Tolerance. It Is Only Showed the Best F-measure Result of Each Reference for Each Database

Year	Autors [Ref.]	Input	Method	Train Set	F-measure (F_1) for Testing Databases			
					MIREX09	RCW-A	RCW-B	SALAMI
Unsupervised Methods								
2007	Turnbull et al. [19]	MFCCs, chromas, spectrogram	Boosted Decision Stump	-	-	-	0.378	-
2011	Sargent et al. [34]	MFCCs, chromas	Viterbi	-	-	-	0.356	-
Supervised Neural Networks								
2014	Ullrich et. al [22]	MLS	CNN	*Private*	-	-	-	0.465
2015	Grill & Schlüter [4]	MLS + SSLMs	CNN	*Private*	-	-	-	0.523
2015	Grill & Schlüter [5]	MLS + PCPs + SSLMs	CNN	*Private*	-	-	-	0.508
2017	Hadria & Peeters [35]	MLS + SSLMs	CNN	*SALAMI*	-	-	-	0.291

with unsupervised methods on Table I are not as high as the results obtained with supervised neural networks because, as it has been mentioned in section A, the main goal of the unsupervised methods is not the boundary detection (segmentation) itself but the full structure identification (labelling).

C. Self-Similarity Matrices (SSMs)

The Self-Similarity Matrix [2] is a tool not only used in music structure analysis but also in time series analysis tasks. In these matrices, the different parts of the structure of a music piece can be identified as homogeneous regions. This representation of the structural elements of music analysis leads this matrix and its combination with spectrograms to be the input of almost every model described in sections A and B. For this work, this matrix is important because music is in itself *self-similar*, in other words, it is formed by similar time series.

Self-Similarity Matrices have been used under the name of Recurrence Plot for the analysis of dynamic systems [23], but their introduction to the music domain was done by Foote [6] in 1999 and since then, there have appeared different techniques for computing these matrices. The SSM relies on the concept of self-similarity, which is measured by a similarity function that is applied to the audio features representation. The similarity between two feature vectors *yn* and *ym* is a function that can be expressed as Eq. 1 shows. The result is a *N*-square matrix SSM $\in \mathbb{R}^{N\times N}$ being *N* the time dimension:

$$\mathrm{SSM}(n,m) = \delta(y_n, y_m) \tag{1}$$

where $n, m \in [1, ..., N]$.

The similarity function is obtained by the calculation of a distance between the two feature vectors *y* mentioned before. In the literature, this distance is usually calculated as the Euclidean distance δ_{eucl} or the cosine distance δ_{cos}:

$$\delta_{\mathrm{eucl}}(y_n, y_m) = \|y_n - y_m\| \tag{2}$$

$$\delta_{\cos}(y_n, y_m) = 1 - \frac{u.v}{\|y_n\|.\|y_m\|} \tag{3}$$

where *u* and *v* are time series vectors.

Self-Similarity Matrices can be computed from different audio features representations, such as MFCCs or chromas, and they can also be obtained by combining different frame-level audio features [15]. Once the similarity function has been computed for each pair of audio

[5] https://www.music-ir.org/mirex/wiki/<<year>>:MIREX<<year>>_Results - headland "Music Structure Segmentation Results".

feature vectors and the SSM has been calculated, we can filter the SSM by applying thresholding techniques, smoothing or invariance transposition. The SSM can also be obtained with other techniques such as clustering methods as Serra et al. proposed [33], where the SSM is obtained by applying the *k-nn* algorithm.

After Foote in 1999 defined the SSM, in 2003, Goto [7] defined a variant of the SSM which is known as the Self-Similarity Lag Matrix (SSLM). The SSLM is a matrix that represents the similarities between low-level features of one point in time and points in the past, up to a certain *lag time*. This representation makes possible to plot the relations between past events and their repetitions in the future. Some approaches calculate this SSLM after computing the SSM or the recurrence plot as we show in Eq. 4:

$$\mathrm{SSLM}(i,j) = \mathrm{SSM}_{k+1,j} \tag{4}$$

with $i = 1, ..., N$, $j = 1, ..., L$ and $k = i + j - 2\,modulus(N)$

The dimensions of this matrix are not $N \times N$ as the SSM, but they are $N \times L$, being L the *lag time factor*. That means that the SSLM is a non-square matrix: $\mathrm{SSLM} \in \mathbb{R}^{N \times L}$.

The choice of the type of audio features representation for computing the SSMs or SSLMs, and the choice of using SSMs or SSLMs is one of the most important steps when solving a MIR task and has to be studied depending on the issue we we want to face.

D. Datasets

Previous works had been tested in the annual Music Information Retrieval Evaluation eXchange (MIREX [36]), which is a framework for evaluating music information retrieval algorithms.

The first dataset of the MIREX campaign for the structure segmentation task was the MIREX09 dataset, consisting on a collection of The Beatles' songs plus another smaller dataset[6]. Beatles dataset have 2 annotation versions, one is Paulus Beatles or Beatles-TUT[7] dataset and the second one is the Isophonic Beatles or Beatles-ISO[8] dataset. The second MIREX dataset was MIREX10, formed by the RWC [37] dataset. This dataset has 2 annotation versions; RWC-A[9] of QUAERO project which is the one which corresponds to MIREX10 and RWC-B[10] [38], which is the original annotated version following the annotation guidelines established by Bimbot el al. [39].

A few years later, the MIREX12 dataset provided a greater variety of songs than the MIREX10 [40]. MIREX12 is a dataset formed by the "Structural Analysis of Large Amounts of Music Information" (SALAMI[11]) dataset which has evolved in its more recent version, the SALAMI 2.0 database. The analysis of MIREX structure segmentation task was published in 2012 [41]. Our work uses the publicly available SALAMI 2.0 dataset.

III. Audio Processing

This work is based on the previous works of Schuler, Grill et al. [3], [4] who propose a pre-proscessing method to obtain the SSLMs from MFCCs features. We will extend these works by calculating the SSLMs from chroma features and applying also the Euclidean distance that has not been considered in preliminary studies, to compute the SSLMs in order to give a comparison and find the best-performing input to the NN model.

[6] http://ifs.tuwien.ac.at/mir/audiosegmentation.html

[7] http://www.cs.tut.fi/sgn/arg/paulus/beatles_sections_TUT.zip

[8] http://isophonics.net/content/reference-annotations

[9] http://musicdata.gforge.inria.fr

[10] http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation

[11] https://ddmal.music.mcgill.ca/research/SALAMI/

A. Mel Spectrogram

The first step of the pre-processing part is to extract the audio features. To do that, we first compute the the Short-Time-FourierTransform (STFT) with a Hanning window of 46ms (2048 samples at 44.1kHz sample rate) and an overlap of 50% as Grill et al. proposed [4]. Then, we obtain a mel-scaled filterbank of 80 triangular filters from 80Hz to 16kHz and we scale logarithmically the amplitude magnitudes to obtain the mel-spectrogram (MLS). We used the *librosa* library [42] to compute the mel-spectrogram. After obtaining the MLS, we apply a max-pooling of $p = 6$ in the temporal dimension to give the Neural Network a manageable size input. The size of the MLS matrix is $P \times N$ with P being the number of frequency bins (that are equal to the number of triangular filters) and N the number of time frames. We define $\mathbf{x}_i$ with $i = 1 \ldots N$ as the i-th frame of the MLS.

B. Self-Similarity Lag Matrix From MFCCs

The method that we used to generate the SSLMs[12] is the same method that Grill and Schluter used in [4] and [5], which in turn derives from Serra et al. [43].

The first step after computing each frame mel-spectrogram $\mathbf{x}_i$ is to pad a vector Φ with noise of -70dB with a duration of L frames at the beginning of the mel-spectrogram.

$$\check{\mathbf{x}}_i = \Phi \| \mathbf{x}_i \tag{5}$$

where Φ is a matrix of size $L \times P$ whose elements are equal to -70dB.

Then, a max-pool of a factor of p_1 is done in the time dimension as shown in Eq. 6.

$$\mathbf{x}'_i = \max_{j=1 \ldots p_1} (\check{\mathbf{x}}_{(i-1)p_1+j}) \tag{6}$$

After that, we apply a Discrete Cosine Transform of Type II to each frame omitting the first element.

$$\widetilde{\mathbf{X}}_i = \mathrm{DCT}^{(\mathrm{II})}(\mathbf{x}'_i)_{[2 \ldots P]} \tag{7}$$

where P are the number of mel-bands.

Now we stack the time frames by a factor m so we obtain the time series in Eq. 8. The resulting $\hat{\mathbf{X}}_i$ vector has dimensions $[(P-1) \cdot \mathrm{m}] \times [(N+L)/p_1]$ where N is the number of time frames before the max-pooling and L the lag factor in frames.

$$\hat{\mathbf{X}}_i = [\widetilde{\mathbf{X}}_i^{\mathrm{T}} \| \widetilde{\mathbf{X}}_{i+m}^{\mathrm{T}}]^{\mathrm{T}} \tag{8}$$

The final SSLM matrix is obtained by calculating a distance between the vectors $\hat{\mathbf{X}}_i$. In our work, we use two different distance metrics: the Euclidean distance and the cosine distance. This will allow us to make a comparison between them and conclude which SSLM performs better.

Therefore, the distance between two vectors $\hat{\mathbf{X}}_i$ and $\hat{\mathbf{X}}_{i-l}$ using the distance metric δ is

$$D_{i,l} = \delta(\hat{\mathbf{X}}_i, \hat{\mathbf{X}}_{i-l}), \quad l = 1 \ldots \left\lfloor \frac{L}{p_1} \right\rfloor \tag{9}$$

where δ is the distance metric as defined in Eqs. 2 and 3.

Then, we compute an equalization factor $\varepsilon_{i,l}$ with a quantile κ of the distances $\delta(\hat{\mathbf{X}}_i, \hat{\mathbf{X}}_{i-j})$ for $j = 1 \ldots \left\lfloor \frac{L}{p} \right\rfloor$

$$\varepsilon_{i,l} = Q_\kappa \left(D_{i,l}, \cdots, D_{i,\left\lfloor \frac{L}{p} \right\rfloor} \| D_{i-l,1}, \cdots, D_{i-l,\left\lfloor \frac{L}{p} \right\rfloor} \right) \tag{10}$$

We now remove the first L/p lag bins in the time dimension of the distances matrix D and in the equalization factor matrix ε, and we apply Eq. 6 with max-pooling factor p_2. Finally we obtain the SSLM applying Eq. 11.

[12] https://github.com/carlosholivan/SelfSimilarityMatrices

$$R_{i,l} = \sigma\left(1 - \frac{D_{i,l}}{\varepsilon_{i,l}}\right) \tag{11}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$.

Once the SSLM has been obtained, we need to pad some noise to the begin and end of the SSLM because the labels which are used to train our model will be given to the NN as Gaussians (see section IV), so the first and last labels need information in their left and right sides respectively. We add the noise to the begin and end of the SSLM and MLS by padding them with γ = 50 time frames of pink noise at the beginning and end of the MLS matrix. Then we then normalized each frequency band to zero mean and unit variance for MLS and each lag band for the SSLMs. Note also that if there are some time frames that have exactly the same values, the cosine distance would give a NAN (not-a-number) value. We avoid this by converting all this NAN values into zero as the last step of the SSLM computation.

C. Self-Similarity Lag Matrix From Chromas

The process of computing the SSLM from chroma features is similar to the method explained in section B. The difference here is that instead of starting with padding the mel-spectrogram in Eq. 5, we pad the STFT. After applying the max-pooling in Eq. 6, we compute the chroma filters instead of computing the DCT in Eq. 7. The rest of the process is the same as described in section B.

All the values of the parameters used to obtaining the SelfSimilarity Matrices are summarized in Table III. In addition to the Euclidean and cosine metrics, and MFCCs and chromas audio features, we will compare two pooling strategies. The first one is to make a max-pooling of factor $p_1 = 6$ to the STFT (from MLS calculation), and to the Chromas or MFCCs for the SSLMs computation, as it is described in Eq. 6. The other pooling strategy is the one showed in Fig. 2, where we first do a pooling of $p_1 = 2$ and then a pooling of $p_2 = 3$ once the SSLMs are obtained. We denote these pooling variants as 6pool and 2pool3 respectively. The total time for processing all the SSLMs (MFCCs and cosine distance) was a factor or 4 faster for 6pool than 2pool3 because by applying a higher padding factor in Eq. 6 the size of the matrices D and ε is much lower so the calculation of these matrices take more time but it also implies a resolution loss that can affect the accuracy of the model as [4] remarks.

TABLE III. Parameter Final Values

Parameter	Symbol	Value	Units
sampling rate	sr	44100	Hz
window size	w	46	ms
overlap	-	50	%
hop length	h	23	ms
lag	L	14	s
pooling factor 6pool	p	6	-
`2pool3`	p_1	2	-
	p_2	3	-
stacking parameter	m	2	-
quantile	κ	0.1	-
final padding	γ	50	frames

The general schema of the pre-processing block is depicted in Fig. 2.

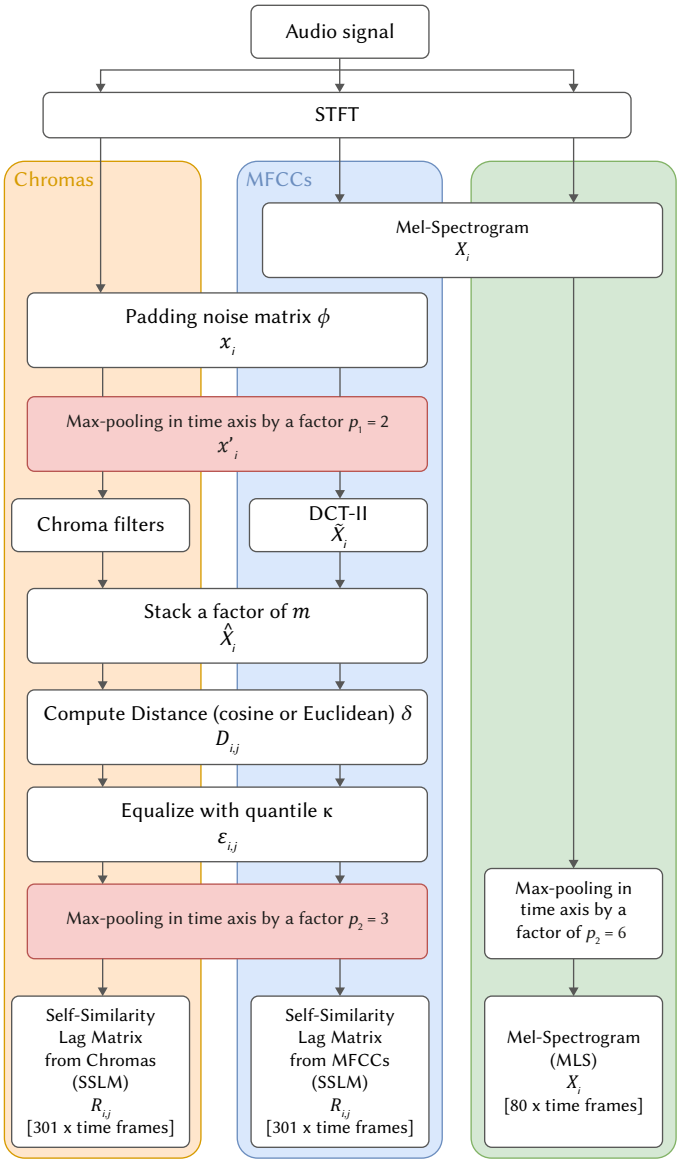


Fig. 2. General block diagram of the pre-processing block in Fig. Each background color contains the steps that are necessary to compute each of the inputs: MLS (green), SSLM from Chromas (orange) and SSLM from MFCCs (blue). The red background in the max-pooling blocks refers to the 2 variants done in this work: `2pool3` is the one showed in the scheme, while `6pool` is computed by applying the max-pooling of factor 6 in the first red block and removing the second red block of the scheme.

IV. Dataset

The algorithm was trained, validated and tested on a subset of the Structural Analysis of Large Amounts of Music Information (SALAMI) dataset [44]. SALAMI dataset contains 1048 double annotated pieces from which we could obtain 1006 pieces since the datasest does not provide the audio files due to copyright restrictions. For the training of the model, we used the text files of labels from annotator 1 and for the songs that were not annotated by annotator 1, we use the same text file but from annotator 2.

It is important to highlight that, as described in [35], previous works such as [3], [4] and [5] use a private non-accessible dataset of 733 songs from which 633 pieces were used for training and 100 for validation. Therefore, we re-implemented the work presented in [4] but we trained it in our dataset composed by only public SALAMI pieces and annotations. We split our 1006 SALAMI audio tracks into

65%, 15% and 20%, resulting in 650, 150 and 206 pieces for training, validation and testing respectively.

A. Labelling Process

As explained in [3], it is necessary to transform the labels of the SALAMI text files into Gaussian functions so that the Neural Network can be trained correctly. We first set the center values of the Gaussian functions by transforming the labels in seconds into time frames as showed in Eq. 12 constructing the vector $y1$ which contains the center of the gaussians and has its dimension equal to the number of labels in the text file. In Eq. 12, label_i are the labels in seconds extracted from SALAMI text file "functions" and p_1, p_2, h, sr and γ are defined in Table III.

$$y_i' = \frac{\text{label}_i}{p_1 \cdot p_2} + \frac{h \cdot sr}{\gamma} \tag{12}$$

Then, we apply a gaussian function with standard deviation $\sigma = 0.1$ and μ_i equal to each label value in Eq.12. In Eq.13 we show the expression of the gaussians of the labels.

$$\text{gaussian\_labels}_\text{i} = \mathbf{g}(y_i', \mu_i, \sigma) \tag{13}$$

with

$$\mathbf{g}(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{14}$$

where μ_i is a vector of $\frac{y_i \cdot \gamma + \frac{w}{2}}{sr}$ frame from $i = 1 \ldots \left[\frac{N}{p_1 \cdot p_2}\right]$.

To train the model, we removed the first tag from each text file due to the proximity of the first two tags in almost every file and the uselessness of the Neural Network identifying the beginning of the file. It's also worth mentioning the fact that we have resampled all the songs in the SALAMI database at a single *sampling rate* of 44100Hz as showed in Table III.

V. Model

Our work and current methods that tackle the problem of boundary detection in MSA use neural network-based models that were originally developed for image processing tasks, in particular Convolutional Neural Networks (CNN) [45], [46], [47], [48]. The model developed in this work for boundary detection is shown in Fig. 3. Once the matrices of the pre-processing step are obtained, they are padded and normalized to form the input of a Convolutional Neural Network (CNN). The obtained predictions are post-processed with a peak-picking and threshold algorithm to obtain the final predictions.

A. Convolutional Neural Network

The model proposed in this paper is nearly the same than the model proposed in [3] and [4], so we could compare the results and make a comparison with different input strategies as Cohen [35] did. However, we take into account more inputs combinations and with high and low dimensions in order to see the better inputs combination for the model.

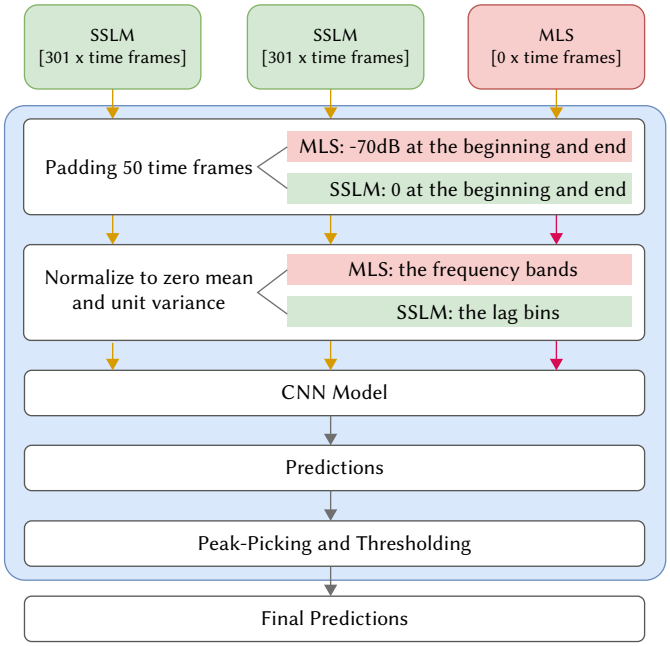


Fig. 3. General block diagram of the Neural Network block in Fig. 1.

The model is composed by a CNN whose relevant parameters are shown in Table IV. The difference between this model and the model proposed in [3] and [4] is that our final two layers are not dense layers but convolutional layers in the time dimension because we do not crop the inputs and get a single probability value at the output, but we give the Neural Network the whole matrix and we obtain a time prediction curve at the output. The general schema of the CNN is shown in Fig. 4.

The parameters of the CNN model have been chosen according to previous literature [4] for a fair comparison in the study of how different input features affect the performance of the MSA task. The changes that have been done from the state-of-the-art model rely on adding the dilation parameter that we use in the layers of our model, and we also changed the last layer of our implementation in comparison with previous literature models. This is because previous studies passed a segment of the SSLM trough the CNN while we pass the entire SSLM to it. The last layer of our implementation outputs one feature map that is passed trough a Sigmoid function which outputs the boundary probability of each time frame of the entire music piece, so the output of the model is a vector of length equal to the time frames of the input. This differs from the literature models where the output is the boundary probability of the segmented part of the input.

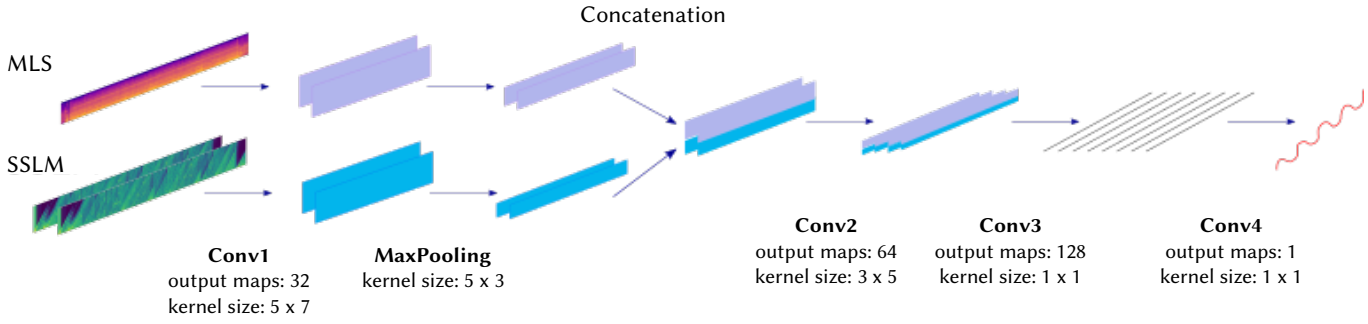


Fig. 4. Schema of the Convolutional Neural Network implemented. The main parameters are presented in Table IV.

TABLE IV. CNN Architecture Parameters of the Schema Presented in Fig. 4

Layer	Parameters
Convolution 1 + Leaky ReLU	output feature maps: 32 kernel size: 5 x 7 stride: 1 x 1 padding: (5-1)/2 x (7-1)/2
Max-Pooling	kernel size: 5 x 3 stride: 5 x 1 padding: 1 x 1
Convolution 2 + Leaky ReLU	output feature maps: 64 kernel size: 3 x 5 stride: 1 x 1 padding: (3-1)/2 x (5-1)*3/2 dilation: 1 x 3
Convolution 3 + Leaky ReLU	output feature maps: 128 kernel size: 1 x 1 stride: 1 x 1 padding: 0 x 0
Convolution 4 + Sigmoid	output feature maps: 1 kernel size: 1 x 1 stride: 1 x 1 padding: 0 x 0

B. Training Parameters

We trained our CNN with *Binary Cross Entropy* or BCEwithLogitsLoss in Pytorch [49] as the loss function which in Pytorch implementation includes a Sigmoid activation function in the last layer of the Neural Network, a *learning rate* of 0.001 and Adam optimizer [50]. We perform *early-stopping* during training to determine the best-performing model. The SSLMs and MLS have to be passed to the GPU one by one because they have different lengths, which means that 1 song is passed forward and backward through the NN at once. However, to get more robust gradients and a more stable optimization process, the optimizer is executed with the average gradients of batchs of 10 songs. We could say that we use a batch size of 1 in terms of GPUs calls but a batch size of 10 in terms of the training. The models were trained on a GTX 980 Ti Nvidia GPU and we used TensorboardX [51] to graph the loss and F-score of training and validation.

C. Peak-Picking

Peak-picking consists on selecting the peaks of the output signal of the CNN that will be identified as boundaries of the different parts of the song. Each boundary on the output signal is considered true when no other boundary is detected within 6 seconds. The application of a threshold helps us to discriminate boundary values that are not higher than an optimum threshold. We calculate the optimum threshold for our experiments by computing the average F_1 in our validation set for all possible threshold values in the range [0, 1] and then we select the highest value. Therefore, the optimum threshold is the value between $[0,1]$ for which the average F_1 is higher in our validation set. It is reasonable to realise that the optimum threshold value may vary when training our model with the different combination of inputs that we show in Table VI. When we train our model with isolated inputs (see Table V) we compute the threshold with the MLS but we do not vary it when testing SSLMs trainings. We vary the threshold value when we train our model with different inputs combinations in order to optimize the each case of study and give the best-performing method (see Table VI). In Fig. 5, we set a thresold of 0.205 for the models using only the MLS as input and for the rest of the models we used the values indicated in Table VI. From the optimum threshold calculation, we can observe that almost all optimum threshold values for each input variant belong to [2:05; 2:6] Fig. 5 shows Recall, Precision and F-score values (see Section A) of the testing dataset evaluated for each possible threshold value.

TABLE V. Results of Boundaries Estimation According to Different Pooling Strategies, Distances and Audio Features for ± 0:5s and a Threshold of 0.205

Tolerance: ± 0:5s and Threshold: 0.205					
	Input	**Epochs**	**P**	**R**	**F1**
6pool	MLS	180	0.501	0.359	0.389
	$SSLM^{MFCCs}_{euclidean}$	180	0.472	0.318	0.361
	$SSLM^{MFCCs}_{cosine}$	180	0.477	0.311	0.355
	$SSLM^{chromas}_{euclidean}$	180	0.560	0.228	0.297
	$SSLM^{chromas}_{cosine}$	180	0.508	0.254	0.312
2pool3	$SSLM^{MFCCs}_{euclidean}$	120	0.422	0.369	0.375
	$SSLM^{MFCCs}_{cosine}$	120	0.418	0.354	0.366
Previous works					
2pool3	MLS	-	0.555	0.458	0.465
	$SSLM^{MFCCs}_{cosine}$	-	-	-	0.430

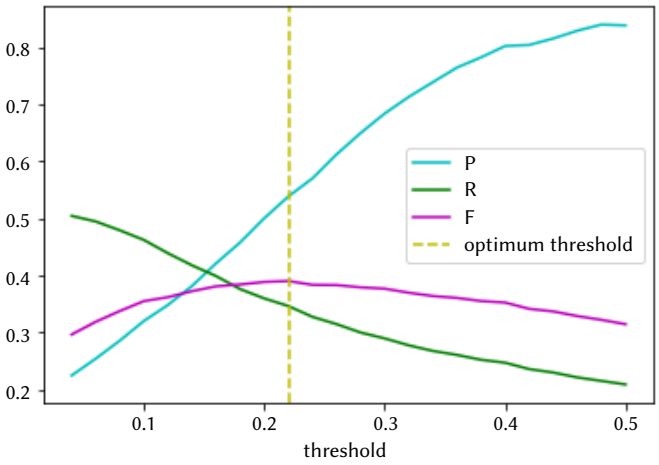


Fig. 5. Threshold calculation through MLS test after 180 epochs of training with MLS.

VI. Experiments and Results

A. Evaluation Metrics

MIREX's campaings use two evaluation measures which are *Median Deviation* and *Hit Rate*. The *Hit Rate* (aslo called F-score or F-measure) is denoted by F_β, where $\beta = 1$ is the measure most frequently used in previous works. Nieto et al. [52] set a value of $\beta = 0.58$, but the truth is that F_1 continues being the most used metric in MIREX works. We will later give our results for both β values. The *Hit Rate* score F_1 is normally evaluated for ± 0:5s and ± 3s time-window tolerances, but in recent works most of the results are given only for ± 0:5s tolerance which is the most restrictive one. We test our model with MIREX algoritm [53] which give us the Precision, Recall and F-measure parameters.

$$\text{Precision: } P = \frac{TP}{TP + FP} \tag{15}$$

$$\text{Recall: } R = \frac{TP}{TP + FN} \tag{16}$$

TABLE VI. Results of Boundary Estimation With Tolerance ± 0:5S and Optimum Threshold in Terms of F-score, Precision and Recall. Note That Results form Previous Works Did Not Use the Same Threshold Value

Input	Train Database	Epochs	Thresh.	P	R	F_1(std)	$F_{0.58}$
Tolerance: ± 0:5s with `2pool3` matrices							
$MLS + SSLM^{MFCCs}_{euclidean}$	SALAMI	140	0.24	0.441	0.415	0.402 (0.163)	0.414
$MLS + SSLM^{MFCCs}_{cosine}$	SALAMI	140	0.24	0.428	0.407	0.396 (0.158)	0.404
$MLS + (SSLM^{MFCCs}_{euclidean} + SSLM^{chromas}_{euclidean})$	SALAMI	100	0.24	0.465	0.400	0.407 (0.160)	0.419
$MLS + (SSLM^{MFCCs}_{cosine} + SSLM^{chromas}_{cosine})$	SALAMI	100	0.24	0.444	0.416	0.404 (0.166)	0.417
$MLS + (SSLM^{MFCCs}_{euclidean} + SSLM^{MFCCs}_{cosine})$	SALAMI	100	0.24	0.445	0.421	0.409 (0.173)	0.416
$MLS + (SSLM^{chromas}_{euclidean} + SSLM^{chromas}_{cosine})$	SALAMI	100	0.24	0.457	0.396	0.400 (0.157)	0.420
$MLS + (SSLM^{chromas}_{euclidean} + SSLM^{chromas}_{cosine} + SSLM^{MFCCs}_{euclidean} + SSLM^{MFCCs}_{cosine})$	SALAMI	100	0.26	0.526	0.374	**0.411 (0.169)**	**0.451**
End-to-end previous works							
$MLS + SSLM^{MFCCs}_{cosine}$ [4] (2015)	Private	-		0.646	0.484	0.523	0.596
$MLS + SSLM^{MFCCs}_{cosine}$ [35] (2017)	SALAMI	-		0.279	0.300	0.273 (0.132)	-
$MLS + (SSLM^{MFCCs}_{cosine} + SSLM^{chromas}_{cosine})$ [35] (2017)	SALAMI	-		0.470	0.225	0.291 (0.120)	-

$$\text{F measure: } F_\beta = (1+\beta^2)\frac{P \cdot R}{\beta^2 \cdot P + R} \quad (17)$$

Where:

- TP: True Positives. Estimated events of a given class that start and end at the same temporal positions as reference events of the same class, taking into account a tolerance time-window.
- FP: False Positives. Estimated events of a given class that start and end at temporal positions where no reference events of the same class does, taking into account a tolerance time-window.
- FN: False Negatives. Reference events of a given class that start and end at temporal positions where no estimated events of the same class does, taking into account a tolerance timewindow.

B. Results

1. Isolated Inputs: Distances, Audio Features and Pooling Strategies

We first trained the Neural Network with each input matrix (see Fig. 3) separately in order to know what input performs better. We trained the model using the MLS and SSLMs obtained from MFCCs and Chromas and applying Euclidean and cosine distances, and we also give the results for both of the pooling strategies mentioned before, `6pool` (lower resolution) and `2pool3` (higher resolution). As mentioned in section IV, we removed the first label of the SALAMI text files corresponding to 0.0s label. Results in terms of F score, Precision and Recall are showed in Table V. Note that the results showed from previous works used a different threshold value.

The best-performing input when training our model with isolated inputs is the MLS which has a F_1 value of 0.389 (see Table V). Taking only into account the `6pool` pooling strategy, regarding the SSLMs computed from audio features (MFCCs and chromas) we found that the best-performing SSLMs are the ones that are computed from the MFCCs with more than a 5% difference with the SSLMs computed from chromas.

According to the distance measures with which we compute the SSLMs, we found that there is not a high impact on the results when computing the SSLMs with Euclidean or cosine distances. The F_1 difference between the SSLMs computed with Euclidean or cosine distances is not higher than 1%. Overall, the best-performing SSLM for the `6pool` pooling strategy is the $SSLM^{MFCCs}_{euclidean}$ with a F_1 value of 0.361, which is a 2.8% less than the MLS $F_!$ value of 0.389.

In view of the results in Table V, we can affirm that doing a max-pooling of 2, then computing the SSLMs and doing another max-pooling of 3 afterwards (`2pool3`) slightly improves the results but it does not make a high impact in the performance. The best-performing (`2pool3`) SSLM, the $SSLM^{MFCCs}_{euclidean}$ has a F_1 value of 0.375, which is less than a 2% of the F_1 value of 0.361 for the same SSLM but computed with the `6pool` pooling strategy. This procedure not only takes much more time to compute the SSLMs but also the training takes also much more time and it does not perform better results in terms of F-score.

In Fig. 6 we show an example of the boundaries detection results for some of our input variants on the MLS and SSLMs. We obtained lower results than [4] but higher results than [35] who tried to re-implement [4]. The reasons for this difference could be that the database used by Grill and Schlüter [4] to train their model had 733 non-public pieces. Cohen and Peeters [35], as in our work, trained their model only with pieces from the SALAMI database, so that our results can be compared with theirs, since we trained, validated and tested our Neuronal Network with the same database (although they had 732 SALAMI pieces and we had 1006).

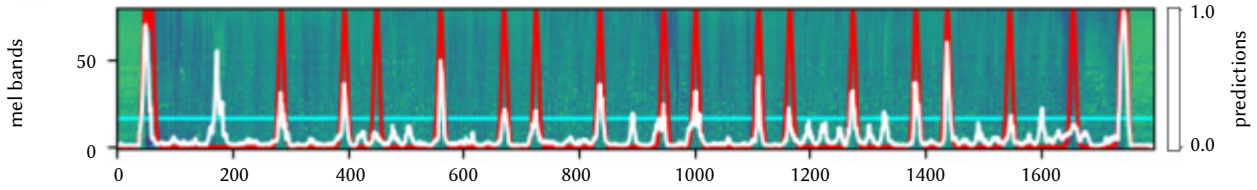
2. Inputs Combination

With the higher results in Table V we make a combination of them as in [4] and later in [35]. A summary of our results can be found in Table VI.

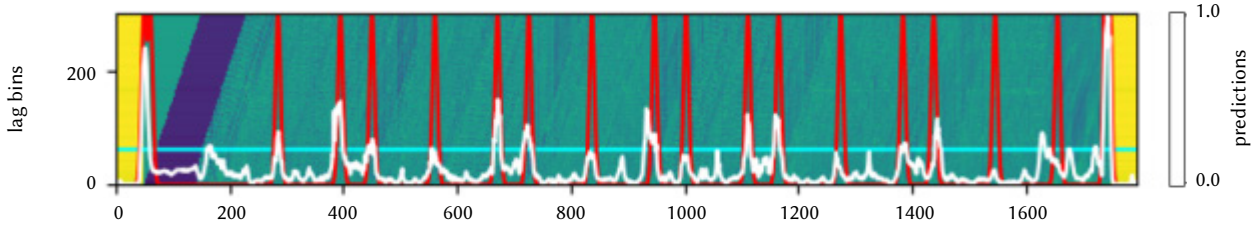
The inputs combination that performs the best in [35] was $MLS + (SSLM^{MFCCs}_{cosine} + SSLM^{chromas}_{cosine})$ for which $F_1 = 0.291$. We overcome that result for the same combination of inputs obtaining they obtained a F score $F_1 = 0.404$. In spite that, previous works [4] says that cosine distance performs better, we proof that in our model the Euclidean distance gives us better results. We also found that the best-performing inputs combination is $MLS + (SSLM^{chromas}_{euclidean} + SSLM^{chromas}_{cosine} + SSLM^{MFCCs}_{euclidean} + SSLM^{MFCCs}_{cosine})$ for which $F_1 = 0.411$. There is not a huge improvement in the F-measure obtained with this combination in comparison with the results obtained with the combination of the MLS with two SSLMs, but it is still our best result.

VII. Discussion

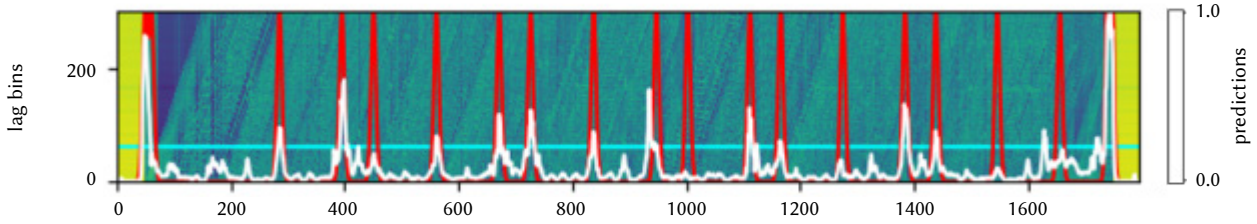
We can affirm that the best-performing input, when training the model with isolated inputs, is the Mel Spectrogram, which has a F_1 equal to 0.389, more than a 2% higher than the next bestperforming input respresentation, the $SSLM^{MFCCs}_{euclidean}$, whose F_1 is equal to 0.361 (Table V).



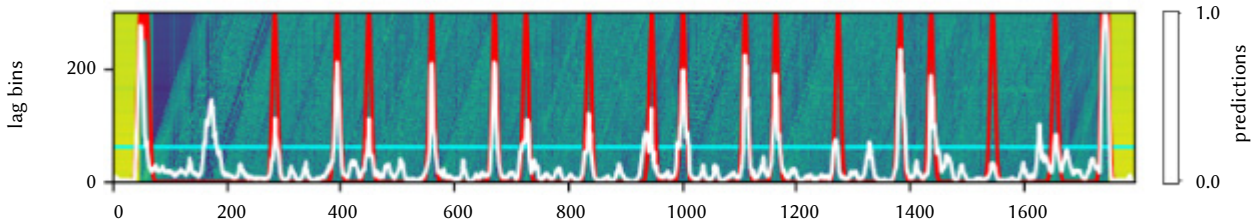
(a) CNN predictions on MLS.



(b) CNN predictions on SSLM calculated with MFCCs and Euclidean distance with 2pool3 (best-performance SSLM input in terms of F-measure). In this case F_1 = 0.486 for a ± 0.5s tolerance.



(c) CNN predictions on SSLM calculated with MFCCs and cosine distance with 2pool3. In this case F_1 = 0.686 for a ± 0.5s tolerance.



(d) CNN predictions on SSLM from MFCCs with cosine distance for model MLS + ($\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$) . In this case F1 = 0.75 for a ± 0.5s tolerance.

Fig. 6. Boundaries predictions using CNN on different inputs obtained from the "Live at LaBoca on 2007-09-28" of DayDrug corresponding to the 1358 song of SALAMI 2.0 database. The ground truth from SALAMI annotations are the gaussians in red, the model predictions is the white curve and the threshold is the horizontal yellow line. Note that the prediction have been rescaled in order to plot them on the MLS and SSLMs images. All these images have been padded according to what is explained in the previous paragraphs and then normalized to zero mean and unit variance.

We have also demonstrated that by computing a max-pooling of factor 6 at the beginning of the process not only takes much less pre-processing time but also the training of the Neural Network is faster and it does not affect the results as much as it could be expected. As an example, the $\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ obtained with the `6pool` method has an F_1 value of 0.361 versus the `2pool3` method for the same input which F_1 is equal to 0.375.

Despite the fact that we could not replicate some previous studies of Ullrich et al. [3] and Grill et al. [4] which used nearly the same model that the one which we described in our work, we outperform the results in Cohen et al. [35] work, who also tried to re-implement the model described in the previous literature. There has to be highlighted the fact that previous studies of Ullrich et al. [3] and Grill et al. [4] had at their disposition a private dataset of 733 pieces that they used for training the model, and in this paper the model has been trained only with the public available dataset of SALAMI 2.0.

Adding more inputs to the model does not improve the results in a significant way and it is very time consuming, specially in our last case of study where we take 4 SSLMs in combination with the Mel Spectrogram, which has a *F*1 value of 0.411 in contrast with the *F*1 value of the MLS + $\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ case which is 0.402, so the difference is less than 1%. This leads us to suggest that the use of another neural network architecture that only uses the Mel spectrogram with a SSLM could outperform the current results.

The results obtained in this work improve those presented previously with the same database. However, the accuracy in obtaining the boundaries in musical pieces is relatively low and, to some extent, difficult to use. This makes it necessary, on the one hand, to continue studying different methods that allow a correct structural analysis of music and, on the other hand, to obtain databases that are properly labeled and contain a high number of musical pieces. In any case, the results obtained are promising and allow us to adequately set out the bases for future work.

VIII. Conclusions

In this work we have developed a comparative study to determine the most efficient way to compute the inputs to a convolutional neural network to identify boundaries in musical pieces, combining different methods of generating SSLM matrices. In order to make the

comparison and analyse the optimal way to perform the boundary detection task in MSA, different audio features and different pooling strategies have been employed, as well as the combination of different inputs to the CNN.

With an adequate combination of input matrices and pooling strategies, we obtain an accuracy F1 of 0.411 that outperforms the current one obtained under the same conditions (same input data and same datasets for training and testing). In spite of the fact that the best result is given by combining four SSLMs and the MLS, the difference in the F-measure value between our best result and experiments which require less input data and whose training time is lower, is not as high as what it could be expected. We can also affirm that current methods that have been used to date to face music boundary detection do not perform well, so MSA task needs further research because it is not solved yet.

Future work should use new Neural Network architectures that have not been used to solve MSA yet. Architectures employed in language models from Natural Language Processing such as Transformers can lead to out-perform the actual results that are presented in this work due to the memory improvement that they provide in comparison with Long-Short Term Memory Networks (LSTMs). In the case of Transformers, the self-attention mechanism can help the model to better-process the SSMs and SSLMs matrices. Further research, as it has been mentioned before, should also take into account to perform some data augmentation on the current public available datasets in order to have more data to train deep Neural Network models. Data augmentation, if done, should be done with pitch-shifting or by adding Gaussian noise to the inputs, but they should not use rotation or scaling techniques which affect the time distances of the input representations (horizontal axes) and thus, the structure of the music pieces.

References

[1] O. Nieto, G. J. Mysore, C. Wang, J. B. L. Smith, J. Schlüter, T. Grill, B. McFee, "Audio-based music structure analysis: Current trends, open challenges, and applications," *Transactions of the International Society for Music Information Retrieval*, vol. 3, no. 1, pp. 246–263, 2020.

[2] M. Müller, *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications*. Springer, 2015.

[3] K. Ullrich, J. Schlüter, T. Grill, "Boundary detection in music structure analysis using convolutional neural networks," in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, 2014, pp. 417–422.

[4] T. Grill, J. Schlüter, "Music boundary detection using neural networks on spectrograms and self-similarity lag matrices," in *23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, August 31 - September 4, 2015*, 2015, pp. 1296–1300, IEEE.

[5] T. Grill, J. Schlüter, "Music boundary detection using neural networks on combined features and two-level annotations," in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, 2015, pp. 531–537.

[6] J. Foote, "Visualizing music and audio using self-similarity," in *Proceedings of the 7th ACM International Conference on Multimedia '99, Orlando, FL, USA, October 30 - November 5, 1999, Part 1*, 1999, pp. 77–80, ACM.

[7] M. Goto, "A chorus-section detecting method for musical audio signals," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '03, Hong Kong, April 6-10, 2003*, 2003, pp. 437–440, IEEE.

[8] T. Zhang, C. J. Kuo, "Heuristic approach for generic audio data segmentation and annotation," in *Proceedings of the 7th ACM International Conference on Multimedia '99, Orlando, FL, USA, October 30 - November 5, 1999, Part 1*, 1999, pp. 67–76, ACM.

[9] J. Paulus, M. Müller, A. Klapuri, "State of the art report: Audio-based music structure analysis," in *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*, 2010, pp. 625–636, International Society for Music Information Retrieval.

[10] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *2000 IEEE International Conference on Multimedia and Expo, ICME 2000, New York, NY, USA, July 30 - August 2, 2000*, 2000, p. 452, IEEE Computer Society.

[11] F. Kaiser, G. Peeters, "Multiple hypotheses at multiple scales for audio novelty computation within music," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, 2013, pp. 231–235, IEEE.

[12] B. Logan, S. M. Chu, "Music summarization using key phrases," in *IEEE International Conference on Acoustics, Speech, and Signal Processing. ICASSP 2000, 5-9 June, 2000, Hilton Hotel and Convention Center, Istanbul, Turkey*, 2000, pp. 749–752, IEEE.

[13] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden markov models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 360–370, 1999.

[14] M. Levy, M. B. Sandler, "Structural segmentation of musical audio by constrained clustering," *IEEE Transactions on Speech and Audio Processing*, vol. 16, no. 2, pp. 318–326, 2008.

[15] C. J. Tralie, B. McFee, "Enhanced hierarchical music structure annotations via feature level similarity fusion," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, 2019, pp. 201–205, IEEE.

[16] B. McFee, J. P. Bello, "Structured training for large-vocabulary chord recognition," in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, 2017, pp. 188–194.

[17] L. Lu, M. Wang, H. Zhang, "Repeating pattern discovery and structure analysis from acoustic music data," in *Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval, MIR 2004, October 15-16, 2004, New York, NY, USA*, 2004, pp. 275–282, ACM.

[18] J. Paulus, A. Klapuri, "Music structure analysis by finding repeated parts," in *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, New York, NY, USA, 2006, p. 59–68, Association for Computing Machinery.

[19] D. Turnbull, G. R. G. Lanckriet, E. Pampalk, M. Goto, "A supervised approach for detecting boundaries in music using difference features and boosting," in *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria, September 23-27, 2007*, 2007, pp. 51–54, Austrian Computer Society.

[20] B. McFee, D. Ellis, "Dp1, mp1, mp2 entries for mirex 2013 structural segmentation and beat tracking," *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2013.

[21] M. C. McCallum, "Unsupervised learning of deep features for music segmentation," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, 2019, pp. 346–350, IEEE.

[22] J. Schlüter, S. Böck, "Improved musical onset detection with convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, 2014, pp. 6979–6983, IEEE.

[23] Y. Zou, M. Thiel, M. C. Romano, J. Kurths, "Analytical description of recurrence plots of dynamical systems with nontrivial recurrences," *International Journal of Bifurcation and Chaos*, vol. 17, no. 12, pp. 4273–4283, 2007.

[24] J. Paulus, A. Klapuri, "Music structure analysis with a probabilistic fitness function in MIREX2009," in *Proceedings of the Fifth Annual Music Information Retrieval Evaluation eXchange*, Kobe, Japan, October 2009. Extended abstract.

[25] M. Mauch, K. C. Noland, S. Dixon, "Using musical structure to enhance automatic chord transcription," in *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*, 2009, pp. 231–236, International Society for Music Information Retrieval.

[26] G. Sargent, S. A. Raczynski, F. Bimbot, E. Vincent, S. Sagayama, "A music structure inference algorithm based on symbolic data analysis." MIREX - ISMIR 2011, Oct. 2011. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00618141, Poster.

[27] F. Kaiser, T. Sikora, G. Peeters, "Mirex 2012-music structural segmentation task: Ircamstructure submission," *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2012.

[28] O. Nieto, J. P. Bello, "Mirex 2014 entry: 2d fourier magnitude coefficients,"

Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX), 2014.

[29] C. Cannam, E. Benetos, M. Mauch, M. E. Davies, S. Dixon, C. Landone, K. Noland, D. Stowell, "Mirex 2015: Vamp plugins from the centre for digital music," *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2015.

[30] O. Nieto, "Mirex: Msaf v0. 1.0 submission," 2016.

[31] J. Schlüter, K. Ullrich, T. Grill, "Structural segmentation with convolutional neural networks mirex submission," *Tenth running of the Music Information Retrieval Evaluation eXchange (MIREX 2014)*, 2014.

[32] T. Grill, J. Schlüter, "Structural segmentation with convolutional neural networks mirex submission," *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, p. 3, 2015.

[33] J. Serrà, M. Müller, P. Grosche, J. L. Arcos, "Unsupervised music structure annotation by time series structure features and segment similarity," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1229–1240, 2014.

[34] G. Sargent, F. Bimbot, E. Vincent, "A regularity-constrained viterbi algorithm and its application to the structural segmentation of songs," in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, 2011, pp. 483–488, University of Miami.

[35] A. Cohen-Hadria, G. Peeters, "Music structure boundaries estimation using multiple self-similarity matrices as input depth of convolutional neural networks," in *AES International Conference Semantic Audio 2017, Erlangen, Germany, June 22-24, 2017*, 2017, Audio Engineering Society.

[36] J. S. Downie, A. F. Ehmann, M. Bay, M. C. Jones, "The music information retrieval evaluation exchange: Some observations and insights," in *Advances in Music Information Retrieval*, vol. 274 of *Studies in Computational Intelligence*, Z. W. Ras, A. Wieczorkowska Eds., Springer, 2010, pp. 93–115.

[37] M. Goto, *et al.*, "Development of the rwc music database," in *Proceedings of the 18th International Congress on Acoustics (ICA 2004)*, vol. 1, 2004, pp. 553–556.

[38] M. Goto, "AIST annotation for the RWC music database," in *ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 8-12 October 2006, Proceedings*, 2006, pp. 359–360.

[39] F. Bimbot, E. Deruty, G. Sargent, E. Vincent, "Methodology and conventions for the latent semiotic annotation of music structure," 2012.

[40] A. F. Ehmann, M. Bay, J. S. Downie, I. Fujinaga, D. D. Roure, "Music structure segmentation algorithm evaluation: Expanding on MIREX 2010 analyses and datasets," in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, 2011, pp. 561–566, University of Miami.

[41] J. B. Smith, E. Chew, "A meta-analysis of the mirex structure segmentation task," in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR, 2013, Curitiba, Brazil*, vol. 16, 2013, pp. 45–47.

[42] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015.

[43] J. Serrà, M. Müller, P. Grosche, J. L. Arcos, "Unsupervised detection of music boundaries by time series structure features," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*, 2012, AAAI Press.

[44] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. D. Roure, J. S. Downie, "Design and creation of a large-scale database of structural annotations," in *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, 2011, pp. 555–560, University of Miami.

[45] A. A. Alvarez, F. Gómez-Martin, "Motivic pattern classification of music audio signals combining residual and LSTM networks," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 6, pp. 208–214, 2021.

[46] K. K. Verma, B. M. Singh, H. L. Mandoria, P. Chauhan, "Two-stage human activity recognition using 2d-convnet," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 2, pp. 1–11, 2020.

[47] M. Khari, A. K. Garg, R. G. Crespo, E. Verdú, "Gesture recognition of RGB and RGB-D static images using convolutional neural networks," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 5, no. 7, pp. 22–27, 2019.

[48] S. Jha, A. Dey, R. Kumar, V. K. Solanki, "A novel approach on visual question answering by parameter prediction using faster region based convolutional neural network," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 5, no. 5, pp. 30–37, 2019.

[49] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett Eds., Curran Associates, Inc., 2019, pp. 8024–8035.

[50] D. P. Kingma, J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[51] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: http://tensorflow.org/, Software available from tensorflow.org.

[52] O. Nieto, M. M. Farbood, T. Jehan, J. P. Bello, "Perceptual analysis of the f-measure to evaluate section boundaries in music," in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, 2014, pp. 265–270.

[53] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. W. Ellis, "Mir_eval: A transparent implementation of common MIR metrics," in *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, 2014, pp. 367–372.

Carlos Hernández Oliván

Carlos Hernández Oliván is a PhD student in Electronics at the Universidad de Zaragoza under the supervision of Dr. José R. Beltrán. He received the B.E. and M.Sc. degrees in Industrial Engineering in 2017 and 2019, respectively. He studied viola at the Professional Conservatory of Zaragoza where he received his professional certificate in 2013. He is a researcher at the Department of Electronic Engineering and Communications, University of Zaragoza. His research interests are focused on Music Information Retrieval, in particular, on the music analysis and generation systems with Artificial Intelligence. He is a student member of the International Society of Music Information Retrieval since March 2021.

José R. Beltrán

José R. Beltrán received the M.Sc. and Ph.D. degrees in Physics from the University of Zaragoza, Zaragoza, Spain, in 1988 and 1994, respectively. He is an Associate Professor with the Department of Electronic Engineering and Communications, University of Zaragoza. He has been involved in different research and development projects on Audio Analysis and Processing. His research interests are focused on the study of Automatic Learning Systems for the analysis, processing and synthesis of the musical signal. In 2008, he was a promoter of an academic spin-off: ARSTIC Audiovisual Solutions S.L. devoted to the use of technologies for the artistic and audiovisual fields. Prof. Beltrán is a member of the Aragon Institute for Engineering Research (I3A), Reseach Group in Advanced Interfaces (AffeciveLab).

David Diaz-Guerra

David Diaz-Guerra is a Ph.D. candidate at the University of Zaragoza (Spain), where he received the Bachelor's and Master's degrees in Telecommunications Engineering in 2017 and 2015, respectively. His research focuses on signal processing and machine learning for audio applications.