This is a repository copy of *Masked Conditional Neural Networks for sound classification*.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/157085/

Version: Published Version

Article:
Medhat, Fady orcid.org/0000-0003-2827-4487, Chesmore, David orcid.org/0000-0002-0688-8376 and Robinson, John Allen orcid.org/0000-0003-0995-3513 (2020) Masked Conditional Neural Networks for sound classification. APPLIED SOFT COMPUTING. 106073. ISSN 1568-4946

https://doi.org/10.1016/j.asoc.2020.106073

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here:
https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

Applied Soft Computing Journal 90 (2020) 106073

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Masked Conditional Neural Networks for sound classification

Fady Medhat *, David Chesmore, John Robinson

Department of Electronic Engineering, University of York, United Kingdom of Great Britain and Northern Ireland

ARTICLE INFO

Article history:
Received 4 July 2019
Received in revised form 5 December 2019
Accepted 9 December 2019
Available online 17 January 2020

Keywords:
Restricted Boltzmann Machine
RBM
Conditional Restricted Boltzmann Machine
CRBM
Music Information Retrieval
MIR
Environmental Sound Recognition
ESR
Conditional Neural Networks
CLNN
Masked Conditional Neural Networks
MCLNN
Deep Neural Networks

ABSTRACT

The remarkable success of deep convolutional neural networks in image-related applications has led to their adoption also for sound processing. Typically the input is a time–frequency representation such as a spectrogram, and in some cases this is treated as a two-dimensional image. However, spectrogram properties are very different to those of natural images. Instead of an object occupying a contiguous region in a natural image, frequencies of a sound are scattered about the frequency axis of a spectrogram in a pattern unique to that particular sound. Applying conventional convolution neural networks has therefore required extensive hand-tuning, and presented the need to find an architecture better suited to the time–frequency properties of audio. We introduce the ConditionaL Neural Network (CLNN)[1] and its extension, the Masked ConditionaL Neural Network (MCLNN) designed to exploit the nature of sound in a time–frequency representation. The CLNN is, broadly speaking, linear across frequencies but non-linear across time: it conditions its inference at a particular time based on preceding and succeeding time slices, and the MCLNN use a controlled systematic sparseness that embeds a filterbank-like behavior within the network. Additionally, the MCLNN automates the concurrent exploration of several feature combinations analogous to hand-crafting the optimum combination of features for a recognition task. We have applied the MCLNN to the problem of music genre classification, and environmental sound recognition on several music (Ballroom, GTZAN, ISMIR2004, and Homburg), and environmental sound (Urbansound8K, ESC-10, and ESC-50) datasets. The classification accuracy of the MCLNN surpasses neural networks based architectures including state-of-the-art Convolutional Neural Networks and several hand-crafted attempts.

© 2020 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

Audio pattern analysis is a broad area of study that has been investigated for years, mostly focusing on speech recognition [1]. Other types of sound recognition problems in the field of Music Information Retrieval (MIR) have also been investigated, and are gaining increased attention from music industry leaders and businesses with the growing use of digital music content shared over the web. In parallel, Environmental Sound Recognition (ESR) is gaining a similar prominence with applications such as surveillance and noise recognition.

MIR involves several sub-areas according to the specificity of the task as discussed by Casey et al. [2] ranging from music identification, copyright monitoring, and melody detection to recommendation and genre recognition. The problem involves the ability to categorize music files to facilitate their retrieval based on the instruments played, the composer, the type of music and other tags that usually have to be labeled manually.

Music genre classification involves challenges related to the large number of variations of musical instruments, melodies, harmonies, timbre, and mixing between two or more genres in the same music piece. Much of the time, a musical piece can also involve human vocals. A number of variables have to be considered for a classification decision. It is beneficial to be able to automatically classify different genres based on the audible music contents to overcome the labor that goes into manual categorization.

The sounds in an environmental sound scene have a similar complexity to music, being composed of different sound mixtures in addition to possessing, to a certain extent, musical perceptual properties such as rhythm and timbre. Though these properties are clearer in music, they are inherent in many other kinds of sounds. The ESR problem gained early attention from the research community by applications related to environmental noise recognition [3] backed by studies that show the effect of noise on the human health [4], which induced international organizations to consider the problem [5]. ESR was also explored for other purposes such as surveillance [6], non-invasive detection of wood-boring pests [7] and assisting the elderly [8].

* Corresponding author.
E-mail addresses: fady.medhat@york.ac.uk (F. Medhat), david.chesmore@york.ac.uk (D. Chesmore), john.robinson@york.ac.uk (J. Robinson).

[1] Code: https://github.com/fadymedhat/MCLNN.

https://doi.org/10.1016/j.asoc.2020.106073
1568-4946/© 2020 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

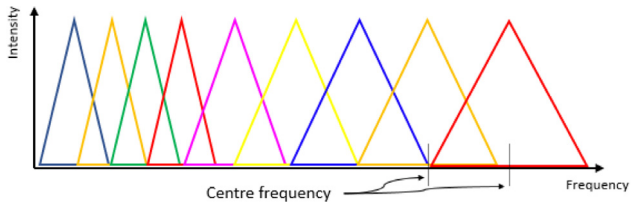

Fig. 1. A set of bandpass filters forming a Filterbank.

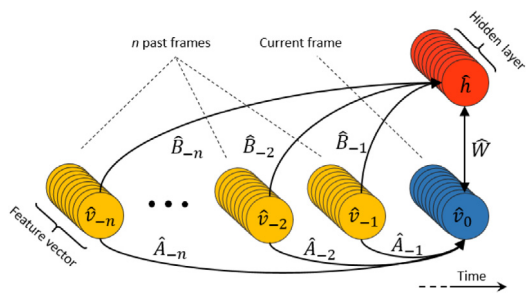

Fig. 2. Conditional Restricted Boltzmann Machine.

Since the 1990s and possibly earlier, there have been successful non-speech classification systems [9]. The majority were and are dependent on hand-crafting the most prominent acoustic features for distinguishing between different sound categories. Recently, with the success of the deep neural architectures, there is an endeavor to eliminate the efforts in hand-crafting the features required for classification. This is approached by applying deep architectures directly to the raw signal [10] or an intermediate representation, e.g. spectrograms. Hand-designed features [11–16] are still superior in most contexts, but the gap is getting smaller with deep learning evolving as a rival to minimize and hopefully eliminate the need to hand design the features required for classification.

The work by Soltau et al. [9] marks an early use of neural network architectures for audio feature extraction rather than for direct classification. A similar advanced method was approached through the use of stacked Restricted Boltzmann Machine (RBM) [17] forming a Deep Belief Net (DBN) [18] architecture, applied to music feature learning by Hamel et al. [19]. A recent attempt by Cakir et al. [20] used a deep neural network architecture to label different sound sources overlapping a temporal instance in an environmental sound scene. Convolutional Neural Networks (CNN) [21] have also been studied for different music tasks in [22,23], and with the scarcity of large labeled sound datasets for training, innovative CNN [24,25] models were introduced aiming to lend the CNN advances in image processing to sound.

A variety of signal representations have been used in the literature for sound recognition, broadly categorized into time and time–frequency methods. Both categories are rarely used in their raw format but often undergo an extensive feature engineering stage to extract distinctive properties that can enhance the recognition performance.

The Short-Time Fourier Transform (STFT) is often used in time–frequency representations. In the STFT, a window of 2^n samples around a time instant are Fourier transformed; the magnitudes of the transformed components are retained and the phase ignored, particularly for recognition systems. The number of frequency bins depends upon the number of samples in the window. The windows overlap and a windowing function (e.g. Hanning, Hamming, Kaiser, etc.) is applied to avoid the high frequency artifacts. Positioning the extracted slices of frequency amplitudes next to each other generates the 2D image-like "spectrogram".

The frequency components of a time–frequency representation at a temporal instance can be combined using filterbanks [26], for example as in Fig. 1. Filterbanks are formed of a group of bandpass filters each suppressing a range of frequencies while allowing others. Filterbanks may have different shapes to provide different scaling factors over the frequencies under consideration. They provide a weighted sum to aggregate the energies across the frequencies residing within the bandwidth of each bandpass filter. Filterbanks are used extensively in applications such as sound source separation [27].

A filterbank is designed based on the number of filters and their shape together with both the center frequency and bandwidth of each, which consequently affects the overlapping distance between the filters. For example, the Melody or Mel-scale is one of the widely adopted scales used for designing filterbanks, since it mimics the human auditory system, which linearly responds to low tones and logarithmically to higher ones. The Mel-scale is adopted by spectrogram transformations such as the Mel-Frequency Cepstral Coefficients (MFCC), used extensively in speech recognition.

Training a neural network with a spectrogram can be done at the frame-level as in DBNs [19] that treat the spectral frames as static, isolated entities ignoring the inter-frame relationship, as in Bag-of-frames classification [28], or using a window of frames to capture the context. Since we are dealing with a 2D image-like spectrogram, the CNN is an intuitively plausible neural-based model. But contrary to images, the two dimensions of a spectrogram have completely different meanings. Moreover, the amplitude of a frequency bin at a certain temporal instance is composed of the sum of energies generated from overlapping sound events. And whereas objects in images tend to be spatially contiguous, the energies of frequencies of a sound event in a spectrogram are distributed about the spectrum. The fundamental frequency, harmonics and overtone frequencies of a sound event will reside at different spatial locations across the frequency bins of a spectrogram, yet all of them contribute to the energy of the same source.

The ConditionaL Neural Network (CLNN)[2] together with its extension the Masked ConditionaL Neural Network (MCLNN), which we introduce in this work, tackle the preservation of the spatial locality (time-wise and frequency-wise) of features in the training of weights. The temporal inter-frame relation is captured by the CLNN while preserving the time and frequency specificity of the features through dedicated connections between each temporal frame and the hidden layer. The MCLNN extends the CLNN using a binary mask. The masking operation, enforced over the connections in the MCLNN, emulates combining frequencies in a controlled fashion. The mask follows a band-like representation in subdividing the frequency bins of a spectrogram into bands. The MCLNN embeds this behavior within the network, allowing it to sustain the non-contiguous distribution of frequencies.

The remainder of the paper is organized as follows: Section 2 reviews the most relevant models to this work. Section 3 explains the ConditionaL Neural Network architecture, and Section 4 extends the discussion to the Masked ConditionaL Neural Networks. We present the experimental findings in Section 5 together with an analysis of this work and a comparison to Convolutional Neural Networks. Section 6 provides a discussion of this work. Finally, we conclude the paper and consider future work in Section 7.

[2] Portions of this work appeared in [29–34], and [35].

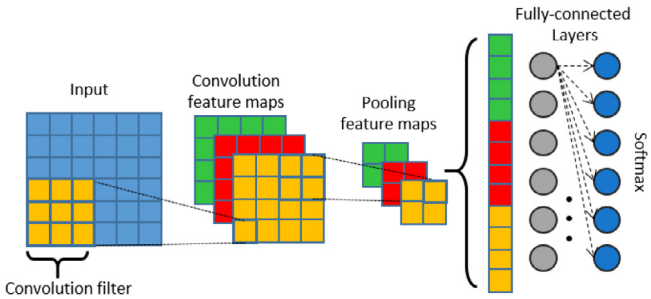

Fig. 3. Convolutional Neural Network.

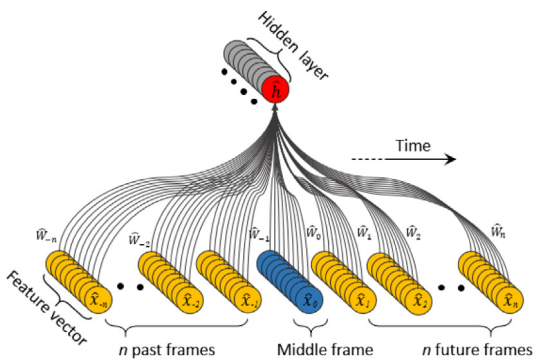

Fig. 4. ConditionaL Neural Network layer. Connections depicted are for a single hidden node.

2. Related models

As noted earlier, several neural network based attempts have been used for sound. Disregarding the nature of the sound type (music, speech or general environmental sounds) the overlap between the methods used for feature extraction or recognition is wide. In this section, we discuss in more detail the most relevant models for this work.

The Conditional Restricted Boltzmann Machine (CRBM) [36] by Taylor et al., an extension to the RBM, is designed to take into consideration the sequential relationship between the feature vectors in a temporal signal. The CRBM architecture, shown in Fig. 2 involves directed links (forming the conditional relation) from the previous visible vectors $\hat{v}_{-n}, \ldots, \hat{v}_{-2}, \hat{v}_{-1}$ (feature vectors) to both the hidden layer $\hat{h}$ and the current visible input vector $\hat{v}_0$, in addition to the undirected links between the visible and hidden nodes depicted in the figure by $\hat{W}$.

The weight tensor $\hat{B}$ is for the weights between the n past frames and the hidden layer. The weight tensor $\hat{A}$ holds the autoregressive relation between the past n frames and the current frame at the visible layer $\hat{v}_0$. The Interpolating CRBM (ICRBM) [37], a CRBM extension, was used for phoneme classification in speech recognition. The ICRBM outperformed the CRBM as it considers the influence of the future frames in addition to past ones. Both the CRBM and ICRBM inspired the models presented in this work. However, they are generative models that behave as a dense network with multiple temporal steps. They are fed multiple frames of a spectrogram when applied to sound disregarding a sound signal dispersion of energies across the frequency bins of a spectrogram or any two-dimensional representation.

The Convolutional Neural Network (CNN) [21], used extensively for processing images [38–41], is an architecture based on the interleaving of several pairs of two operations named convolution and pooling. As shown in Fig. 3, these two operations generate a number of feature maps that are flattened to a single feature vector classified using one or more densely connected layers or passed onto a more conventional classifier such as an SVM. As discussed above a time–frequency spectrogram is image-like but with different spatial properties to a natural image. This fact led to work by Abdel-Hamid et al. [42] to tailor the CNN filters to cope with the nature of the sound signal using limited weight sharing. Pons et al. [23] proposed different CNN architectures exploiting music-related properties using different filter shapes within the same model for each of the time and frequency dimensions. Kereliuk et al. explored a similar model in [43]. Another tailored deep architecture that considered a set of filters dedicated to music and another set for speech with a merging stage for the features extracted from both types of filters was proposed by Barros et al. in [44]. Wyse [45] investigated using a CNN channel for each frequency bin in a spectrogram.

Long Short-Term Memory (LSTM) [46] is a sequence labeling recurrent neural network (RNN) that was initially introduced to tackle the vanishing and exploding gradient [47] problems faced by RNN. These problems occur in the RNN while learning dependencies over a long temporal window (Backpropagation through time). The RNN incorporates a feedback loop from the output state of the previous input of a sequential signal to be considered with the current input. LSTM upgraded this behavior through the use of an internally controlled memory cell to preserve the state. LSTM has achieved breakthrough results in handwriting recognition [48]. It has been used by Graves et al. [49] in deep LSTM-RNN architecture for phoneme recognition in speech. The success of the LSTM in text recognition is far more than its adoption to images or sound signals. This is attributed to the increased complexity and the training difficulties introduced by the additional weights of the memory cell especially with the increased number of features (frequency bins) in sound compared to text (literal mapped to numerical indices). This induced attempts similar to that of Choi et al. [50], where they introduce a Convolutional RNN (CRNN) for music classification tagging. Their CRNN is a combination of a CNN and an LSTM. They used a CNN to summarize the local features of a spectrogram that are further introduced to an LSTM to exploit the long-term dependencies across the frames.

Convolutional RBM (ConvRBM) by Lee et al. [51] is another extension to the RBM generative models. The ConvRBM adapted terminologies of the CNN to the RBM to allow scaling the RBM unsupervised training to images by sharing weights and by implementing a probabilistic pooling layer. The ConvRBM is formed of a binary visible layer and groups of hidden binary units. The nodes of each group are linked with a shared weight filter in addition to a shared bias. A probabilistic max-pooling layer is introduced in their work that follows the convolution layer. ConvRBM was adapted to sound in [52], where it was applied to several speech and music tasks. However, this work did not adapt to the particular structure of a processed spectrogram and therefore has the same shortcomings we discussed earlier for a CNN.

3. Conditional Neural Networks

The ConditionaL Neural Network (CLNN), we present in this work, is the main structure over which the mask in our Masked ConditionaL Neural Network (MCLNN), described in the next section, is applied.

The CLNN, like other previously proposed temporal models, operates over a window of frames to exploit interframe relationships. The CLNN implements the windowing behavior by including conditional connections similar to the directed connections from the visible to the hidden layer of the generative CRBM [36]. It considers future frames, in addition to past ones as studied in the ICRBM [37].

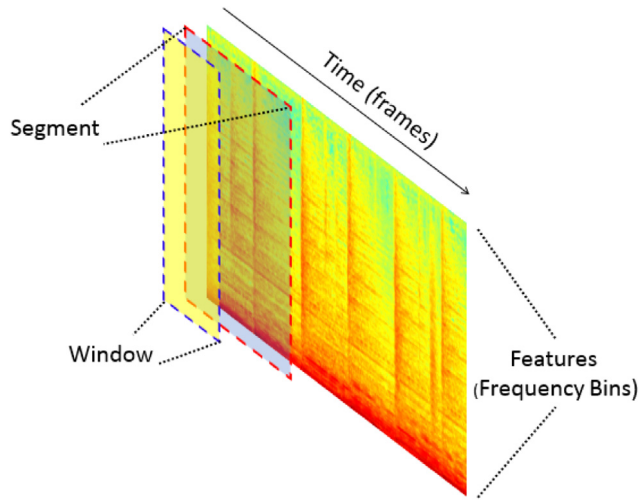

Fig. 5. A CLNN Segment of frames relative size to a Window.

Fig. 4 shows a single CLNN layer. The input is composed of d frames $(\hat{x}_{-n}, \ldots \hat{x}_{-2}, \hat{x}_{-1}, \hat{x}_0, \hat{x}_1, \hat{x}_2 \ldots \hat{x}_n)$ each of size l (i.e. l features). The middle frame of the window is $\hat{x}_0$, which is considered with n (referred to as the order) frames on either temporal directions. There are dense connections between each feature in any frame and each neuron in the hidden layer. The figure depicts the dense connection for a single neuron for simplicity, but there are similar connections between each neuron in the hidden layer and each of the feature vectors in the window. The number of frames in the window d follows (1).

$$d = 2n + 1, n \geq 1 \tag{1}$$

where n is the order. Accordingly, the observed pattern of the activation vector at the hidden layer for the middle frame of a window is conditioned on n previous and n succeeding frames. A single temporal step over d frames using a CLNN generates a single activation vector. Accordingly, the input frames required by a deep CLNN architecture have to account for the reduction in the number of frames at each layer. The input segment extracted from a spectrogram has to have q frames following (2)

$$q = (2n)\, m + k, \quad n, m \, and \, k \geq 1 \tag{2}$$

where n is the order, m is the number of layers and k is for the extra frames. These extra frames remain beyond the CLNN layers, which can be either flattened or globally pooled, as discussed in [53] for images, feature-wise to generate a single vector. The relative sizes between the spectrogram, the segment and the window are depicted in Fig. 5.

The temporal pooling behaves as aggregation over a texture window, which was studied in [54] for music. For example, at $n = 6$, $m = 3$ and $k = 10$, the input segment q at the first layer is $(2 \times 6) \times 3 + 10 = 46$ frames. The output of the first layer has $2n$ fewer frames than its input, i.e. $46 - 2n = 46 - (2 \times 6) = 34$ frames. Similarly, the output of the second and the third layers is 22 and 10, respectively. The 10 frames at the output of the third layer are the k extra frames. The extracted segments from the spectrogram can overlap with a maximum of $q - 1$ frames and a minimum of zero.

Fig. 6 shows the architecture of a two-layered CLNN ($m = 2$) with an order $n = 1$, i.e. each middle frame in a window is considered with one previous frame and one succeeding frame. Accordingly, each layer holds a 3-dimensional weight tensor $\hat{W}^b$, where $b = 1, 2, \ldots, m$. For an order $n = 1$, the depth of the weight tensor is 3. Therefore, for each of the frames within a window (3 frames at $n = 1$) there is a dedicated weight matrix having the same index u to process the frame through a vector–matrix multiplication. Accordingly, $\hat{W}_0^b$ is for the middle frame of the window at $u = 0$, and $\hat{W}_{-1}^b$ and $\hat{W}_1^b$ are for the off-center frames at $u = -1$ and $u = 1$, respectively. As shown in Fig. 6, the first CLNN layer feeds $q - 2n$ frames to the second CLNN layer, which in turn performs in a similar manner to generate another representation for succeeding layers. The final output for these two layers is one or more (based on the k extra frames) representative frames at the output of the second CLNN, which can be flattened or pooled across then fed to a densely connected network before the final softmax layer for classification.

The middle frame of a window together with its neighboring $2n$ frames are considered when the CLNN processes a frame at index t within a segment (The middle frame of a window, at $u = 0$, is also the frame at index t of the segment). Successive windows may overlap depending on the stride size. The activation of a single neuron of the hidden layer is formulated in (3)

$$y_{j,t} = f\left(b_j + \sum_{u=-n}^{n} \sum_{i=1}^{l} x_{i,u+t} W_{i,j,u}\right) \tag{3}$$

where $y_{j,t}$ is the output of neuron j of the hidden layer and t is the index of the frame within the segment q. The activation function is f, and the bias at the hidden neuron is b_j. The term $x_{i,u+t}$ is for feature i of the vector $\hat{x}_{u+t}$. Each feature in a vector of length l is multiplied by its corresponding weight element $W_{i,j,u}$, belonging to the matrix $\hat{W}_u$. The indices $[i, j]$ refers to the connection between the ith feature in the feature vector and jth hidden node. The index u is in the interval $[-n, n]$. Accordingly, the number of weight matrices matches the number of frames in a window. The output at the hidden layer formulated in a vector form is given in (4)

$$\hat{y}_t = f\left(\hat{b} + \sum_{u=-n}^{n} \hat{x}_{u+t} \cdot \hat{W}_u\right) \tag{4}$$

where $\hat{y}_t$ is the activation with respect to the frame at index t of the segment q. The transfer function f and the bias vector $\hat{b}$. The term $\hat{x}_{u+t}$ is the vector at index u in a window. A vector–matrix multiplication is applied between the frame at index u within the window d and its corresponding weight matrix at the same index. The dimensions of $\hat{W}_u$ is $[l, e]$, where e is the hidden layer width. The conditional distribution $p\left(\hat{y}_t \mid \hat{x}_{-n+t}, \ldots \hat{x}_{-1+t}, \hat{x}_t, \hat{x}_{1+t}, \ldots \hat{x}_{n+t}\right) = \sigma\left(\ldots\right)$ can be captured using a logistic transfer function, where σ can be the hidden layer sigmoid or the final layer softmax.

4. Masked conditional neural networks

We discussed earlier the filterbank used in signal analysis. The MCLNN, extending the structure of the CLNN, mimics a filterbank-like behavior through a systematic sparseness enforced over the connections between the input and the hidden layer within the network through a binary mask. The mask follows the structural pattern of the frequency bands in a spectrogram as shown in Fig. 7. The mask design is controlled by two tunable hyper-parameters namely: The Bandwidth bw and the Overlap ov. The Bandwidth controls the number of features to be considered in the same band (similar to a bandpass filter in a filterbank), and the Overlap controls the superposition distance between successive bands (mimicking the overlap between filters). For example, Fig. 7.a shows an example of a binary masking pattern of a $bw = 5$; this is represented by the consecutive ones positioned in a single column. The same masking pattern has an $ov = 3$; this is represented by the superposition of the binary patterns across

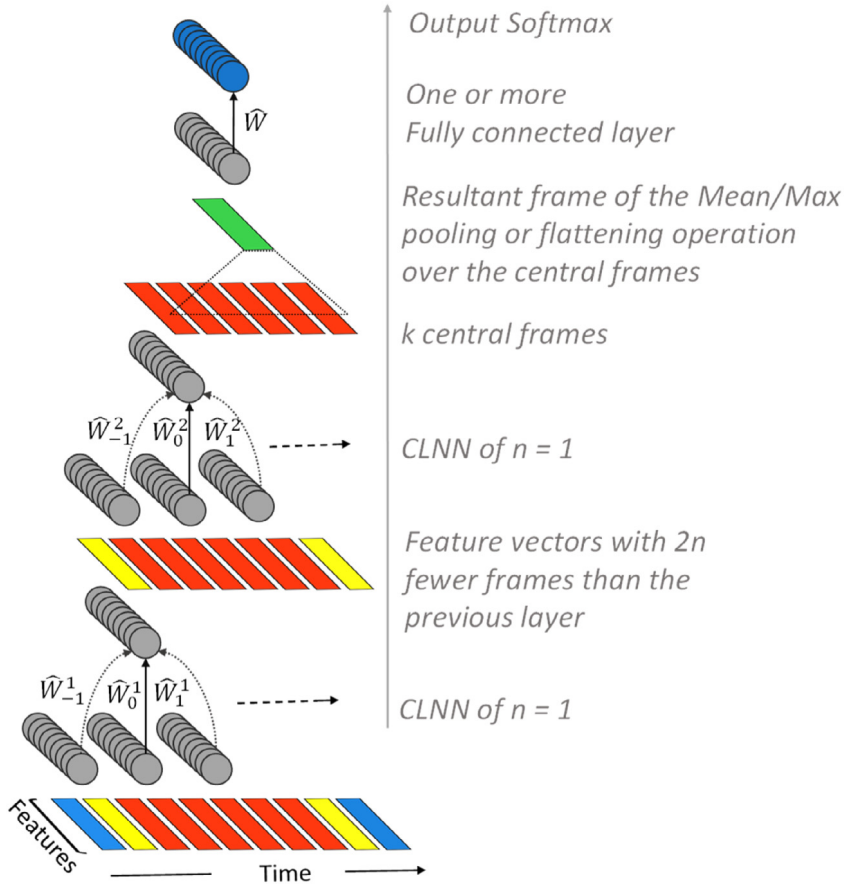

Fig. 6. A two-layer CLNN model with $\boldsymbol{n} = 1$.

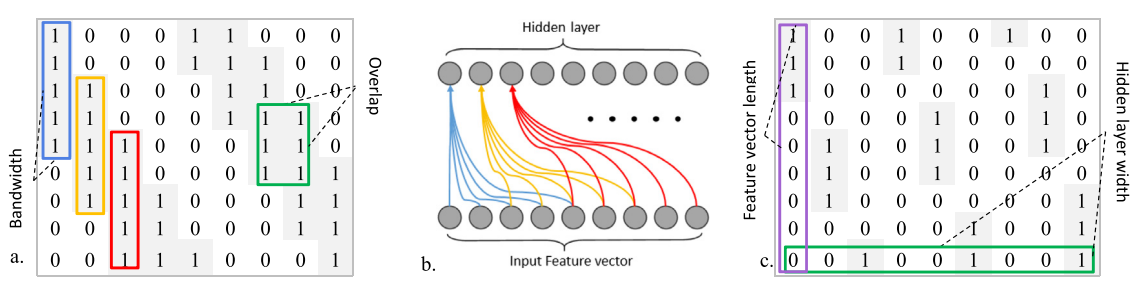

Fig. 7. Examples of the Mask patterns. (a) A bandwidth of 5 with an overlap of 3, (b) The allowed connections matching the mask in a. across the neurons of two layers, (c) A bandwidth of 3 and an overlap of -1.

the successive columns. Fig. 7.b shows the enabled links between the input and the hidden layer matching the masking pattern in Fig. 7.a. The Overlap can be assigned negative values that refer to the non-overlapping distance across the columns as shown in Fig. 7.c. Therefore, as the bandwidth increases, the scope of observation of a single hidden node over a partition of the feature vector is widened. The overlap ov and the bandwidth bw control the linear spacing of the 1's positions within a mask following (5)

$$lx = a + (g - 1)\,(l + (bw - ov)) \tag{5}$$

where lx is the linear index, bw is the bandwidth, ov is the overlap and l is the length of the feature vector (number of frequency bins). The values of a are within the interval $[1, bw]$ and g within the interval $[1, \lceil (l \times e)/(l + (bw - ov)) \rceil]$.

The filterbank-like pattern induces each neuron in the hidden layer to be activated through the influence of distinct active regions in the feature vector following the mask design. Meanwhile, the spatial locality of the learned features is preserved as the active weights' locations are fixed in position. The systematic sparseness allows the connections within a certain band of the input (as if they are frequency bins) to contribute to the activation of the hidden node.

Hand-crafted features do not only involve finding the best individual features, but also the optimum combination of features. The mask automates this process by embedding shifted versions of the filterbank-like pattern. This allows each neuron to learn differently about different regions of the feature vector. For example, in Fig. 7.a (with the columns mapping to neurons) the first neuron in a hidden layer (i.e. the first column in the mask) will learn about the first five features. Meanwhile, the fifth neuron will learn about the first and the last two features in the feature vector. Similarly, in Fig. 7.c. Accordingly, different feature combinations are considered concurrently.

This spatial constraint enforced over the weights together with the shifted versions of the masking pattern allows the selective fusing of the localized distributions of features that contribute to

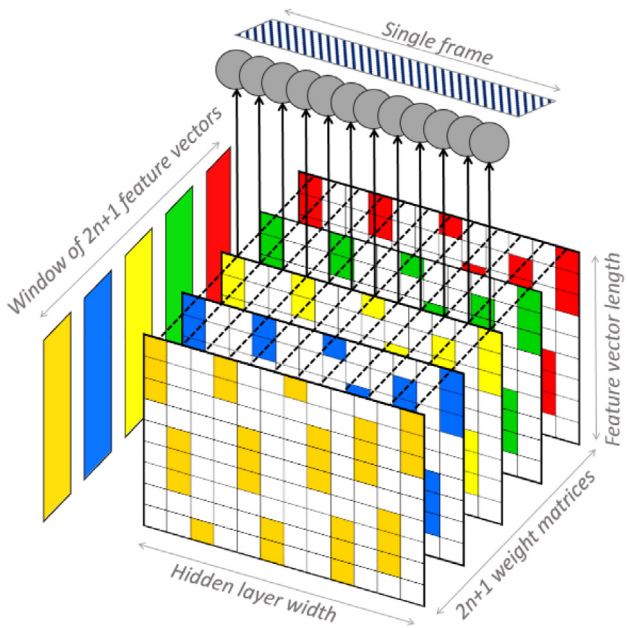

Fig. 8. A single step of MCLNN.

a certain neuron following the shifted binary pattern. This cope with the non-contagious distribution of the energies in a time–frequency representation, which the CNN filters are not designed to handle. Further to add, is the temporal aspect considered by the CLNN.

The masking operation is applied through an elementwise multiplication of a binary mask that has the same dimensions of the weight matrix. This is formulated in (6)

$$\hat{Z}_u = \hat{W}_u \circ \hat{M} \tag{6}$$

where $\hat{W}_u$ is the original weight matrix, $\hat{M}$ is the masking pattern and $\hat{Z}_u$ is the new weight matrix after the element-wise multiplication by the mask to substitutes $\hat{W}_u$ in (4).

Fig. 8 shows a single step of an MCLNN of order$n$. Accordingly, a window of frames of size $2n + 1$ is being processed with a weight tensor having matrices of a count (depth) $= 2n + 1$. Each frame in the window at an index u is processed with its corresponding matrix at the same index. The highlighted cells in the figure depict the active connections. The output of a step of an MCLNN over a window of frames is a single resultant frame.

Algorithm 1

MCLNN operation over t^{th} segment of a sound clip

$\hat{\boldsymbol{X}}$: two-dimensional input segment of size $\boldsymbol{q}$
$\hat{\boldsymbol{Y}}$: two-dimensional output segment of size $\boldsymbol{q} - 2\boldsymbol{n}$
$\hat{\boldsymbol{W}}$: three-dimensional weight tensor of shape $[2\boldsymbol{n} + 1, \boldsymbol{l}, \boldsymbol{e}]$
$\hat{\boldsymbol{M}}$: two-dimensional mask of binary patterns of shape $[\boldsymbol{l}, \boldsymbol{e}]$.
for $\boldsymbol{u} < 2\boldsymbol{n} + 1$
  $\hat{\boldsymbol{Z}}_{\boldsymbol{u}} = \hat{\boldsymbol{W}}_{\boldsymbol{u}} \circ \hat{\boldsymbol{M}}$ (weight matrix masked with the binary mask)
  $\hat{\boldsymbol{X}}' : \hat{\boldsymbol{X}}' + \hat{\boldsymbol{X}}_{[\boldsymbol{u}:\boldsymbol{u}+(\boldsymbol{q}-2\boldsymbol{n}),\boldsymbol{l}]} \cdot \hat{\boldsymbol{Z}}_{\boldsymbol{u}}$ (slice of $\hat{\boldsymbol{X}}$ from $\boldsymbol{u}$ to $\boldsymbol{u} + (\boldsymbol{q} - 2\boldsymbol{n})$)
$\hat{\boldsymbol{Y}} : \boldsymbol{f}\left(\hat{\boldsymbol{b}} + \hat{\boldsymbol{x}}'\right)$ (transfer function $\boldsymbol{f}$, bias $\hat{\boldsymbol{b}}$)

Algorithm 1 lists a pseudocode of the MCLNN operation over the tth segment of a sound clip. The temporal index t maps to the segment index of the clip, which is considered within the preprocessing of the clips into segments as discussed in Section 3.

Table 1
MCLNN Hyper-parameters for the MUSIC datasets.

Layer	Type	Nodes	Mask bandwidth	Mask overlap	Order n
1	MCLNN	220	40	−10	15
2	MCLNN	200	10	3	15

5. Experiments

The experiments in this section are split into three main categories. The first set of experiments are targeted to compare the performance of the proposed architecture against other neural network based attempts in the literature; the second set is focused on analyzing the effect of varying different hyperparameters; and the final set is dedicated to examining the performance of a standalone layer of an MCLNN, CNN, and Locally Connected Network (LCN) [55] that is similar to a CNN but without weight sharing. The evaluation throughout the experiments is conducted using several widely used datasets in the literature for either music (Ballroom, Homburg, ISMIR2004, and GTZAN) or environmental sound recognition (ESC-10, ESC-50, and UrbanSound8K). We present a dedicated section for each dataset with their relevant highlights.

5.1. Methodology

The evaluation of the MCLNN for sound classification follows the Multiple Instance Learning paradigm [56] used in most of the attempts referenced in the literature, where the original tag of the sound file is used to label each segment in isolation and the final decision for the clip follows a majority or probability voting mechanism across the predicted labels of the input segments.

All sound files for all datasets were resampled to a monaural channel with a sampling rate of 22050 Hz at a word depth of 16-bit. The resampled signal is transformed into a logarithmically Mel-scaled spectrogram. For music, it is 256 bins with an FFT window of 2048 and an overlap of 50%. For environmental sounds, it is 60 bins with the delta (the first derivative between frames) using an FFT window of 1024 sample and an overlap of 50%. The spectrogram and its delta are concatenated column-wise to form a feature vector of 120 bins. Further preprocessing involved extracting segments following (2). Therefore, the training of an MCLNN involves minimizing the categorical cross-entropy between the predicted labels of each segment and the target one using ADAM [57].

The final category of a clip is determined based on probability voting across the predicted vectors at the network's output corresponding to the input segments of a clip following (7)

$$Category = argmax_{j=1\ldots c}\left(\sum_{i=1}^{r} o_{ji}\right) \tag{7}$$

where r is the number of predicted output vectors following the number of total segments extracted from the spectrogram of the clip. Each output vector $\hat{o}$ has a length c, where c is the number of classes predicted by the softmax. The clip's category is decided by summing the predicted distributions across all the r predictions per class and choosing the maximum sum.

All reported accuracies are the mean of 10-fold cross-validation unless otherwise stated. The training folds are standardized feature-wise, and the z-score mean, and variance are used to standardize the validation and testing folds.

Regarding the model architecture, Rectified Linear Unit (ReLU) [58] is an activation function defined as $f(x) = \max(0, x)$. Despite the non-differentiability at zero, the function provides faster

Table 2
MCLNN Hyper-parameters for the Environmental Sound datasets.

Layer	Type	Nodes	Mask bandwidth	Mask overlap	Order n
1	MCLNN	300	20	−5	15
2	MCLNN	200	5	3	15

conversion and generalization compared to logistic activations without the need for unsupervised pre-training. ReLU is used extensively in deep architectures especially due to their ability to overcome the vanishing gradient occurring in sigmoid or tanh. The Parametric Rectified Linear Unit (PReLU) [59] is a generalization of the ReLU formulated as $f(x) = \max(0, x) + amin(0, x)$, which avoids zero gradients through the trainable coefficient a. In [59] an evaluation of the PReLU showed an outperformance regarding the classification accuracy compared to the ReLU. Accordingly, we adopted PReLU in this work. We used Dropout [60] as a regularization technique that simulates averaging several thinned networks.

We conducted a range of experiments using different architectures, and we chose the best performing models to report in this paper. But it is worth mentioning that despite the different distributions of the datasets used in these experiments, the MCLNN architectures are very similar with few differences of tunable hyperparameters across the datasets. The hyperparameters selection process of the MCLNN, similar to any neural network based model, is based on grid-search using the validation dataset and the mean accuracy of the cross-validation. In addition to the usual neural network parameters, the MCLNN involves the order n to specify the number of future and past frames. The bandwidth bw and overlap ov to control the binary mask design, and finally k to specify the number of frames to be used for pooling. The hyperparameters, used unless otherwise stated, are listed in Table 1 (for the music datasets, i.e. Ballroom, Homburg, GTZAN, and ISMIR2004) and Table 2 (for environmental sounds datasets, i.e. Urbansound8k, ESC-10, and ESC-50). The MCLNN layers are followed by a single-dimension global mean pooling layer [53], and either a fully-connected single or double layers of different sizes.

The experiments were carried out using Theano [61] and Keras [62] (operating on a TitanX GPU) for the implementation of the model, FFmpeg [63] for the sound files format conversion and LibROSA [64] for the signal transformation.

5.2. Sound classification

In this section, we evaluate the MCLNN performance against other attempts in the literature with a dedicated section for each dataset. It is important to highlight that we used a simple single or two-layered architecture of an MCLNN compared to other proposed models using complicated CNN architectures.

5.2.1. Ballroom

The Ballroom [65] dataset was used in 2004 within the ISMIR tempo contest. It is composed of 698 files, of 30 s length each, for eight sub-genres of ballroom music (unbalanced number of files per genre): ChaCha, Jive, Quickstep, Rumba, Samba, Tango, Viennese Waltz and Slow Waltz.

We explored a deep architecture composed of two MCLNN layers and another shallow architecture of higher order and wider k. Both architectures were followed by a global single dimensional mean pooling layer to pool the extra frames $k = 10$ and $k = 55$, respectively. Following the pooling layer, two fully-connected layers of 50 and 10 neurons were used before the final 8-way softmax for the classification decision. Wider segments showed a performance enhancement providing aggregation over a "texture window" [54,72], though Increasing the extra frames k beyond a certain limit depending on the dataset causes a degradation in the performance due to the inverse relation with the number of training samples available.

Table 3 lists the accuracies achieved on the Ballroom dataset using the MCLNN and other approaches from the literature. MCLNN achieved a mean accuracy across the 10-folds of 92.5% for the shallow architecture with long segments and 90.4% for a deep architecture with shorter segments. A neural network-based attempt was in [66], where Pons et al. designed musically motivated filters for a shallow CNN. The filters were designed to exploit the number of beats in the Ballroom dataset. On the other hand, a shallow MCLNN layer achieved a comparable without using any hand-crafted features or musical perceptual properties. Chen et al. in [67] explored the use of a three CNNs to capture local properties in combination with an LSTM for the long term temporal dependences. The three CNNs act as channels to learn the pitch, tempo and base musical features with specially designed filters for each channel. Further on, a fusing layer is used to combine the features learned by the three channels to be forwarded to the LSTM. The CNN in [23] achieved 87.68%. This attempt explored using dedicated filters to convolve both the frequency and time dimensions separately combined in the same model with filters designed for the Ballroom dataset.

With regard to handcrafted features, Peeters [16] exploited the Tempo annotations released with the dataset as one of the musical properties. He achieved an accuracy of 96.13%, but when he reapplied the classification without the tempo annotations, his method reached an accuracy of 88%. Marchand et al. [68] achieved 93.12% using a multi-stage feature extraction method involving onset-energy, auto-correlation and Modulation Scale Spectra, in addition to several other intermediate processing and design constrains. The 92.44% achieved by Seyerlehner et al. [69] applied several spectral features extracted from overlapping blocks of a spectrogram. Gouyon et al. [70] reported the accuracies using the tempo annotations in combination with their proposed features, and in another experiment using the tempo annotations alone. They reported an accuracy of 90.1% and 82.3% respectively. The MCLNN achieve competitive accuracy compared to highly crafted attempts either based on neural networks or handcrafted features without exploiting any musical or perceptual properties. Fig. 9 shows the confusion matrix using MCLNN on the Ballroom dataset. The true positive samples reported in [16] are comparable to the ones achieved by the MCLNN.

5.2.2. Homburg

The Homburg dataset [80] is composed of 1886 music clips for 9 music genres (Alternative, Blues, Electronic, Folk-Country, FunkSoul-Rnb, Jazz, Pop, RapHiphop, and Rock).

We used the MCLNN architecture described in the common section, but with an order $n = 5$ and extra frames $k = 2$. The pooling layer is followed by two densely-connected layers before the output softmax. Table 4 lists the accuracy achieved on the dataset through several attempts in the literature. The MCLNN surpassed several hand-crafted attempts in addition to the neural network-based Mean-Covariance RBM (mcRBM), a variant of the RBM, which achieved an accuracy of 55.3% in [73].

Other attempts [75,76] on the dataset employed a set of hand-crafted features using an auditory cortical representation, which models the cochlea using a Constant Q-Transform followed by a Wavelet transformation to extract a 4D cortical representation. This is combined with a set of MFCC and chroma features. The extracted features were used in combination with a specially designed classifier to exploit sparseness properties within the

Table 3
Performance on Ballroom dataset.

	Classifier and features	Acc. % (SD)
Neural	**MCLNN (Shallow, $n = 20, k = 55$) + Mel-Spec. (this work)**	**92.55** (2.6)
	CNN + Mel-Spectrogram [66]	92.27
	LSTM + CNN + Mel-Spectrogram [67]	91.90 (2.3)
	MCLNN (Deep, $n = 15, k = 10$) + Mel-Spec. (this work)	90.40 (2.6)
	CNN + Mel-Spectrogram [23]	87.68 (4.4)
Non-Neural	SVM + 28 feature using known tempo annotations [16]	96.13
	KNN + Modulation Scale Spectrum [68]	93.12
	Manhattan Distance + Block-Level features [69]	92.44
	SVM + Rhyth., Hist., Statist., Onset, Symb. [15]	90.40
	KNN + 15 MFCC-like descriptors+Tempo [70]	90.10
	KNN + Rhythm and Timbre [71]	89.20
	SVM + 28 features without tempo annotations [16]	88.00
	SVM + Rhyth. + Hist. + Statist. features [13]	84.20
	KNN + Tempo [70]	82.30

Table 4
Performance on Homburg dataset.

	Classifier and features	Acc. % (SD)
Neural	**MCLNN (Deep, $n = 5, k = 2$) + Mel-Spec. (this work)**	**61.45** (1.4)
	KNN + mcRBM, PCA, MVG-MFCC [73]	55.30
	KNN + mcRBM, PCA, Mel-Spectrogram [74]	45.50
Non-Neural	JSLRR + Cortical, MFCC, Chroma [75]	63.46 (2.5)
	LRSM + Cortical, MFCC, Chroma [76]	62.40 (3.7)
	KNN + LFP, VDSP, CP, SCP [77]	61.20
	SVM + ESA-MFCC [78]	57.81
	KNN + Rhythm and Timbre [71]	57.00
	SVM + Marsyas features ([72]) [79]	55.00
	KNN + Multiple features [80]	53.23
	SVN + Novelty Functions [81]	51.10

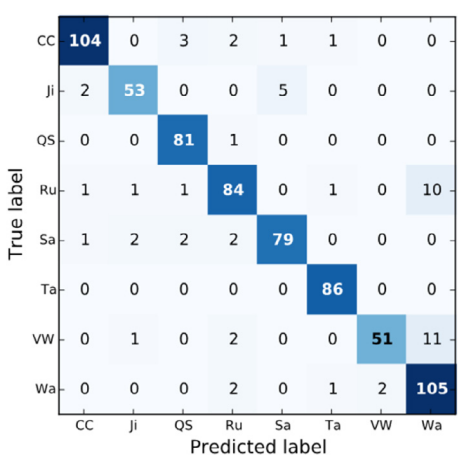

Fig. 9. Ballroom confusion using the MCLNN. Classes: ChaCha (CC), Jiva (Ji), QuickStep (QS), Rumba (Ru), Samba (Sa), Tango (Ta), Viennese Waltz (VW) and Slow Waltz (Wa)

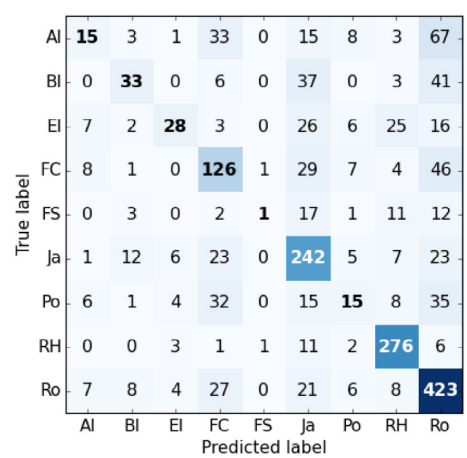

Fig. 10. Homburg confusion using the MCLNN. Classes: Alternative (Al), Blues (Bl), Electronic (El), FolkCountry (FC), FunkSoulRnb (FS), Jazz (Ja), Pop (Po), RapHiphop (RH) and Rock (Ro).

features. The MCLNN performed only slightly below these complicated handcrafted methods, achieving a mean accuracy of 61.45% without any special handling.

Fig. 10 shows the confusion across the genres of the Homburg dataset using the MCLNN. It is clear that less confusion occurs for genres having a higher number of samples, but in general, MCLNN achieved lower confusion than [80].

5.2.3. GTZAN

The dataset was introduced by Tzanetakis et al. [72]. It consists of 1000 music files categorized into 10 music genres (blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock) with 100 files per category. The files are 30 s in length each. The ISMIR2004 (discussed in the next section) and the GTZAN datasets are two of the oldest datasets that have been widely used in the literature with different experimental settings, especially with regard to the data split. Accordingly, through the experiments on the GTZAN and the ISMIR2004 we will evaluate: (a) the sustainability of the MCLNN performance against the influence of the data split, (b) the performance while constraining the training data, and (c) the generalization of a model across datasets having different distributions.

For both GTZAN and ISMIR2004, discussed later, we used an order $n = 4$ and one 50-neuron densely connected layer. The decision of a clip's category is based on majority voting to be comparable to the rest of the works reported in the literature. Table 5 lists the accuracies on the GTZAN. The MCLNN achieved a competitive accuracy compared to other neural network-based

Table 5
Performance on GTZAN dataset.

	Classifier and features	Acc.% (SD)
Neural	Two CNNs + Residual + (Spectrogram, Musical Features) [82][b]	91.00 (1.2)
	CNN + Residual + Spectrogram [83][b]	87.40
	MCLNN + Mel-Spectrogram (this work)[b]	**85.**10 (3.0)
	CNN + Spectrogram [83][b]	84.80
	RBF-SVM + Spectrogram – DBN [19][d]	84.30
	MCLNN + Mel-Spectrogram (this work)[c]	**84.10** (4.0)
	Random Forest + Spectrogram – DBN [84]	83.00 (1.1)
	Two CNNs + Mel-Spectrogram [85][e]	80.30 (2.9)
Non-Neural	Compressive Sampling + Multiple feature sets [86][b]	92.70
	SRC + LPNTF + Auditory Cortical features [87][b]	92.40 (2.0)
	RBF-SVM + Scattering Transform [88][b]	91.40 (2.2)
	Linear SVM + PSD on Octaves [89][c]	83.40 (3.1)
	AdaBoost + Several features [54][a]	83.00
	RBF SVM + Spectral Covariance [90][b]	81.00
	Linear SVM + PSD on Frames [89][c]	79.40 (2.8)
	SVM + DWCH [11][b]	78.50 (4.1)
	Logistic Regression + Spectral Covariance [90][b]	77.00
	LDA + MFCC, FFT, Beat and Pitch [12][b]	71.10 (7.3)
	GMM +MFCC, FFT, Rhythm and Pitch [72][b]	61.00 (4.0)

[a]5-fold cross-validation.
[b]10-fold cross-validation.
[c]10×10-fold cross-validation (i.e. 100 runs).
[d]50% training, 20% validation and 30% testing.
[e]4-fold cross-validation.

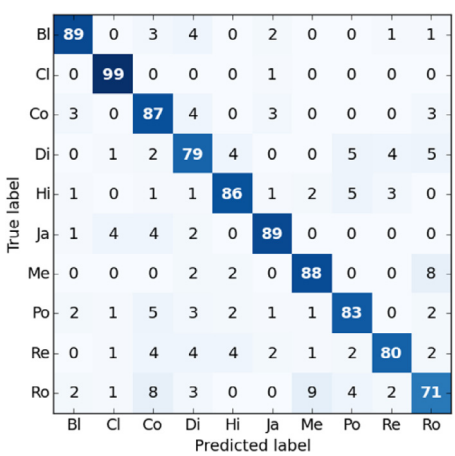

Fig. 11. GTZAN confusion using the MCLNN. Classes: Blues (Bl), Classical (Cl), Country (Co), Disco (Di), Hiphop (Hi), Jazz (Ja), Metal (Me), Pop (Po), Reggae (Re) and Rock (Ro).

attempts especially when details of their design are investigated. For example, the work Senac et al. [82] used a three-layered CNN of 256 filters each. The output of the third layer is fused with the output of the first layer through a residual connection that is later pooled with both max and mean pooling concurrently before feeding the pooling output to a densely connected network for the final classification. Senac proposed using two networks of such architecture, where one is presented with musically designed features and the second one is presented with a spectrogram. To achieve the 91% accuracy, Senac fused the output of these two networks using probability and majority voting following an aggregation stage. A similar architecture to Senac was proposed by Zhang et al. [83]. Zhang used a three-layered CNN with 256 filters each with a residual connection from the first layer to the output of the third layer. The Residual fusion is followed by mean and max pooling applied concurrently before a densely connected network. Zhang also proposed another architecture without the residual connection and with the convolutional layers interleaved with max-pooling layers. The residual network achieved 87.4% and on the other hand the non-residual variant achieved 84.8%, which shows that competitiveness of the MCLNN since it achieved 85.1% with a two-layered architecture without any special handling. Similarly, the work of Schindler et al. [85] achieved 80.3% accuracy using a two parallel CNNs of four layers each with a merging stage, where one is used to capture the frequency relations and another for the temporal. They reported an accuracy of 82.2% using the same architecture, but with pitch shifting and temporal stretching augmentations.

Table 6
GTZAN Random and Fault-Filtered accuracy using the splits by Kereliuk et al. [43].

	Random Acc. %	Filtered Acc. %
MCLNN (this work)	84.4	65.8
DNN [43]	81.2	49.0

The accuracy of 92.7% in [86] proposed using a set of 10 features designed to exploit long-time and short-time acoustic features for genre classification, e.g. octave-based spectral contrast, octave-based modulation spectral contrast, modulation spectral flatness measure, to name a few, in addition to MFCC, spectral flux and others. They also used a specially designed classifier based on Compressive Sampling. A comparable accuracy of 92.4% was also achieved in [87] with a similar complicated system using an auditory cortical representation that is dimensionally reduced using non-negative matrix factorization and finally classified using a sparse based classifier. The work in [88] achieved 91.4% using a specially designed signal transform that aims to provide a frequency shift-invariant representation of the signal. The accuracy reported in this method is also tightly coupled with the fine-tuning of the grid-search for the optimum RBF-SVM parameters. The MCLNN surpasses several state-of-the-art methods that are dependent on hand-crafted features or neural networks, achieving an accuracy of 85.1% over 10-fold cross-validation. Fig. 11 shows the confusion matrix for the proposed architecture on the GTZAN dataset, where the MCLNN classifies classical music with an accuracy of 99%. The lowest confusion was in the Rock category, which overlaps with GTZAN analysis by Sturm in [91].

Different experiment settings reveal the effect of the data split on the fluctuation in the accuracies reported in this work and other works in the literature. Considering the GTZAN dataset as an example, the dataset suffers analytical problems as discussed by Sturm [91], which retains a wide influence on reported accuracies that does not consider the data split. Referring to one of the highest accuracies reported with a neural network based architecture on the GTZAN by Hamel et al. [19] using a DBN for features extraction and an SVM for classification. Hamel used a fixed split (50% training, 20% validation and 30% testing) without applying cross-validation for the parameters search of the SVM. In an attempt to replicate the work by Hamel, Kereliuk et al. [43] used the same architecture achieving 81.5% compared to the 84.3% reported by Hamel, which could be the data split effect. Sigtia et al. [84] took the split influence into account, where they repeated the experiment for 4 times with a split of 50% training, 25% validation and 25% testing and they achieved 83% using their proposed approach. In a similar context related to the experimental settings, the work by Henaff et al. [89] reported an accuracy of 83.4%. In their work, they applied an unsupervised learning stage on the whole dataset rather than a training and test split, which falsely increases the final accuracy, as discussed by Henaff, since the model is aware of the test samples. Accordingly, they reapplied their approach to the training data only; the final accuracy they achieved was 80%.

To gauge the MCLNN sustainability against the data split influence, we repeated the 10-folds cross-validation for 10 times with the splits of the data randomized in each run. The MCLNN achieved 84.1% over 100 training runs (10×10-folds). As a further re-evaluation of the MCLNN with regard to the split, we adapted the publicly available splits released by Kereliuk et al. [43]. They released two versions of splits for the GTZAN files: The first split is a randomly stratified split with 50% training, 25% validation and 25% testing. The second version is a fault filtered version, where they cleared out all the mistakes in the GTZAN as reported by Sturm [91], e.g. repetitions, distortion, etc. Table 6 shows the outperformance of the MCLNN compared to the DNN used by Kereliuk for both the random and fault-filtered splits in their attempt to reproduce the work by Hamel.

We also investigated the effect of data size used in training. Referring to the works in [19,43,54,89,90], an FFT window of 1024 or 50 ms according to the sampling rate was used. On the other hand, we applied the experiments on both the GTZAN and the ISMIR2004 datasets using a 2048 FFT window, i.e. 100 ms, which decreases the number of feature vectors to be used in classification by 50% and consequently, the training complexity for larger datasets.

5.2.4. ISMIR2004

Released within the ISMIR music genre contest, the dataset comprises two splits of 729 files each. The splits have 6 unbalanced categories of music genres (classical, electronic, jazz-blues, metal-punk, rock-pop and world) of full-length recordings. Following [92] we extracted 30 s from each clip after the first 30 s of the clip. Further preprocessing followed the one applied for the GTZAN dataset.

We performed the experiments on the ISMIR2004 using the same model used for the GTZAN experiments without tuning any hyperparameter except for the batch size to measure the extent of the generalization of the model to different datasets possessing different distributions.

The first experiment followed the split proposed by the ISMIR genre classification contest using 729 files for training (we will refer to these files as the development set) and another 729 files for testing. We divided the 729 development files into randomly stratified splits of 90% training and 10% validation. To account for the effect of the training-validation split on the data, we repeated the experiment 10 times with randomly seeded generator. The second experiment involved combining the development and testing splits to form a dataset of 1458 files and following a 10-fold cross-validation scheme that is repeated for 10 times, as in the GTZAN.

Table 7 lists accuracies achieved by several methods including the MCLNN on the ISMIR2004. The mean accuracy across 10-folds is 86% using the MCLNN. The highest neural based accuracy of 87.5% was achieved using a combination of the scattering transform [88] and an LSTM. A comparable accuracy was achieved in [94] using two CNNs operating in parallel. Costa et al. [95] used a CNN with several handcrafted features to achieve 86.7%. Schindler et al. [85] achieved 85% using two CNNs and a Mel-Spectrogram as a signal representation. An MCLNN achieved 86% using a simple architecture without any special handling or handcrafted features.

With respect to handcrafted attempts, the highest, reported in [87], exploits psychophysiological properties of the human ear in addition to using a multilinear dimensionality reduction. Despite achieving 94.4% using a Sparse Representation Classifier, the method did not exceed an accuracy of 75% on using a Linear-SVM for classification. Other attempts such as [71] and [69] used rhythmic and hand-crafted features. On the other hand, there was no special handling for the signal used by the MCLNN. Fig. 12 shows the confusion matrix across the ISMIR2004 genres using the MCLNN for 10-fold cross-validation of the combined development and test sets.

5.2.5. ESC-10

The ESC-10 [98] dataset is composed of 10 categories of environmental sounds: Dog Bark, Rain, Sea Waves, Baby Cry, Clock Tick, Person Sneeze, Helicopter, Chainsaw, Rooster and Fire Cracking, evenly balanced with 40 sound files per category of 5 s each. The dataset is provided pre-distributed into 5-folds, which we used to avoid the influence of the data split. Following the MCLNN layers, two densely connected layers were used, with 100 neurons each. Table 8 reports the mean accuracy across the 5-folds of the ESC-10 dataset. The MCLNN achieved the highest accuracy of 85.5% and 83% at $k = 40$ and $k = 1$, respectively, compared to the CNN proposed in the work of Piczak [99] (Piczak-CNN). The Piczak-CNN is composed of two convolution and two pooling layers followed by two fully-connected layers for classification with a total of around 25 million trainable parameters. The Piczak-CNN used two different channels, one for the 60 bins Mel-scaled spectrogram and another for the Delta. We followed the same intermediate transformation used by Piczak (60 Mel-spec and Delta) to benchmark the performance of an MCLNN with respect to a CNN. It is also worth mentioning that the accuracy of 80% achieved by the Piczak-CNN used 10 variants of data augmentations, where he applied different time delays to each sound file. Therefore, the dataset is increased 10 times the original size, which increases the accuracy of the model as investigated in the work of Salamon et al. in [100]. In a different experiment, we applied 12 augmentations: 8 variants of pitch-shifts and 4 time-delays for each sound clip. The MCLNN achieved an accuracy of 85.25%, knowing that we did not increase the capacity of the trainable parameters to cope with the increase in the training data. Another consideration, is the computational complexity; The MCLNN used 12% of the parameters required by the Piczak-CNN, where the MCLNN used approximately 3 million parameters compared to the 25 million parameters used in the Piczak-CNN. The attempt of Aytar et al. [24] (SoundNet) achieved 82.3% using 5 convolutional layers interleaved with 3 max-pooling layers. The accuracy degraded to 75.5% when they increased the number of convolutional layers to 8, in addition to the 3 max-pooling

Table 7
Performance on ISMIR2004 dataset.

	Classifier and Features	Acc. % (SD)
Neural	LSTM + Scattering Transform [93]	87.50
	Two CNNs + Spectrogram, Harmonic and Percussion [94]	87.10
	CNN + SSD + RLBP [95]	86.70
	MCLNN + Mel-Spectrogram (this work)[d]	**86.04** (1.4)
	Two CNNs + Mel-Spectrogram [85][g]	85.18 (1.2)
	MCLNN + Mel-Spectrogram (this work)[e]	**84.83** (3.0)
	MCLNN + Mel-Spectrogram (this work)[a]	**84.77**
	MCLNN + Mel-Spectrogram (this work)[b]	**83.13** (1.5)
	DNN + Spectrogram [84]	73.47
Non-Neural	SRC + NTF + Auditory Cortical features [87][a]	94.38
	KNN + Rhythmic descriptors and timbre [71][d]	90.04
	SVM + Several Block-Level features [69][h]	88.27
	GMM + NMF [96][c]	83.50
	SVM + Audio and Symbolic features [15][d]	81.40
	Nearest Neighbour + Spec. Similarity and FP [14][f]	81.00
	SVM + High-Order SVD [97][d]	80.95 (3.3)
	SVM + Rhythmic Patterns and SSD [13][a]	79.70

[a]Train 729 file/test 729 file.
[b]10 × (Train 729 file/test 729 file).
[c]5-fold cross-validation.
[d]10-fold cross-validation.
[e]10 × 10-fold cross-validation (i.e. 100 runs).
[f]Leave-one-out cross-validation.
[g]4-fold cross-validation.
[h]Not referenced.

Table 8
Performance on ESC-10 dataset.

Classifier and features	Acc. % (SD)
MCLNN (layers = 2, $\boldsymbol{k = 40}$) + Mel-Spec. (this work)[b]	85.50 (4.9)
MCLNN (layers = 2, $\boldsymbol{k = 20}$) + Mel-Spec. (this work)[a]	85.25 (4.7)
MCLNN (layers = 2, $\boldsymbol{k = 1}$) + Mel-Spec. (this work)[b]	83.00 (4.0)
SoundNet (layers = 5) + Raw Waveform [24][b]	82.30
MCLNN (layers = 2, $\boldsymbol{k = 25}$) + Mel-Spec. (this work)[b]	82.00 (5.0)
Piczak-CNN (layers = 2) + Mel-Spectrogram [99][a]	80.00 (4.8)
CLNN (layers = 2, $\boldsymbol{k = 25}$) + Mel-Spec. (this work)[b]	77.50 (4.3)
CLNN (layers = 2, $\boldsymbol{k = 40}$) + Mel-Spec. (this work)[b]	75.75 (3.2)
SoundNet (layers = 8) + Raw Waveform [24][b]	75.50
CLNN (layers = 2, $\boldsymbol{k = 1}$) + Mel-Spec. (this work)[b]	73.25 (5.2)
Random Forest + MFCC [98][b]	72.70 (8.1)

[a]Augmentation.
[b]Without augmentation.

Table 9
Performance on ESC-50 dataset.

Classifier and features	Acc. % (SD)
MCLNN (layers = 1, $\boldsymbol{k = 40}$) + Mel-Spec. (this work)[a]	**66.60** (1.5)
SoundNet (layers = 5) + Raw Waveform [24][b]	65.00
Piczak-CNN (layers = 2) + Mel-Spec. [99][a]	64.50 (1.8)
MCLNN (layers = 1, $\boldsymbol{k = 40}$) + Mel-Spec. (this work)[b]	**62.85** (2.4)
L^3 -Net (layers = 8) + Log-Spec. [25][b]	62.50
MCLNN (layers = 2,$\boldsymbol{k = 5}$) + Mel-Spec. (this work)[b]	**61.75** (2.2)
SoundNet (layers = 8) + Raw Waveform [24][b]	51.10
Random Forest + MFCC [98][b]	44.00 (2.6)

[a]Augmentation.
[b]Without Augmentation.

layers, which is probably due to overfitting. The table also lists the accuracies achieved by a CLNN of the same hyperparameters of an MCLNN. The MCLNN surpasses the CLNN, which shows the effect of the masking operation enforced by the MCLNN. Fig. 13 shows the confusion across the ESC-10 sounds using the MCLNN; it is noticed that the highest confusion occurs in sounds of short events such as Clock Ticks getting confused with Fire Cracking. This confusion is also noticeable between sounds possessing low tonal components that resemble random noise as in Helicopter and Rain sounds.

5.2.6. ESC-50

The ESC-50 [98] dataset has a collection of 50 classes of environmental sounds released into 5-folds. It is the parent dataset of the ESC-10 dataset. We used two architectures of single and double layers having an order $n = 14$.

Table 9 lists the accuracies achieved on the ESC-50 dataset. The MCLNN achieved an accuracy of 66.6% using 4 augmentation variants composed of 2 pitch-shifts and 2 time-delays per clip using a single MCLNN layer of 1 million trainable parameters, and an accuracy of 62.85% on the original dataset without any augmentation with the same shallow architecture. The Piczak-CNN [99] achieved 64.5% using 4 augmentation variants and a model composed of 25 million trainable parameters. The SoundNet [24] achieved 65% and 51.1% using 5 and 8 convolutional layers, respectively. A randomly initialized L^3-Net [25], having a similar architecture to the SoundNet, was used to extract features to train a classifier. The 8 convolutional layers of the L^3-Net achieved 62.5%.

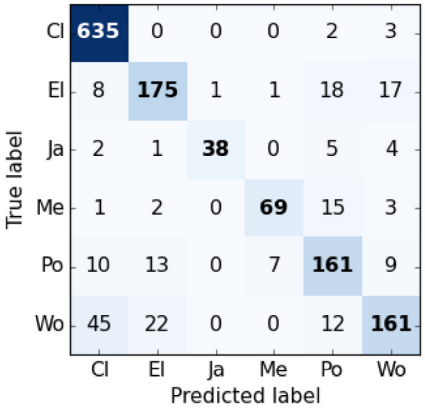

Fig. 12. ISMIR2004 confusion using the MCLNN. Classes: Classical (Cl), Electronic (El), Jazz/Blues (Ja), Metal/Punk (Me), Pop/Rock (Po) and World (Wo).

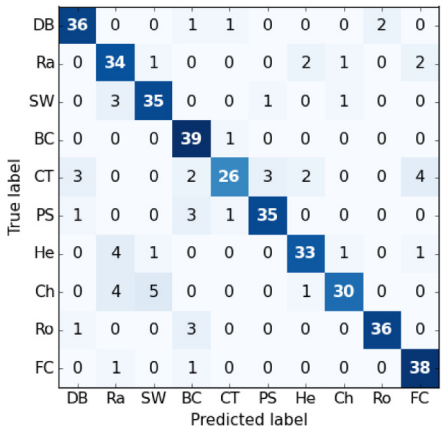

Fig. 13. ESC-10 confusion using the MCLNN. Classes: Dog Bark (DB), Rain (Ra), Sea Waves (SW), Baby Cry (BC), Clock Tick (CT), Person Sneeze (PS), Helicopter (He), Chainsaw (Ch), Rooster (Ro) and Fire Cracking (FC)

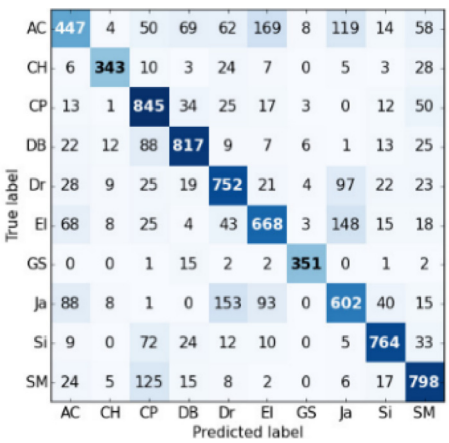

Fig. 14. Urbansound8k confusion using the MCLNN. Classes: Air Conditioner (AC), Car Horns (CH), Children Playing (CP), Dog Bark (DB), Drilling (Dr), Engine Idling (EI), Gun Shot (GS), Jackhammers (Ja), Siren (Si) and Street Music (SM)

A single or double layer of an MCLNN outperformed the deep architectures, composed of 5 to 8 convolutional layers in addition to the pooling layers, proposed in the SoundNet and the L^3-Net for either the ESC-10 and the ESC-50 dataset. It is worth mentioning that the work of Aytar et al. in *SoundNet* [24] proposed training the audio network using a visually pre-trained network. The "student–teacher" paradigm they proposed uses an image recognition network to extract features that influence the training of an audio network. They trained their architecture on 2 million videos (over one year of continuous sound and video), which showed an outperformance to the results in Tables 8 and 9. They experimented using deep architectures such as VGG [101] and AlexNet [38], each pretrained on two public datasets composed of millions of images (ImageNet is 1.2 million and Places is 10 million) to instruct the audio network while learning the audio channel of the 2 million videos. They further used the audio network to extract audio features that were classified using an SVM. The L^3-Net by Arandjelovic et al. [25] proposed a similar paradigm to the SoundNet, but they explored the performance of training both the visual and audio networks from scratch rather than using a pre-trained visual network as in the SoundNet. The L^3-Net showed an outperformance to the results in Table 9 when trained over 0.5 million videos.

The work reported here has used significantly smaller networks compared to such deep models so that the results in

Table 10
Performance on Urbansound8K dataset.

Classifier and features	Acc. % (SD)
MCLNN (Shallow, $k = 50$) + Mel-Spectrogram (this work)	**74.22** (6.5)
Random Forest + Spherical K-Means + PCA + Mel-Spec. [102]	73.70 (4.7)
MCLNN (Deep, $k = 5$) + Mel-Spectrogram (this work)	**73.28** (5.1)
Piczak-CNN + Mel-Spectrogram [99]	73.10 (4.7)
S&B-CNN + Mel-Spectrogram [100]	73.00 (5.5)
RBF-SVM + MFCC [103]	68.00 (4.1)

Tables 8 and 9 are genuine comparisons of the type of layers used and not of scale or the techniques utilized in training. Our aim in this paper is to show that the changes in structure that we have introduced yield significant performance increases without either hand-crafting or rescaling of networks. Future work will investigate extending the MCLNN layers to networks many times larger, getting the same scale advantages of training on millions of samples with a knowledge transfer from vision networks.

5.2.7. Urbansound8K

The dataset is composed of 8732 urban environment sound files of 4 s each: Air Conditioner, Car Horns, Children Playing, Dog Bark, Drilling, Engine Idling, Gun Shot, Jackhammers, Siren, and Street Music. The dataset is released into 10-folds, which we used to report the mean accuracy of the cross-validation using the MCLNN. A stride of two for the running window was used to extract the segments following (2). The MCLNN layers are followed by the two dense layers of 100 neurons each, as the ones used for the ESC-10.

Table 10 lists the accuracies achieved on the Urbansound8k dataset. A deep MCLNN with a small window achieved 73.3%, and a shallow one with a wider window achieved 74.2%, which is the highest accuracy achieved on the dataset using a neural-based architecture. The Piczak-CNN achieved 73.1% on the Urbansound8k. The MCLNN shallow and deep variants used 4% and 12%, respectively, of the parameters used by Piczak-CNN that used 25 million parameters. Salamon et al. [100] achieved 73% using a deeper CNN model compared to Piczak-CNN with even fewer parameters. Accordingly, we will consider deeper MCLNN architectures in future work. The highest non-neural network accuracy on the Urbansound8k is 73.7%. It was achieved by the unsupervised learning proposed by Salamon et al. in [102], which exploits the capabilities of Spherical K-Means as a clustering technique and Random Forest for classification in addition to the use of PCA for dimensionality reduction.

Fig. 14 shows the confusion matrix for the Urbansound8k using MCLNN. The highest confusion rate is among the Air Conditioner, Jackhammer, Drilling and Engine Idling. This is accounted for the low tonal properties that are common to these sounds as noted by Salamon et al. in [100].

5.3. Runtime analysis

In this section, we discuss the execution time of the MCLNN and its dependence on the hyperparameters.

Table 11 lists the trained values of the hyperparameters that have a direct effect on the execution time; other parameters that influence the training time but are fixed across all datasets are not included, e.g. learning rate.

The processed sound clips are transformed to spectrograms, these are of varying number of frames as listed in Table 11. Depending on the number of layers m, the order n and the extra frames k, the segment length presented to the MCLNN is decided. For example, a clip of the Ballroom dataset is 30 s, which generates a 600 frames spectrogram. A segment length based on the model's hyperparameters used for the Ballroom dataset

Table 11
Model parameters, processed input and runtime across different datasets.

Dataset	Clip length (Sec.)	Spectro. frames	m	n	k	q	q per clip	q per epoch (thousands)	Batch size	Param. (millions)	Epoch duration (s.)	Test clip inference (ms.)
Ballroom	30	600	1	20	55	96	505	280	600	2.5	240	140
Homburg	10	200	2	5	1	22	179	270	800	1	32	19
GTZAN	30	600	2	4	10	27	574	460	600	1	90	32
ISMIR	30	600	2	4	10	27	574	670	1,000	1	120	29
ESC-10	5	200	2	15	20	101	120	24	600	3	25	160
ESC-50	5	200	1	1	40	69	132	792	300	1	360	76
Urbansound8k	4	170	1	15	50	81	90	318	500	1.2	145	74

($m = 1$, $n = 20$ and $k = 55$) is composed of 96 frames. Since the number of frames per clip is 600 and the segment length is 96 frames with a stride of 1, the number of segments is 505. It can be inferred from these values that varying m, n and k has a direct effect on the number of training samples, which is also depicted in the table for other datasets. The table also lists the number of samples per epoch and the batch size per epoch, e.g. the Ballroom dataset has an average of 280,000 segments for training per epoch with a batch size of 500 samples. The number of segments may vary among the cross-validation folds in unbalanced datasets. The complexity of the model depends on the number of parameters, which directly influence the inference per clip at test time as listed in the table in milliseconds.

As shown in Table 11, the MCLNN provides competitive runtime performance on large datasets. Additionally, since an MCLNN layer resembles the structure of any feedforward network, methods used in online learning [104,105] can be incorporated with the help of a buffering mechanism to store the number of frames required for a segment of size q presented to the network during its temporal progression.

5.4. Hyperparameters analysis

This section provides an in-depth sensitivity analysis of the MCLNN. The experiments explore the effect of tuning different hyperparameters introduced through the MCLNN i.e. order n, overlap ov, bandwidth bw and extra frames k.

We used the ESC-10 dataset, because of its reasonable size, to investigate the effect of the mask and different hyperparameters introduced in the course of the CLNN and MCLNN architectures. We adapted the deep model referenced earlier in benchmarking the ESC-10 dataset as a baseline, and we gradually changed each parameter of the first layer in a 2-layered architecture, while fixing the other parameters to the baseline values without a special finetuning to the step value. All experiments for the upcoming analysis is based on the mean value of 5-folds cross-validation totaling to 300 runs (60 × 5-folds).

Fig. 15 shows the effect of the increasing the order n on the accuracy in percentage for the two-layered CLNN and MCLNN. The figure shows that on average the accuracy is directly proportional to the order n with regard to the MCLNN, but it then decreases beyond $n = 15$. This is explained by the increase in the number of neurons together with the decrease in the number of training samples with the increasing$n$. The plot also shows the effect of the masking operation in the MCLNN compared to the accuracies achieved by a CLNN. The masking operation in the MCLNN boosted the accuracy at different values of n accounting for the properties achieved by the mask as discussed previously.

Fig. 16 shows the effect of the aggregation operation over various k for both the CLNN and MCLNN. Still, the effect of the masking is clear in the accuracies of the MCLNN compared to the CLNN with a slight increase in the accuracy over the increasing k.

Figs. 17 and 18 show the effect of varying the Bandwidth and the Overlap of the first layer of the MCLNN. In Fig. 17, it is noticeable that increasing the bandwidth beyond $bw = 20$ causes

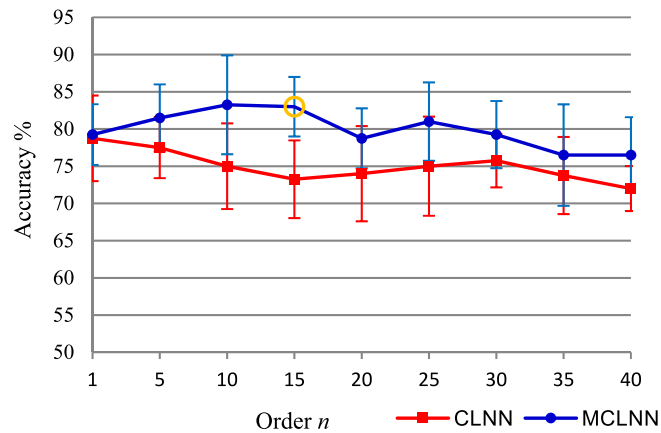

Fig. 15. Accuracy with varying the Order ***n*** (baseline circled)

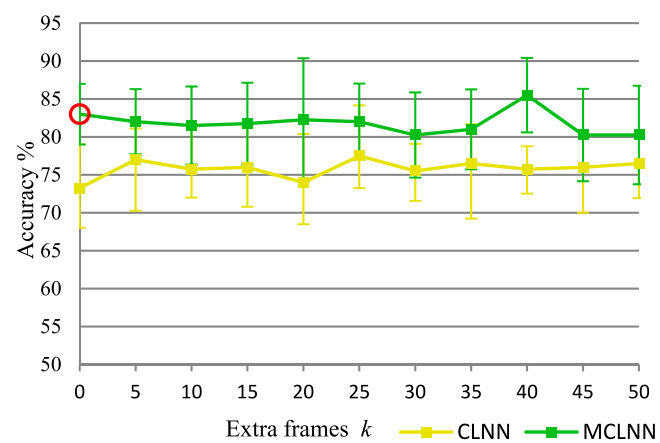

Fig. 16. Accuracy with varying Extra frames ***k*** (baseline circled)

a decrease in the accuracy this is accounted for by widening the scope of observation of a hidden node, which consequently prevents the node from learning about the distinctive features in a more focused region. On the other hand, decreasing the overlap in Fig. 18 using negative overlap values slightly increases the values with the increased sparseness, which suppresses the effect of the smearing of the energy across the frequency bins. This effect appears with the slight increase in the average accuracies of negative overlaps compared to positive ones.

5.5. Conditional and convolutional comparison

In this section, we evaluate the performance of the MCLNN, CNN, and LCN in isolation from architectural, signal representation and hardware influences. The experiments in this section are applied to all the datasets used in this work. This section is not seeking to find the optimum architecture, but rather to provide an

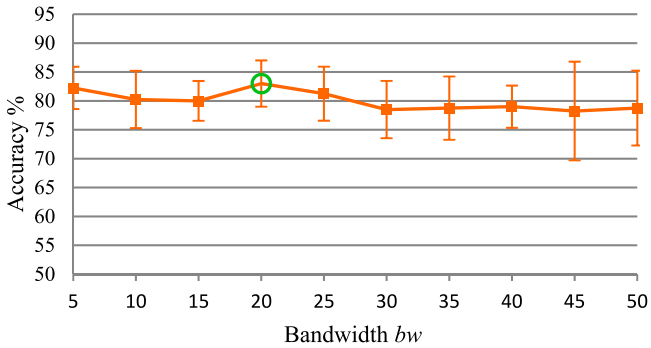

Fig. 17. Accuracy with varying the Bandwidth ***bw*** (baseline circled)

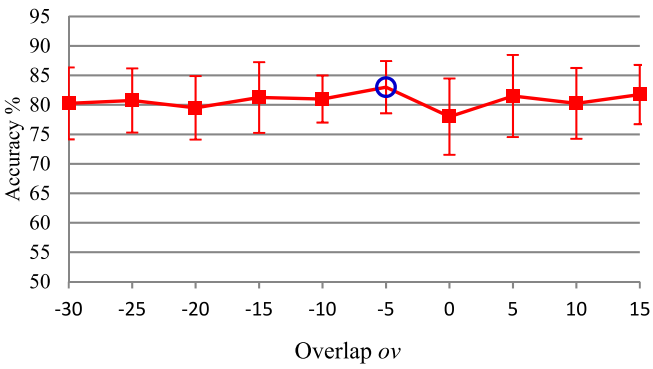

Fig. 18. Accuracy with varying the Overlap ***ov*** (baseline circled).

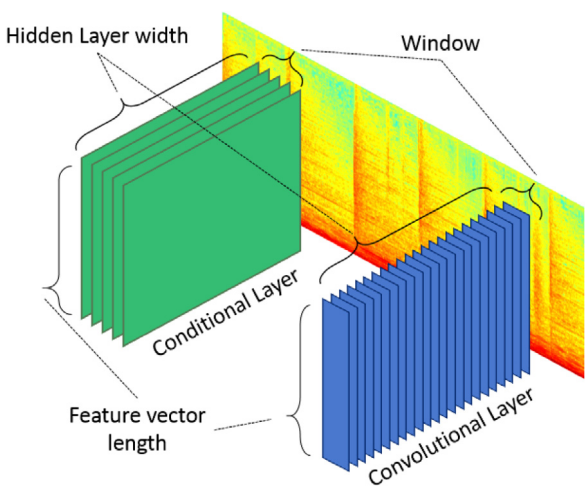

Fig. 19. Conditional weight matrices behavior scanning a spectrogram compared to the Convolutional filters.

unbiased comparison of the accuracies between layers of different structures.

Fig. 19 shows the CLNN compared to the CNN structure used in this section. The convolutional layer is composed of a set of filters each having a length matching the feature vector. The number of filters matches the number of hidden nodes used in an equivalent CLNN and the width of each CNN filter matches the window d of a CLNN. Both the CLNN and CNN use the same number of weights. However, their behavior in processing a spectrogram differs as shown in the figure. The masking of an MCLNN has no effect on the number of weights as it only exploits the underlying structure of a CLNN.

Table 12
Mean accuracy in % across 10/5-folds of datasets using shallow architectures of MCLNN, CNN, and LCN layers, along with the number of parameters to the nearest million.

	MCLNN	CNN	LCN	Parameter#	
				MCLNN, CNN	LCN
Ballroom	**83.1**	73.1	71.8	2	129
GTZAN	**83.2**	79.9	78.4	0.5	5.6
ISMIR	**83.5**	82.6	82.4	0.5	3
Homburg	**55.4**	54.9	54.4	0.6	1.2
ESC-10	**74.3**	69.5	70.3	1	57
ESC-50	**50.3**	41.1	41.3	1	42
Urbansound8k	**67.3**	60.5	59.9	1	57

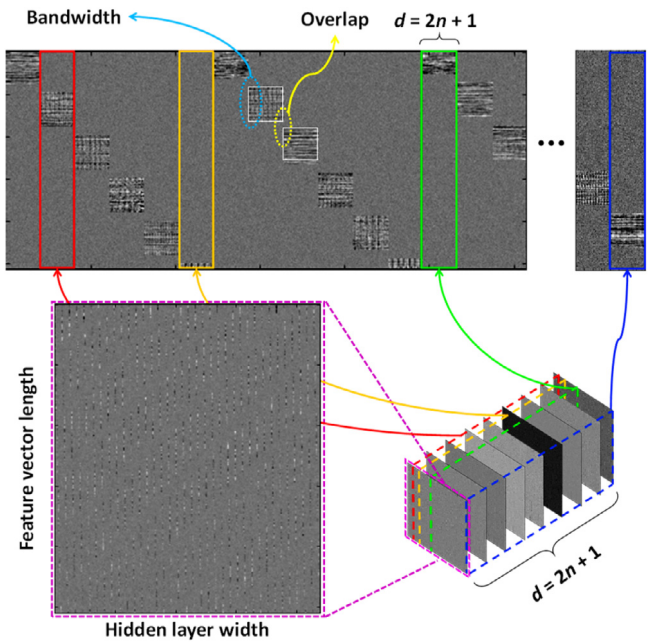

Fig. 20. Conditional weight matrices behavior scanning a spectrogram compared to the Convolutional filters.

We used the same spectrogram representation corresponding to each dataset as discussed previously. The models involve a shallow, single layer, architecture followed by a pooling layer with the same k and n values referenced earlier in the relevant section of each dataset with the absence of fully-connected layers. Thus, the output of the pooling layer is directly fed to a softmax layer. The mean accuracy of 10/5-folds cross-validation for each dataset is reported in Table 12.

The MCLNN achieved the highest accuracy across all the used datasets compared to the CNN and LCN. For example, with respect to the Ballroom dataset, an MCLNN achieved an accuracy of 83.1% compared an accuracy of 73.1% and 71.8% achieved by a CNN and LCN, respectively. The table also lists the number of weights used by each layer for different datasets. An LCN uses 129 million weights, compared to 2 million weights used by either an MCLNN or a CNN for the Ballroom dataset. Despite having the same number of weights for both the CNN and the MCLNN, the outperformance of the conditional layer of the MCLNN is accounted for allowing the independent training between the frames compared to the convolutional filters, since each frame has a dedicated weight matrix. The vector–matrix transformation, between an individual frame and the weight matrix, projects the frame in a different dimensional space matching the number of hidden nodes while preserving the projection spacing between successive projected vectors together with the selectivity applied over the features through the mask.

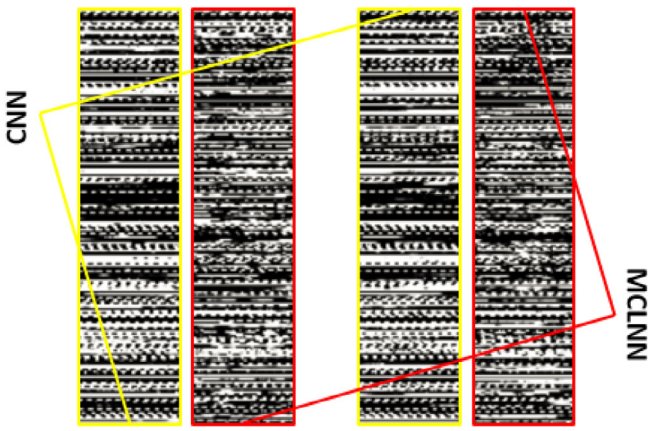

Fig. 21. Segments of the same sound file generated by the MCLNN and their corresponding generation of the CNN.

Fig. 20 shows the learned weight matrices of an MCLNN. The figure shows $d = 2n + 1$ matrices following the window size. The figure also includes a close-up view of the first weight matrix of the d set, which clearly shows a diagonal pattern following the masking pattern applied. A concatenation of d columns extracted from a cross-section of the d weight matrices forms a slice in the top section of the figure. The number of slices should match the number of hidden nodes in the hidden layer. The figure also depicts the width and the space shifts of the active weights controlled by the bandwidth and the overlap, respectively.

Fig. 21 illustrates the output generated from an MCLNN and the corresponding output for the same input segments from a CNN. The pattern fluctuations shown in the MCLNN segments compared to the repetitive patterns appearing in the CNN pose that some features are left out by the CNN, which could be distinguishing properties to be used in classification.

6. Discussion

The axes of the frequency domain representation of a temporal signal e.g. a spectrogram, have a different meaning compared to a two-dimensional representation of an image and the objects in it. Contrary, to images where an object exists as a co-located contiguous region of pixels, the frequencies of a single sound source are scattered across several frequency bins in a spectrogram. Moreover, the amplitude of energy of a frequency bin may arise from the contribution of more than one sound source overlapping in the acoustic scene. Convolutional models have been applied to both types of representations. However, their success in image pattern recognition surpasses that of spectrograms. The CNN filters operate as edge detectors in image processing. The relation between energies of different frequency bins of the same sound source is not enforced during training, but rather the network may capture it through deep layers of abstractions. On the other hand, the MCLNN, we are presenting in this work, controls which features can contribute to activating a hidden neuron following a pattern that resembles the scattering of energies across a spectrogram. Additionally, the multiple shifted patterns enforced by the mask allows for various features combinations to be considered concurrently, emulating the usual manual exploration operation. The spatial location of energy within a frequency bin is a distinctive feature for a specific sound, i.e. two different sounds sources may have the same amplitude of energy, but it is the positioning of this amplitude in a specific frequency bin and not another that defines these two sounds. The dense connections of an MCLNN preserve the spatial location ensuring that the positioning of the energies is considered during training. Furthermore, the dedicated weight matrix for each temporal stride allows for the independent training of weights per frame compared to convolving a window of frames in a CNN, which allows the weights to capture frame level perturbation at a higher resolution.

Training an MCLNN resembles training any other neural architecture. This involves tuning various hyperparameters in terms of the number of layers, the activation function, the regularization, the learning rate together with the optimizer,…etc. An MCLNN involves additional tunable hyperparameters. These are the order n, the extra frames k, the bandwidth bw and the overlap ov. Increasing the number of weights in a neural model can cause overfitting. In an MCLNN the number of weights is directly proportional with the order n. However, due to the systematic sparseness enforced by the mask, the bandwidth is the only region allowed to contribute to a hidden neuron's activation. Therefore, the probability of overfitting is lessened especially with the help of a strong regularizer such as dropout. The extra frames k controls the number of aggregated frames over a texture window [54]. Increasing k can downgrade the performance as it may eliminate distinguishable features, especially if mean pooling is used over a large segment. Also, it decreases the number of samples used for training. The bandwidth controls the number of active features at the hidden neuron and concurrently with the overlap, different shifted versions of a filterbank-like pattern are enforced over the input feature vector. Through experiments, it is recommended to use wider bandwidth at the lower layers of a deep MCLNN with narrower bandwidth for layers near the output. This allows more features to propagate through the network with the top layers learning about finer features structures over narrow bandwidth with possibly negative overlaps.

We evaluated the MCLNN using several publicly available datasets widely adopted in the literature for music genre classification and environmental sound recognition tasks. The datasets have different distributions, classes and sizes. However, the MCLNN architectures used among the datasets have minor differences in terms of the hyperparameters. Our experiments have shown that a single or double-layered MCLNN architecture achieves competitive performance compared to complicated deep structures of CNNs or hand-crafted features applied for sound recognition. Future work will consider multichannel temporal signals, this is motivated by the generalized structure of an MCLNN, which extends to the input standardization applied per feature compared to the per image standardization applied in CNN.

7. Conclusions and future work

The ConditionaL Neural Network (CLNN) provides a condition-based mechanism to exploit the sequential relationship across frames in a temporal signal. Additionally, extending from the CLNN, the Masked Conditional Neural Network (MCLNN) enforces a systematic sparseness that follows a frequency band-like pattern to exploit the non-contiguous distribution of energies in a time–frequency representation. Through an extensive set of experiments using publicly available datasets of music genre and environmental sounds, we have shown that the MCLNN, without exploiting any special rhythmic or timbral properties, sustains accuracies that surpass existing neural networks-based approaches and often outperform hand-crafted feature extraction approaches too. Meanwhile, MCLNN still preserves the generalization that allows it to be adapted for any other multi-dimensional temporal representations. Future work will include optimizing the mask patterns, considering different combinations of the order across the layers, and using the MCLNN as a stand-alone feature extractor for other pattern analysis tasks with the help of deeper

architectures. We will also consider applying the MCLNN to other signals possessing a temporal nature such as EEG and EMG. Similar to a CRNN, merging the MCLNN to extract the local feature with the long-term dependencies captured by an LSTM will be explored.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2020.106073.

Acknowledgment

This work is funded by the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 608014 (CAPACITIE).

References

[1] K.H. Davis, R. Biddulph, S. Balashek, Automatic recognition of spoken digits, J. Acoust. Soc. Am. 24 (1952) 637–642.
[2] M.A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, M. Slaney, Content-based music information retrieval: Current directions and future challenges, Proc. IEEE 96 (2008) 668–696.
[3] A. Dufaux, L. Besacier, M. Ansorge, F. Pellandini, Automatic sound detection and recognition for noisy environment, in: European Signal Processing Conference, EUSIPCO, 2000.
[4] M.S. Hammer, T.K. Swinburn, R.L. Neitzel, Environmental noise pollution in the United States: Developing an effective public health response, Environ. Health Perspect. 122 (2014) 115–119.
[5] The European Parliament and of the Council, Directive 2002/49/EC of the European Parliament and of the Council Relating To the Assessment and Management of Environmental Noise, 2002.
[6] M. Cristani, M. Bicego, V. Murino, Audio-visual event recognition in surveillance video sequences, IEEE Trans. Multimed. 9 (2007) 257–267.
[7] D. Chesmore, J. Schofield, Acoustic detection of regulated pests in hardwood material, Eur. Mediterr. Plant Prot. Organ. Bull. 40 (2010) 46–51.
[8] M. Popescu, A. Mahnot, Acoustic fall detection using one-class classifiers, in: Annual International Conference of the Engineering in Medicine and Biology Society, EMBC, 2009, pp. 2–6.
[9] H. Soltau, T. Schultz, MartinWestphal, A. Waibel, Recognition of music types, in: International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 1998.
[10] S. Dieleman, B. Schrauwen, End-to-end learning for music audio, in: International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2014.
[11] T. Li, M. Ogihara, Q. Li, A comparative study on content-based music genre classification, in: ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR, 2003.
[12] T. Li, G. Tzanetakis, Factors in automatic musical genre classification of audio signals, in: IEEE workshop on Applications of Signal Processing to Audio and Acoustics, 2003.
[13] T. Lidy, A. Rauber, Evaluation of feature extractors and psycho-acoustic transformations for music genre classification, in: International Conference on Music Information Retrieval, ISMIR, 2005.
[14] E. Pampalk, A. Flexer, G. Widmer, Improvements of audio-based music similarity and genre classification, in: International Conference on Music Information Retrieval, ISMIR, 2005.
[15] T. Lidy, A. Rauber, A. Pertusa, J.M. Inesta, Improving genre classification by combination of audio and symbolic descriptors using a transcription system, in: International Conference on Music Information Retrieval, 2007.
[16] G. Peeters, Spectral and temporal periodicity representations of rhythm for the automatic classification of music audio signal, IEEE Trans. Audio Speech Lang. Process. 19 (2011) 1242–1252.
[17] S.E. Fahlman, G.E. Hinton, T.J. Sejnowski, Massively parallel architectures for al: Netl, thistle, and Boltzmann machines, in: National Conference on Artificial Intelligence, AAAI, 1983.
[18] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (2006) 504–507.
[19] P. Hamel, D. Eck, Learning features from music audio with deep belief networks, in: International Society for Music Information Retrieval Conference, ISMIR, 2010.
[20] E. Cakir, T. Heittola, H. Huttunen, T. Virtanen, Polyphonic sound event detection using multi label deep neural networks, in: International Joint Conference on Neural Networks, IJCNN, 2015, pp. 1–7.
[21] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.
[22] A.v d. Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in: Neural Information Processing Systems, NIPS, 2013.
[23] J. Pons, T. Lidy, X. Serra, Experimenting with musically motivated convolutional neural networks, in: International Workshop on Content-based Multimedia Indexing, CBMI, 2016.
[24] Y. Aytar, C. Vondrick, A. Torralba, Soundnet: Learning sound representations from unlabeled video, in: Neural Information Processing Systems, NIPS, 2016.
[25] R. Arandjelovic, A. Zisserman, Look, listen and learn, in: IEEE International Conference on Computer Vision, ICCV, 2017.
[26] A.V. Oppenheim, A.S. Willsky, S.H. Nawab, Signals & Systems, Prentice-Hall, USA, 1996.
[27] X. Li, L. Girin, R. Horaud, Expectation-maximization for speech source separation using convolutive transfer function, CAAI Trans. Intell. Technol. 4 (2019) 47–53.
[28] J.J. Aucouturier, B. Defreville, F. Pachet, The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music, J. Acoust. Soc. Am. 122 (2007) 881–891.
[29] F. Medhat, D. Chesmore, J. Robinson, Masked conditional neural networks for audio classification, in: International Conference on Artificial Neural Networks, ICANN, 2017, pp. 349–358.
[30] F. Medhat, D. Chesmore, J. Robinson, Automatic classification of music genre using masked conditional neural networks, in: IEEE International Conference on Data Mining, ICDM, 2017, pp. 979–984.
[31] F. Medhat, D. Chesmore, J. Robinson, Environmental sound recognition using masked conditional neural networks, in: International Conference on Advanced Data Mining and Applications, ADMA, 2017, pp. 373–385.
[32] F. Medhat, D. Chesmore, J. Robinson, Music genre classification using masked conditional neural networks, in: International Conference on Neural Information Processing, ICONIP, 2017, pp. 470–481.
[33] F. Medhat, D. Chesmore, J. Robinson, Masked conditional neural networks for environmental sound classification, in: SGAI International Conference on Artificial Intelligence, AI, 2017, pp. 21–33.
[34] F. Medhat, D. Chesmore, J. Robinson, Recognition of acoustic events using masked conditional neural networks, in: IEEE International Conference On Machine Learning And Applications, ICMLA, 2017.
[35] F. Medhat, D. Chesmore, J. Robinson, Masked conditional neural networks for automatic sound events recognition, in: IEEE International Conference on Data Science and Advanced Analytics, DSAA, 2017.
[36] G.W. Taylor, G.E. Hinton, S. Roweis, Modeling human motion using binary latent variables, in: Advances in Neural Information Processing Systems, NIPS, 2006, pp. 1345–1352.
[37] A.-R. Mohamed, G. Hinton, Phone recognition using restricted Boltzmann machines in: IEEE International Conference on Acoustics Speech and Signal Processing, ICASSP, 2010.
[38] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Neural Information Processing Systems, NIPS, 2012.
[39] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2015, pp. 1–9.
[40] M. Sajid, N. Ali, S.H. Dar, N. Iqbal Ratyal, A.R. Butt, B. Zafar, et al., Data augmentation-assisted makeup-invariant face recognition, Math. Probl. Eng. 2018 (2018) 1–10.
[41] S. Sun, Y. Yin, X. Wang, D. Xu, W. Wu, Q. Gu, Fast object detection based on binary deep convolution neural networks, CAAI Trans. Intell. Technol. 3 (2018) 191–197.
[42] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, D. Yu, Convolutional neural networks for speech recognition, IEEE/ACM Trans. Audio Speech Lang. Process. 22 (2014) 1533–1545.
[43] C. Kereliuk, B.L. Sturm, J. Larsen, Deep learning and music adversaries, IEEE Trans. Multimed. 17 (2015) 2059–2071.
[44] P. Barros, C. Weber, S. Wermter, Learning auditory neural representations for emotion recognition, in: IEEE International Joint Conference on Neural Networks, IJCNN/WCCI, 2016.
[45] L. Wyse, Audio spectrogram representations for processing with convolutional neural networks, in: International Workshop on Deep Learning and Music, 2017.
[46] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780.
[47] S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 06 (1998) 107–116.
[48] A. Graves, J. Schmidhuber, Offline handwriting recognition with multidimensional recurrent neural networks, in: Presented at the Advances in Neural Information Processing Systems, NIPS, 2009.

[49] A. Graves, A.-r. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2013.
[50] K. Choi, G. Fazekas, M. Sandler, K. Cho, Convolutional recurrent neural networks for music classification, 2016, arXiv preprint arXiv:1609.04243.
[51] H. Lee, R. Grosse, R. Ranganath, A.Y. Ng, Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: Proceedings of the 26th Annual International Conference on Machine Learning, ICML, 2009, pp. 1–8.
[52] H. Lee, Y. Largman, P. Pham, A.Y. Ng, Unsupervised feature learning for audio classification using convolutional deep belief networks, in: Neural Information Processing Systems, NIPS, 2009.
[53] M. Lin, Q. Chen, S. Yan, Network in network, in: International Conference on Learning Representations, ICLR, 2014.
[54] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, B. Kégl, Aggregate features and adaboost for music classification, Mach. Learn. 65 (2006) 473–484.
[55] Y.-h. Chen, I. Lopez-Moreno, T.N. Sainath, M. Visontai, R. Alvarez, C. Parada, Locally-connected and convolutional neural networks for small footprint speaker recognition, in: INTERSPEECH, 2015.
[56] M.-A. Carbonneau, V. Cheplygina, E. Granger, G. Gagnon, Multiple instance learning: A survey of problem characteristics and applications, Pattern Recognit. 77 (2018) 329–353.
[57] D. Kingma, J. Ba, ADAM: A method for stochastic optimization, in: International Conference for Learning Representations, ICLR, 2015.
[58] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, J. Mach. Learn. Res. (2011).
[59] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on imagenet classification, in: IEEE International Conference on Computer Vision, ICCV, 2015.
[60] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[61] R. Al-Rfou, G. Alain, A. Almahairi, et al., Theano: A python framework for fast computation of mathematical expressions, 2016, arXiv e-prints, arXiv:abs/1605.02688.
[62] F. Chollet, Keras, 2015, Available: https://github.com/fchollet/keras.
[63] FFmpeg Developers, Ffmpeg, 2016, Available: http://ffmpeg.org/.
[64] M. McVicar, C. Raffel, D. Liang, O. Nieto, E. Battenberg, J. Moore, et al., Librosa, 2016, Available: https://github.com/librosa/librosa.
[65] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, et al., An experimental comparison of audio tempo induction algorithms, IEEE Trans. Audio Speech Lang. Process. 14 (2006) 1832–1844.
[66] J. Pons, X. Serra, Designing efficient architectures for modeling temporal features with convolutional neural networks, in: International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 2017.
[67] N. Chen, S. Wang, High-level music descriptor extraction algorithm based on combination of multi-channel CNNs and LSTM, in: International Society for Music Information Retrieval Conference, ISMIR, 2017.
[68] U. March, G. Peeters, The modulation scale spectrum and its application to rhythm-content description, in: International Conference on Digital Audio Effects, DAFx, 2014.
[69] K. Seyerlehner, M. Schedl, T. Pohle, P. Knees, Using block-level features for genre classification, tag classification and music similarity estimation, in: Music Information Retrieval EXchange, MIREX, 2010.
[70] F. Gouyon, S. Dixon, E. Pampalk, G. Widmer, Evaluating rhythmic descriptors for musical genre classification, in: International AES Conference, 2004.
[71] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, G. Widmer, On rhythm and general music similarity, in: International Society for Music Information Retrieval, ISMIR, 2009.
[72] G. Tzanetakis, P. Cook, Musical genre classification of audio signals, IEEE Trans. Speech Audio Process. 10 (2002).
[73] C. Osendorfer, J. Schluter, J. Schmidhuber, P. v. d. Smagt, Unsupervised learning of low-level audio features for music similarity estimation, in: Workshop on Speech and Visual Information Processing in conjunction with the International Conference on Machine Learning, ICML, 2011.
[74] J. Schluter, C. Osendorfer, Music similarity estimation with the mean-covariance restricted boltzmann machine, in: International Conference on Machine Learning and Applications, ICMLA, 2011, pp. 118–123.
[75] Y. Panagakis, C.L. Kotropoulos, G.R. Arce, Music genre classification via joint sparse low-rank representation of audio features, IEEE/ACM Trans. Audio Speech Lang. Process. 22 (2014) 1905–1917.
[76] Y. Panagakis, C. Kotropoulos, Music classification by low-rank semantic mappings, EURASIP J. Audio Speech Music Process. (2013).
[77] K. Seyerlehner, G. Widmer, Fusing block-level features for music similarity estimation, in: International Conference on Digital Audio Effects, DAFx-10, 2010.
[78] K. Aryafar, A. Shokoufandeh, Music genre classification using explicit semantic analysis, in: International ACM workshop on Music Information Retrieval With User-Centered and Multimodal Strategies, MIRUM, 2011.
[79] F. Moerchen, I. Mierswa, A. Ultsch, Understandable models of music collections based on exhaustive feature generation with temporal statistics, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD, 2006.
[80] H. Homburg, I. Mierswa, B. Moller, K. Morik, M. Wurst, A benchmark dataset for audio classification and clustering, in: International Symposium on Music Information Retrieval, 2005.
[81] A. Lykartsis, A. Lerch, Beat histogram features for rhythm-based musical genre classification using multiple novelty functions, in: Conference on Digital Audio Effects, DAFx-15, 2015.
[82] C. Senac, T. Pellegrini, F. Mouret, J. Pinquier, Music feature maps with convolutional neural networks for music genre classification, in: International Workshop on Content-Based Multimedia Indexing, CBMI, 2017, pp. 1–5.
[83] W. Zhang, W. Lei, X. Xu, X. Xing, Mproved music genre classification with convolutional neural networks, in: Interspeech, 2016, pp. 3304–3308.
[84] S. Sigtia, S. Dixon, Improved music feature learning with deep neural networks, in: International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 2014.
[85] A. Schindler, T. Lidy, A. Rauber, Comparing shallow versus deep neural network architectures for automatic music genre classification, in: Forum Media Technology, FMT, 2016.
[86] K.K. Chang, J.-S.R. Jang, C.S. Iliopoulos, Music genre classification via compressive sampling, in International Society for Music Information Retrieval, ISMIR, 2010.
[87] Y. Panagakis, C. Kotropoulos, G.R. Arce, Music genre classification using locality preserving non-negative tensor factorization and sparse representations, in: International Society for Music Information Retrieval Conference, ISMIR, 2009.
[88] J. Anden, S. Mallat, Deep scattering spectrum, IEEE Trans. Signal Process. 62 (2014) 4114–4128.
[89] M. Henaff, K. Jarrett, K. Kavukcuoglu, Y. LeCun, Unsupervised learning of sparse features for scalable audio classification, in: International Society for Music Information Retrieval, ISMIR, 2011.
[90] J. Bergstra, M. Mandel, D. Eck, Scalable genre and tag prediction with spectral covariance, in: International Society for Music Information Retrieval, ISMIR, 2010.
[91] B.L. Sturm, The state of the art ten years after a state of the art: Future research in music information retrieval, J. New Music Res. 43 (2014) 147–172.
[92] Y. Panagakis, C. Kotropoulos, G.R. Arce, Non-negative multilinear principal component analysis of auditory temporal modulations for music genre classification, IEEE Trans. Audio Speech Lang. Process. 18 (2010) 576–588.
[93] J. Dai, S. Liang, W. Xue, C. Ni, W. Liu, Long short-term memory recurrent neural network based segment features for music genre classification, in: International Symposium on Chinese Spoken Language Processing, ISCSLP, 2016.
[94] L. Nanni, Y.M.G. Costa, R.L. Aguiar, C.N. Silla, S. Brahnam, Ensemble of deep learning visual and acoustic features for music genre classification, J. New Music Res. 47 (2018) 383–397.
[95] Y.M.G. Costa, L.S. Oliveira, C.N. Silla, An evaluation of convolutional neural networks for music classification using spectrograms, Appl. Soft Comput. 52 (2017) 28–38.
[96] A. Holzapfel, Y. Stylianou, Musical genre classification using nonnegative matrix factorization-based features, IEEE Trans. Audio Speech Lang. Process. 16 (2008) 424–434.
[97] I. Panagakis, E. Benetos, C. Kotropoulos, Music genre classification: A multilinear approach, in: International Society for Music Information Retrieval, ISMIR, 2008.
[98] K.J. Piczak, ESC: Dataset for environmental sound classification, in: ACM International Conference on Multimedia 2015, pp. 1015–1018.
[99] K.J. Piczak, Environmental sound classification with convolutional neural networks, in: IEEE International Workshop on Machine Learning for Signal Processing, MLSP, 2015.
[100] J. Salamon, J.P. Bello, Deep convolutional neural networks and data augmentation for environmental sound classification, IEEE Signal Process. Lett. (2016).
[101] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations, ICLR, 2015.
[102] J. Salamon, J.P. Bello, Unsupervised feature learning for urban sound classification, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 2015..
[103] J. Salamon, C. Jacoby, J.P. Bello, A dataset and taxonomy for urban sound research, in: Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, USA, 2014, pp. 1041–1044.
[104] N.Y. Liang, G.B. Huang, P. Saratchandran, N. Sundararajan, A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Trans. Neural Netw. 17 (2006) 1411–1423.
[105] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradientmethods for online learning and stochastic optimization, J. Mach. Learn. Res. 12 (2011) 2121–2159.