Real-Time Monophonic and Polyphonic Audio Classification from Power Spectra

Maxime Baelde, Christophe Biernacki, Raphaël Greff

▶ **To cite this version:**

Maxime Baelde, Christophe Biernacki, Raphaël Greff. Real-Time Monophonic and Polyphonic Audio Classification from Power Spectra. 2018. hal-01834221v1

HAL Id: hal-01834221

https://hal.science/hal-01834221v1

Preprint submitted on 10 Jul 2018 (v1), last revised 11 Mar 2019 (v3)

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

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

Real-Time Monophonic and Polyphonic Audio Classification from Power Spectra

Maxime Baelde[a,b,*], Christophe Biernacki[b], Raphaël Greff[a]

[a] *A-Volute, 19 rue de la Ladrié, 59491 Villeneuve d'Ascq, France*
[b] *Inria, Univ. Lille, CNRS, UMR 8524 - Laboratoire Paul Painlevé, F-59000*

Abstract

This work addresses the recurring challenge of real-time monophonic and polyphonic audio source classification. The whole power spectrum is directly involved in the proposed process, avoiding complex and hazardous traditional feature extraction. It is also a natural candidate for polyphonic events thanks to its additive property in such cases. The classification task is performed through a nonparametric kernel-based generative modeling of the power spectrum. Advantage of this model is twofold: it is almost hypothesis free and it allows to straightforwardly obtain the *maximum a posteriori* classification rule of online signals. Moreover it makes use of the monophonic dataset to build the polyphonic one. Then, to reach the real-time target, the complexity of the method can be tuned by using a standard hierarchical clustering preprocessing of sound models, revealing a particularly efficient computation time and classification accuracy trade-off. Finally, it is shown that the resulting real-time audio classification method outperforms the state-of-the-art methods in the monophonic and polyphonic cases on benchmark and owned datasets, even in real-time situation.

Keywords: real-time, audio classification, machine learning, monophonic, polyphonic, generative model, nonparametric estimation.

1. Introduction

Audio source classification has been a challenging research subject for the past thirty years, beginning with speech recognition [1], and currently known as the vast field of *sound event detection* (SED). The latter consists in detecting and classifying audio sources present in *monophonic* (one source active at a time) and *polyphonic* (several sources at a time) audio streams. Many methodologies and algorithms were created to solve SED, and can be distributed in three

*Corresponding author
Email addresses: maxime.baelde@a-volute.com (Maxime Baelde), christophe.biernacki@math.univ-lille1.fr (Christophe Biernacki), raphael.greff@a-volute.com (Raphaël Greff)
URL: http://rare.lille.inria.fr (Maxime Baelde)

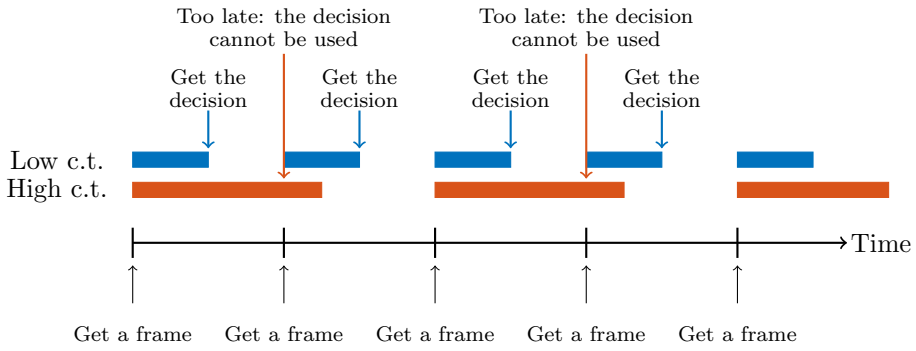

Figure 1: Illustration of the latency of a real-time audio classification system. A real-time process gets the frames at specific time clocks (black bottom arrows). If the computation time (c.t.) is low (Low c.t., blue filled rectangles), the decision will be used by the system because it will be available before the next frame. If the computation time is high (High c.t., red filled rectangles), the decision is discarded because it cannot be used by the system.

main topics. The most prominent topic is automatic speech recognition (ASR) whose goal is to identify speech (in particular phonemes) in audio recording [2]. The next topic is music information retrieval aiming at analyzing musics and extracting relevant information such as the musical genre [3] (rock, classical, *etc.*) or the different instruments [4]. The last topic is environmental sound recognition which aims at recognizing sounds such as airplanes, dog coughs, trains, gunshots, *etc.* [5]. SED can be performed offline – using the whole signal – or online – audio data come on the fly as *time frames*. Online or *real-time* processing relates to two criteria [6]: *speed* and *latency*. First, speed is the time to make the decision and is related to how many time frames the system uses. Second, latency is related to the computation time. For instance, if a time frame lasts 50ms, the decision has to be made within these 50ms, otherwise the result will not be used in the process. The latter is illustrated on Figure 1.

State-of-the-art SED methods typically involve two steps: *feature extraction* and *supervised learning*. Feature extraction – a universal stage in Machine Learning – summarizes the available information to a set of (expected) discriminant features. The usual audio features are known as *audio descriptors* [7], distributed in three groups. Temporal features use the raw audio signal (as a function of time) and consist in the energy, the autocorrelation coefficients and the zero crossing rate (*i.e.* how many times the signal crosses zero) for instance. Spectral features are extracted using the Fourier Transform (FT) and are mainly the spectral moments (centroid, spread, skewness, kurtosis). Cepstral and perceptual features are computed using the inverse FT of the log-magnitude FT on Mel-scale (for the Mel Frequency Cepstral Coefficient (MFCC)), and a harmonic decomposition (for the fundamental frequency, the inharmonicity, *etc.*), respectively. The relevance of the features depends on the context: for instance, the MFCC are good features for glass break, but not for gun shot recognition [8].

Several supervised learning methods have been applied to monophonic SED using the previous features. Monophonic SED corresponds to a multi-class single-label classification. First, Gaussian Mixture Models (GMM) are generative models assuming that the distribution generating the data given a class is a mixture of Gaussian distributions. The decision is computed using the *maximum a posteriori* (MAP), which is the maximum posterior probability of the classes. This modeling has been applied with MFCC for gunshot detection [9, 10] and real-time voice detection in medical application [11, 12]. Second, Hidden Markov Models (HMM) are used to model temporal continuity of audio signals, alone or coupled with GMM. Bietti *et al.* modeled the normalized audio spectra using HMM in an online setting [13], whereas Heittola *et al.* fed a HMM with histograms of MFCC [14]. Third, Support Vector Machines (SVM) are binary classifiers that map the features into a high dimensional space using the kernel trick and perform the classification in this space. The SVM can be extended to multi-class classification by considering learning strategies like One-Versus-One or One-Versus-All. SVM have been used with the energy and the signal spectrogram in [15] and [16] for surveillance application. Finally, Neural Networks (NN) – and their deep variants such as Deep Convolutional/Recurrent Neural Networks (DC/RNN) – have recently focused the attention of researchers. Biondi *et al.* used a normalized spectrum as input for a standard NN [17] and Dadula *et al.* considered the MFCC [18]. Palaz *et al.* [19] and Piczak [5] considered directly the raw audio signal and the spectrogram, respectively, using a deep architecture (DCNN).

Other algorithms are used in case of polyphonic SED. Polyphonic SED corresponds to a multi-class multi-label classification. It is often cast to a multi-class single-label classification but the output of the system is thresholded to predict the active classes. Çakır *et al.* [20] developed a Convolutional Recurrent Neural Network (CRNN), taking advantages of the two structures (convolutional and recurrent). However, NN-based algorithms require a huge amount of labeled data to train the network, which is not always available (no public dataset or time-consuming data recording and labeling). Apart from NN, two main modeling are considered: PLCA (Probabilistic Latent Component Analysis) [21] and NMF (Nonnegative Matrix Factorization) [22]. Benetos *et al.* [23] proposed a probabilistic modeling of ERB (Equivalent Rectangular Bandwidth) spectra using PLCA coupled with HMM. NMF has been used as a task-driven modeling of MFCC spectra [24] or as coupled training of sound spectra and class annotation [25]. Heittola *et al.* [26] constructed a two-step method based on unsupervised source separation (with NMF) and classification of the separated sources (using a GMM-HMM).

The previous review arouses three unresolved problems. First, the methods rely on context-based (and not always relevant) features. Second, the algorithms are not often designed for real-time processing: either they need lots of frames (low speed) or the computations are too heavy (high latency). Third, typical methods for polyphonic SED suffer from three drawbacks: the number of active sources is assumed to be known, the output of the system is thresholded, and a dataset

of sound mixtures is needed. Even for general multi-label learning task, usual methods include Binary Relevance (learn a classifier for each label individually: simple but does not learn correlations between labels) or Label Powerset (consider multi-label output as a new single-label: complex and combinatorial) [27], which are far from optimal solutions. To overcome the previous problems, a novel method is proposed in this paper that can perform both monophonic and polyphonic real-time SED without assuming the number of active sources to be known and by using the audio spectrum itself (not audio descriptors) for the classification.

The method developed in this paper is based on a generative model of the whole power spectrum, releasing the need of (sometimes perilous) feature extraction. In particular, the use of the power spectrum instead of standard magnitude spectrum is useful for the polyphonic modeling task thanks to the additive property of uncorrelated signals. Consequently, a suitable decomposition of the polyphonic spectra using monophonic ones allows to dispense with learning mixture of sounds, which is not possible with classical predictive modeling. In addition, the generative model has two advantages. Firstly, it can be considered as a very low assumption situation since it is related to a kernel density estimation using multinomial kernels (nonparametric framework). Secondly, it allows to straightforwardly derive a temporal MAP for the online classification. However, using the model as it, the real-time target is not reached because of the involved computational load. A model preprocessing using hierarchical clustering is thus developed to reduce this computational load, leading finally to an efficient accuracy - computation time trade-off. The proposed method is a worthwhile extension of the one presented in our two previous conference papers [28] and [29], in a more formalized way and with added extensive experiments.

The contribution of the present article can be summed up by the combination of the following three points:

1. **Data**: The *use of the whole power spectrum* instead of usual features extracted from the audio signals, which is moreover essential for polyphonic event;
2. **Modeling**: A *very general generative modeling* of the power spectra designed for real-time audio classification, that uses monophonic models to build the polyphonic models;
3. **Real-time**: A *model reduction* technique based on hierarchical clustering preprocessing of the sound models.

The paper is organized as follows. The monophonic modeling is disclosed in Section 2 and is extended to polyphonic cases in Section 3. The reduction of the complexity is detailed in Section 4. The experiments to assess the performance of the method are presented in Section 5. Finally, Section 6 concludes the paper.

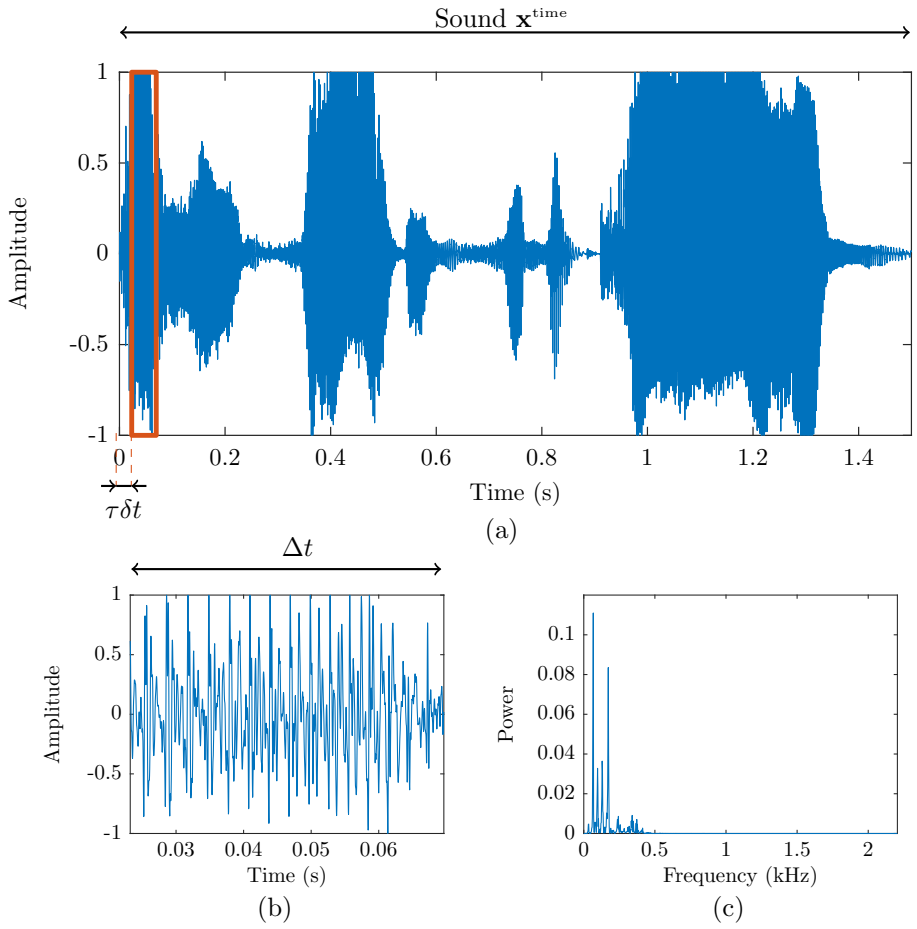

Figure 2: (a) Example with a real sound $\mathbf{x}^{\text{time}}$ containing a voice. This sound is split into time frames $\mathbf{x}_\tau^{\text{time}}$ (red rectangle) of size Δt and shifted by $\tau\delta t$ (b), and each time frame is converted into the normalized power spectrum $\mathbf{x}_\tau$ (c).

2. Monophonic modeling of the classes

2.1. Problem statement

The purpose of this paper is to provide a real-time sound classification method. Consider the case where the objective is to classify sounds coming from video games (case study from the company A-Volute[1]). The classification uses only on the audio mix coming from the video game, *without* any additional knowledge – meta-data from the game for instance. The sound is assumed to contain events coming from some *classes of sounds* – for instance a gunshot or an airplane – which have to be inferred (see Figure 2(a)). The objective of classifying at time t a sound can be written as follows:

$$\widehat{z} = \underset{z}{\operatorname{argmax}}\, p\left(z \middle| f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right), t\right), \tag{1}$$

where $\widehat{z}$ is an estimate of the label $z \in \{1, ..., K\}$ representing the class of sound at time t (for instance, the class $z = 1$ is composed of airplane sounds, $z = 2$ is composed of gunshot sounds, *etc.*), $\mathbf{x}^{\text{time}}$ is a sound considered as a process of length T and $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ is the observed sound in the time interval $[t - \Delta T, t]$, ΔT is the period of observation, $f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right)$ is a function that computes features from $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ according to the constraints and objectives described below, and $p\left(z \middle| f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right), t\right)$ is the probability of the class z given the features extracted from the sound $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$. The class label z is assumed to be the same over the time period $[t - \Delta T, t]$ (continuity of z and one active sound at most): this is the definition of a *monophonic frame*.

Several constraints are considered for real-time classification. The first is the time related to the data acquisition which constraints t to be a multiple of the *shift length* δt, that is $t \in \{0, \delta t, 2\delta t, ...\}$. This shift length is important because the classification has to be done at every multiple of δt, therefore the speed of the recognition system has to be less than this shift length. The second constraint is related to ΔT which is the *period of observation* of the sound which has to be set according to the processing time (see Section 2.2). The problem formulation as an argmax of a distribution is suitable because it is mathematically well-posed and it is a classical framework in probabilistic modeling. From the objective in Eq. (1), $f(\cdot)$ and $p(\cdot)$ have to be chosen carefully, and are disclosed in the following sections.

Remark. In practice, all the processing are done in discrete time, due to the processing on computers. As a result, the sound $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ will be a real vector after sampling on the computer at a rate F (see the value F in Section 5).

2.2. Feature design

The function $f(\cdot)$ is used to compute relevant features from $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ to perform the classification. Indeed, time domain signals are not well suited

[1] A software editor company specialized in 3D sound.

for audio classification since they are not discriminant enough features: they convey the phase information and also the volume, from which we want the method to be invariant. The sound is converted into the frequency domain: more particularly the *normalized power spectrum* is considered. Therefore by using normalized power spectrum we reach the invariance property. Moreover this transformation will be particularly relevant for polyphonic spectrum since it preserves the additivity of uncorrelated signals (see Section 3). Consequently, a possible definition of $f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right)$ can be $f_{\text{norm}} \circ f_{\text{FT}}\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right)$, where f_{FT} is the function that computes the Fourier Transform and f_{norm} is the function that normalizes the complex spectrum to get the normalized power spectrum (defined later in Eq. (5)).

However, since the classification has to be done in real-time such a large amount of data contained in $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ cannot be used. This is why this piece of sound is split into *time frames*:

$$(\mathbf{x}^{\text{time}}_{1t}, ..., \mathbf{x}^{\text{time}}_{Nt}) = f^{(\Delta t)}_{\text{frame}}\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right), \tag{2}$$

where $N = \lfloor (\Delta T - \Delta t)/\delta t \rfloor$ is the number of frames that can be effectively computed within the time interval $[t - \Delta T, t]$ and each frame $\mathbf{x}^{\text{time}}_{\tau t} \in \mathbb{R}^{\Delta t}$ is defined by:

$$\mathbf{x}^{\text{time}}_{\tau t} = \mathbf{x}^{\text{time}}_{[t-\Delta T+\tau\delta t, t-\Delta T+\tau\delta t+\Delta t]},\ \tau = 1, ..., N. \tag{3}$$

The frame $\mathbf{x}^{\text{time}}_{\tau t}$ is a portion of $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ of size Δt, overlapping the previous frames by a duration $\tau\delta t$ (see Figure 2(b)). The frame length Δt is set to a small value to allow fast processing for the Fourier Transform: as a counterpart the frame will contain less information than a large frame (like previously). As a result, a collection of several normalized power spectra is computed instead of a single large spectrum. The framed normalized power spectra are denoted by:

$$(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}) = f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right) = f_{\text{norm}} \circ f_{\text{FT}} \circ f^{(\Delta t)}_{\text{frame}}\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right), \tag{4}$$

where f_{FT} and f_{norm} operate frame-wise. Each normalized power spectrum is computed as follows:

$$\mathbf{x}_{\tau t} = \frac{\left|\mathbf{x}^{\text{freq}}_{\tau t}\right|^2}{||\mathbf{x}^{\text{freq}}_{\tau t}||^2}, \tag{5}$$

where $\mathbf{x}^{\text{freq}}_{\tau t} \in \mathbb{C}^B$ is the Fourier Transform of $\mathbf{x}^{\text{time}}_{\tau t}$ (only $B \leq \Delta t$ frequency bins are kept), $|\cdot|$ is the element-wise modulus and $||\cdot||$ is the ℓ_2-norm (see Figure 2(c)). The choice of Δt results in a trade-off between the computation time and the quality of the approximation: the larger is Δt the better is the approximation but the larger is the computation time.

2.3. Model design

The question now is to define the distribution $p(\cdot)$. We consider a generative modeling because a similar generative process will be used in the polyphonic case that simplifies the classification task (considers only the monophonic dataset to construct the polyphonic one). This is a real advantage compared to standard predictive modeling or usual multi-label learning. The process for generating a sequence of N spectra is detailed through the following generative model:

- **Random**: Draw a class label: $z \sim \text{Mult}_K(1, \mathbf{p})$,
- **Random**: Draw a sound of length T from class z: $\mathbf{x}^{\text{time}} \sim p_{\text{sound}}(\cdot|z)$,
- **Random**: Draw a time index: $t \sim \mathcal{U}([0, T])$,
- **Deterministic**: Compute the features: $(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}) = f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T, t]}\right)$,

where $\text{Mult}_K(1, \mathbf{p})$ is a multinomial distribution over K categories and 1 draw with probabilities $\mathbf{p} = (p_k)_{k=1}^K \left(p_k > 0,\ \sum_{k=1}^K p_k = 1\right)$ and $\mathcal{U}([a, b])$ is the uniform distribution over the interval $[a, b]$. The generative modeling we adopt has the main advantage of being *hypothesis free*: the distribution that generates the class is the very general multinomial distribution and the distribution that generates the sounds is assumed to be a general distribution over real-valued processes of size T denoted by $p_{\text{sound}}(\cdot|z)$. The whole generative process is completely defined up to the distribution $p_{\text{sound}}(\cdot|z)$ which will be treated further in the conditionaling modeling.

Recall the objective in Eq. (1), we can decompose this objective using the Bayes theorem as follows:

$$p(z|\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}, t) \propto p(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}|z, t)\, p(z|t). \tag{6}$$

The generative process models the joint distribution $p(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}, t, \mathbf{x}^{\text{time}}, z)$, however only the conditional $p\left(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}\middle|z, t\right)$ is useful for the monophonic classification task. Moreover, as a consequence of the generative model, we have that $p(z|t) = p(z)$.

Several hypotheses are made to simplify the distribution. The first hypothesis is an *independence* hypothesis. Assuming that the shift length δt is large enough, two consecutive frames can be considered independent. In practice for $\delta t = \Delta t/2$ or $\delta t = \Delta t/4$ (our future chosen values) there is approximately independence between two consecutive frames. The distribution of a sequence of spectra can thus be approximated using the following conditional independence assumption:

$$p\left(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}\middle|z, t\right) = \prod_{\tau=1}^{N} p\left(\mathbf{x}_{\tau t}\middle|z, t\right). \tag{7}$$

Eq. (7) can be viewed either as an aggregation (as in [28]) or as an independence assumption. The second hypothesis is an hypothesis of *stationarity in time*, so

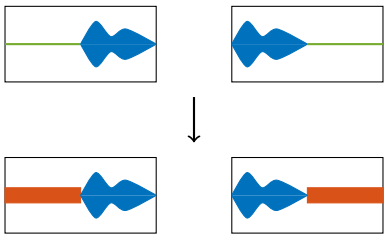

Figure 3: We illustrate the following phenomenon. In real-time, the sound (blue filled curve) neither begins at the beginning of the frame (top left) nor ends at the ending (top right). This is why we add (bottom left and right) Gaussian white noise (red rectangle) so as to fill the "silence" (green line).

that the distribution of spectrum $\mathbf{x}_{\tau t}$ given the class z does not depend on the time t:

$$p\left(\mathbf{x}_{\tau t}\middle|z,t\right) = p\left(\mathbf{x}_{\tau t'}\middle|z,t'\right). \tag{8}$$

The third hypothesis is an hypothesis of *stationarity over the frames*, so that the distribution of a spectrum does not depend on the shift index τ (since the class is the same in the time interval $[t-\Delta T, t]$):

$$p\left(\mathbf{x}_{\tau t}\middle|z\right) = p\left(\mathbf{x}_{\tau' t}\middle|z\right). \tag{9}$$

Given Eq. (8) and Eq. (9), the indexes τ, t can be removed so that $\mathbf{x}_{\tau t} = \mathbf{x}$, and finally the distribution can be written:

$$p\left(\mathbf{x}_{\tau t}\middle|z,t\right) = p\left(\mathbf{x}\middle|z\right). \tag{10}$$

2.4. Estimation of the model

We suppose that a learning set $\left(\mathbf{x}_{(i)}, z_{(i)}\right)_{i=1}^{n}$ is available using the generative process introduced in the previous section. The conditional distribution of Eq. (10) is estimated using a *kernel density estimation* of the form:

$$\widehat{p}\left(\mathbf{x}|z\right) = \frac{1}{n_z} \sum_{\substack{i \\ z_{(i)}=z}} K\left(\mathbf{x}, \mathbf{x}_{(i)}\right), \tag{11}$$

where n_z is the number of samples in the class z and $K\left(\cdot, \mathbf{x}_{(i)}\right)$ is a kernel that has to be chosen. From the definition of $\mathbf{x}$, several kernels can be used: the Dirichlet kernel – though it is not flexible enough for our purpose – or a mixture of Dirichlet kernel – which is more flexible but in a real-time context the involved computational load is to high. We chose to use a *multinomial kernel* by quantizing the spectrum so that it becomes a vector of integers that sums to a quantization factor $q \in \mathbb{N}$. Indeed this kernel gathers the advantages to be easy to evaluate, and reaches an efficient computation time - accuracy trade-off as described in Section 4, because only dot products are required to compute the

distribution (the normalization constant is the same for each kernel). Consider $\mathbf{x}_{(i)}^{(q)} \in \mathbb{N}^B$ the closest integer vector to $q\mathbf{x}_{(i)}$ which sums to q defined by:

$$\mathbf{x}_{(i)}^{(q)} = \underset{\mathbf{x}\in\mathbb{N}^B}{\operatorname{argmin}} \left|\left|q\mathbf{x}_{(i)} - \mathbf{x}\right|\right|. \tag{12}$$

We define the approximated kernel:

$$K^{(q)}\left(\cdot, \mathbf{x}_{(i)}^{(q)}\right) = \mathrm{Mult}_B\left(\cdot; q, \mathbf{p}_{(i)}^{(q)}\right), \tag{13}$$

where the parameter $\mathbf{p}_{(i)}^{(q)}$ is defined by:

$$\mathbf{p}_{(i)}^{(q)} = \frac{1}{q}\mathbf{x}_{(i)}^{(q)}. \tag{14}$$

The approximated kernel $K^{(q)}\left(\cdot, \mathbf{x}_{(i)}^{(q)}\right)$ converges to $K\left(\cdot, \mathbf{x}_{(i)}\right)$ as q goes to infinity (see Appendix A for a proof). For large enough q this approximation will be correct. In practice, values of q close to B are good enough (see Section 5). Finally, the probability of the classes are estimated by maximum likelihood:

$$\widehat{p}_z = \frac{n_z}{n} \tag{15}$$

Remark. In practice, the learning set is built using a slightly different generative process for practical reasons. A set of sounds already sampled from $p_{\mathrm{sound}}(\cdot|z)$ is supposed to be available. We first draw a class label $z_{(i)}$ and then a sound. The step of drawing a time index is different but *mimick* the generative process: every time index are considered and creates several time frames. The remaining of the process is the same: the time frames are transformed into the normalized power spectrum. The framing implies that in a given frame the sound neither necessarily begins at the beginning of this frame nor ends at the ending. Rather than adding silence – which contains some information – Gaussian white noise (GWN) is added – which conveys no statistical information – to fill this "blank" (see Figure 3).

3. Polyphonic modeling of the classes

3.1. Motivation and polyphonic features

The polyphonic classification relies on the monophonic dataset, built in Section 2.4 and uses a similar framework than the monophonic one. Using a suitable decomposition of the polyphonic spectrum, the method uses only the monophonic spectra and does not have to learn mixture of sounds: it is an advantage of the proposal. The case for a mixture of two sources is considered: this means that a frame has two simultaneous labels, denoted by $\mathbf{z} = (z_1, z_2) \in \{1, ..., K\}^2,\ z_1 \neq z_2$. Recalling the objective in Section 2, the polyphonic decision rule is thus written as:

$$\widehat{\mathbf{z}} = \underset{\mathbf{z}}{\operatorname{argmax}}\, p\left(\mathbf{z}\middle| f\left(\mathbf{x}_{[t-\Delta T, t]}^{\mathrm{time}}\right), t\right), \tag{16}$$

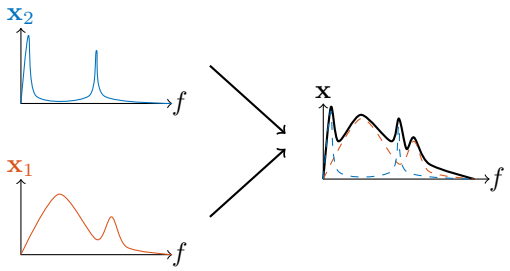

Figure 4: Consider that two power spectra $\mathbf{x}_1$ (red, left bottom curve) and $\mathbf{x}_2$ (left top blue curve) are mixed. The resulting spectrum $\mathbf{x} = \phi\mathbf{x}_1 + (1-\phi)\mathbf{x}_2$ (thick black right curve) will depend on the relative powers of the two spectra, represented by ϕ.

where $\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}$ is the observed polyphonic sound, defined by the sum of the monophonic sounds $\mathbf{x}^{\text{time}}_{1[t-\Delta T,t]}$ and $\mathbf{x}^{\text{time}}_{2[t-\Delta T,t]}$ from the corresponding classes z_1 and z_2, as:

$$\mathbf{x}^{\text{time}}_{[t-\Delta T,t]} = \mathbf{x}^{\text{time}}_{1[t-\Delta T,t]} + \mathbf{x}^{\text{time}}_{2[t-\Delta T,t]}. \tag{17}$$

The same features are used since the power spectrum keeps the additivity of the spectra – it was designed to this purpose. Then, the polyphonic feature can be expressed as a combination of the underlying monophonic features. Indeed, some calculi (see Appendix B) on the features lead to the following result:

$$f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right) = \boldsymbol{\phi}_t \cdot f\left(\mathbf{x}^{\text{time}}_{1[t-\Delta T,t]}\right) + (\mathbf{1}-\boldsymbol{\phi}_t) \cdot f\left(\mathbf{x}^{\text{time}}_{2[t-\Delta T,t]}\right), \tag{18}$$

where $\mathbf{x} \cdot \mathbf{y}$ is the element-wise multiplication between $\mathbf{x}$ and $\mathbf{y}$. By using normalized power spectra the modeling induces proportions $\boldsymbol{\phi}_t = (\phi_{1t}, ..., \phi_{Nt})$ which represent the ratio between the power of one source and the power of the two sources (see Figure 4). At a given time shift τ the proportion is defined by:

$$\phi_{\tau t} = \frac{||\mathbf{x}^{\text{freq}}_{1\tau t}||^2}{||\mathbf{x}^{\text{freq}}_{1\tau t}||^2 + ||\mathbf{x}^{\text{freq}}_{2\tau t}||^2}, \tag{19}$$

where $\mathbf{x}^{\text{freq}}_{1\tau t}$ is defined as in the monophonic feature design section 2.2. A detailed explanation is available in Appendix B. This framework can easily be extended to a mixture of more than two sources.

Remark. We assume that the time at which the sounds are observed is the same, but the individual sounds may have two different starting time: we do not care about the underlying synchronization of the sound for the simplicity of the presentation here but it is obviously taken into account by the proposed method.

3.2. Model design

Based on the monophonic generative model, the following generative model of polyphonic spectra (for two classes) is defined by:

- **Random**: Draw independently without replacement two different class labels $z_1, z_2 \sim \text{Mult}_K\left(1, \widehat{\mathbf{p}}\right)$,
- **Random**: Draw independently two different sound: $\mathbf{x}_1^{\text{time}} \sim p_{\text{sound}}(\cdot|z_1)$ and $\mathbf{x}_2^{\text{time}} \sim p_{\text{sound}}(\cdot|z_2)$
- **Random**: Draw a time indexe: $t \sim \mathcal{U}\left([0, T]\right)$
- **Deterministic**: Compute the features: $(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}) = \boldsymbol{\phi}_t \cdot f\left(\mathbf{x}^{\text{time}}_{1[t-\Delta T,t]}\right) + (\mathbf{1} - \boldsymbol{\phi}_t) \cdot f\left(\mathbf{x}^{\text{time}}_{2[t-\Delta T,t]}\right)$.

This polyphonic generative process models the joint distribution $p(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}, t, \mathbf{x}_1^{\text{time}}, \mathbf{x}_2^{\text{time}}, z_1, z_2)$, and again only the conditional distribution is used here. The Bayes theorem allows to write:

$$p\left(z_1, z_2|\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}, t\right) \approx p\left(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}|z_1, z_2, t\right) p\left(z_1, z_2|t\right). \tag{20}$$

From the generative model we have that $p\left(z_1, z_2|t\right) = p\left(z_1, z_2\right)$. The frame independence assumption of Eq. (7) is still valid so it leads to the following approximation:

$$p\left(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}|z_1, z_2, t\right) \approx \prod_{\tau=1}^{N} p\left(\mathbf{x}_{\tau t}|z_1, z_2, t\right). \tag{21}$$

Moreover the stationary hypotheses are also valid in this context (stationarity in time as in Eq. (8) and stationarity over the frames as in Eq. (9)). As $\phi_{\tau t}$ depends on $\mathbf{x}_{\tau t}$, the proportion will inherit from the stationarity hypotheses. As a consequence, the indexes are removed: $\mathbf{x}_{\tau t} = \mathbf{x}$ and $\phi_{\tau t} = \phi$, and the polyphonic conditional distribution can be expressed as:

$$p\left(\mathbf{x}_{\tau t}|z_1, z_2, t\right) = p\left(\mathbf{x}|z_1, z_2\right). \tag{22}$$

3.3. Polyphonic dataset building and Estimation of the model

The main advantage of the previous modeling is that it allows to build a new dataset for polyphonic sounds using only monophonic ones: this releases the need to record and label manually polyphonic events which can be hard and time consuming. The procedure is rather simple: based on the polyphonic generative model, we first draw two class labels from which two sounds are drawn. Then each sound is split in time frames and converted into normalized power spectra, and the energy of each frequency domain frame is computed. Finally pair-wise normalized spectra are computed by weighting two normalized monophonic spectra by the corresponding proportion $\phi_{(i)}$ (defined in Eq. 19), in the form:

$$\mathbf{x}_{(i)} = \phi_{(i)}\mathbf{x}_{1(i)} + \left(1 - \phi_{(i)}\right)\mathbf{x}_{2(i)}. \tag{23}$$

This procedure leads to a learning set $\left(\mathbf{x}_{(i)}, \mathbf{z}_{(i)}\right)_{i=1}^{n}$, where $\mathbf{z}_{(i)} = \left(z_{1(i)}, z_{2(i)}\right)$. The previous distribution is estimated using a *kernel density estimation* of the form:

$$\widehat{p}\left(\mathbf{x}|\mathbf{z}\right) = \frac{1}{n_{z_1} n_{z_2}} \sum_{\substack{i \\ \mathbf{z}_{(i)}=\mathbf{z}}} K\left(\mathbf{x}, \mathbf{x}_{(i)}\right). \tag{24}$$

As for the monophonic estimation, the kernel will be approximated using a multinomial kernel with the same convergence property. By construction, this method can theoretically only recover polyphonic sounds with the same proportions as in the learning set, but we will see in Section 5 that the method performs well even with random proportions between sounds.

4. Reducing the computational load

The main objective of this paper is to provide a real-time audio classification method. The previous two sections defined such a method from a real-time point of view (split sounds in short frames, use a kernel density estimate with multinomial kernels). However, the computational load for computing the distribution is currently too high to be used in practice. This is the point of this section, that is to reduce the computational load.

4.1. Evaluating the complexity of the classification task

Consider the following writing of the monophonic distribution in Eq. (11):

$$\widehat{p}\left(\mathbf{x}|z\right) \propto \sum_{\substack{i \\ z_{(i)}=z}} \exp\left(\left(\mathbf{x}^{(q)}\right)^{\top} \log\left(\mathbf{p}_{(i)}^{(q)}\right)\right), \tag{25}$$

where $\mathbf{x}^{\top}$ is the transpose $\mathbf{x}$. The complexity of the algorithm is roughly $\mathcal{O}\left(n\right)$ since it consists in computing dot products between the unknown spectrum and all the $\mathbf{p}_{(i)}^{(q)}$. Even for small datasets, the number of models can be very large (typically 100k models), and therefore the computation time is larger than the duration of a frame. As the previous equations are derived from the generative model (and cannot be changed), we can essentially reduce the complexity by reducing the number of models n.

4.2. Hierarchical clustering of the monophonic models

A model reduction technique was already considered in [25] for NMF dictionaries: the authors used a k-means clustering technique and kept the centroid of the clusters as their reduced models. Their results suggested that the accuracy of the resulting system was not monotonic with the considered number of clusters. Contrary to these authors, the proposed model reduction algorithm of the present work allows to control the complexity and leads to an efficient computation time - accuracy trade-off.

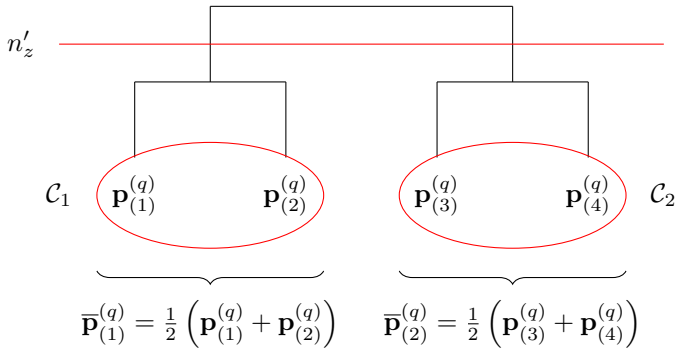

Figure 5: Illustration of the complexity reduction using hierarchical clustering. Consider that in a given class there are 4 models parameterized by $\mathbf{p}^{(q)}_{(1)}, ..., \mathbf{p}^{(q)}_{(4)}$. The hierarchical clustering results in a tree representing how the models are structured. Choose a number of models after reduction n' (here $n' = 2$) and "cut" the tree so as to get the clusters $\mathcal{C}_1, \mathcal{C}_2$. Finally merge the models in each cluster by taking the mean mixture of these models, to get $\overline{\mathbf{p}}^{(q)}_{(1)}$ and $\overline{\mathbf{p}}^{(q)}_{(2)}$.

The idea is to perform class-wise hierarchical clustering of the kernels parameterized by the $\mathbf{p}^{(q)}_{(i)}$ for $i|z_{(i)} = z$, and use the resulting tree to create mixtures of these parameters according to the clusters. Hierarchical clustering requires a distance between the elements and a linkage criterion. We consider the Hellinger distance [30] since the elements to cluster are discrete probability distributions. The elements are linked using the Ward criterion [31] – a usual linkage criterion.

In class z, a hierarchical clustering is performed as described previously using a class-wise reduction factor r_z (or a global reduction r). This factor is defined as the initial number of models n_z over the number of models after reduction n'_z: $r_z = n_z/n'_z$. For each cluster $\mathcal{C}_{i'} \in \{\mathcal{C}_1, ..., \mathcal{C}_{n'_z}\}$ resulting from the hierarchical clustering, $\mathcal{I}_{i'} = \left\{ i \,\middle|\, \mathbf{p}^{(q)}_{(i)} \in \mathcal{C}_{i'} \right\}$ denotes the set of the indexes such that $\mathbf{p}^{(q)}_{(i)}$ belongs to the cluster $\mathcal{C}_{i'}$ $(i' = 1, ..., n'_z)$. A mixture $\overline{\mathbf{p}}^{(q)}_{(i')}$ of the parameters $\mathbf{p}^{(q)}_{(i)}$ in cluster $\mathcal{C}_{i'}$ is defined as:

$$\overline{\mathbf{p}}^{(q)}_{(i')} = \frac{1}{\text{card}\left(\mathcal{I}_{i'}\right)} \sum_{i \in \mathcal{I}_{i'}} \mathbf{p}^{(q)}_{(i)}. \tag{26}$$

The label associated to $\overline{\mathbf{p}}^{(q)}_{(i')}$ is noted $z_{(i')}$. An illustration of this process is displayed in Figure 5. It is shown in Section 5 that the accuracy - computation time trade-off is monotonically changed with the reduction factor.

4.3. Controlling the trade-off using a threshold procedure

A heuristic algorithm to reduce the model in a more efficient way was developed based on the previous algorithm. Instead of reducing all the classes with the same factor r_z, the classes are reduced independently by choosing the reduction factor so that the class-wise test accuracy reaches a given threshold

t_{accuracy}. Consider an initial reduction factor for the classes $r_z^{(0)}$. The models are reduced using this factor and the class-wise test accuracy is computed: if it is above the threshold t_{accuracy} the procedure stops, else the factor is reduced so as to increase the accuracy. This procedure is iterated until the threshold is reached. This procedure can be used to reach a given computation time threshold instead of a class-wise test accuracy threshold. An example of some iterations of this procedure is displayed in Table 5 in Section 5.

Remark: Polyphonic modeling reduction. Experiments in Section 5 show that a very reduced model can be used to perform monophonic classification without loosing too much accuracy. Therefore polyphonic classification will be performed using the reduced models parameterized by the $\overline{\mathbf{p}}_{(i')}^{(q)}$ described in Section 4.2. The goal is to construct a new learning set containing relevant mixtures models. The method consists in computing pair-wise mixture models from the different classes and reduce this collection of models using a hierarchical clustering like in Section 4.2: the reduction factor is set so that the computation time does not exceed the objective.

5. Numerical experiments on real-world datasets

5.1. Databases and baseline systems

Databases. Two datasets are considered for monophonic classification. The first is the A-Volute dataset, composed of 704 video game sounds divided into 5 classes (alarm, detonation, step, vehicle, voice). The second is the ESC-10 dataset [32], composed of 400 sounds divided into 10 classes. For polyphonic classification, mixtures from the A-Volute dataset and audio recordings from the video game Battlefield 1 are considered, the latter containing events of 3 different classes (detonation, step and voice), alone and by mixture of 2 and 3 classes. The TUT-SED 2017 [33] dataset is also considered, which contains real-life recordings in a street context.

Parameters tuning. All the sounds are resampled to $F = 44.1$kHz and centered. The frame size considered are $\Delta t = 512, 1024, 2048$ samples (11.6ms, 23.2ms and 46.4ms respectively) and the time shift is fixed to $\delta t = 512$ samples (11.6ms). The number of frequency bins $B = \Delta t/2 + 1$ (the half-right of the spectrum is discarded and the Nyquist frequency is kept). The quantization of the spectra is set to $q = B$: in practice, this seems to be an optimal number. The learning sets are divided using the v-fold scheme to perform cross-validation (with $v = 5$): precisely the split is performed on the frames and not on the sounds, unlike other authors do. Frame sequences length is set to $N = 40$: this corresponds to approximately 500ms of audio data to take the decision.

Evaluation metrics. For monophonic classification, the evaluation metric is the classification accuracy in cross-validation, that is the number of correctly classified frames over the total number of frames. For polyphonic classification, the evaluation metric is the segment-based error rate and segment-based F1 score (see [34] for a detailed explanation of these metrics).

Table 1: Mean cross-validation classification accuracy and standard deviation (in %) for different values of Δt on the A-Volute dataset and for the different methods. The results are obtained using the full model (not reduced).

	A-Volute		
Δt	512	1024	2048
Proposed method	**95.41** (0.23)	**98.69** (0.17)	**99.89** (0.01)
GMM based [9]	67.96 (4.05)	73.34 (0.52)	75.47 (0.97)
DCNN based [5]	51.50 (10.9)	70.32 (4.59)	59.70 (5.99)
DMM based	81.88 (0.99)	85.65 (1.23)	87.94 (1.89)
Kernel based	36.29 (0.24)	40.72 (0.18)	45.65 (0.15)
Human	-	-	91.7

Monophonic baseline systems. The proposed method for monophonic classification is compared with several baseline systems. The first is a GMM using 20 MFCC and their first and second derivatives, which is the baseline of the DCASE challenge. The second is Deep Convolutional Neural Network (DCNN) using log-mel spectrograms inspired by [5]. A Dirichlet Mixture Model (DMM) [35] is considered, so that a class is modeled by a mixture of Dirichlet distribution, and the number of components per class is selected using the BIC (Bayesian Information Criterion). Finally a purely non-parametric method is studied consisting in a kernel density estimation [36, Chapter 6] using Dirichlet kernels: each spectrum in a class is considered as the mode of a Dirichlet distribution and we model the distribution of a new model given a class as the mixture of these distributions. Human listening results were gathered for the two monophonic datasets: the results for ESC-10 are available in [32]. For the A-Volute dataset an experiment was carried out where 27 subjects classified 50 0.5s-sounds randomly chosen in the 5 classes.

Polyphonic baseline systems. The polyphonic classification method was also compared with a neural network based method, called CRNN (Convolutional Recurrent Neural Network) inspired by [20]. It uses log-mel spectrogram to feed a neural network that consists in convolutional layers (for feature extraction) followed by recurrent layers (for temporal detection) and ended by a dense layer that performs classification.

5.2. Results on the monophonic classification

The influence of the value of Δt on the A-Volute dataset is reported in Table 1. The larger Δt the better the methods perform. Indeed, they have more audio data to compute the decision. There is a gap from $\Delta t = 512$ to $\Delta t = 1024$: this can be explained by the fact that there is no overlap between the time frames in the learning set in the case of $\Delta t = 512$, and so this is more difficult to classify an unseen frame.

The classification accuracies for the different datasets are available in Table 2. The proposed method outperforms the considered baselines and even the humans: it performs nearly perfectly (99.73%), followed by the mixture of Dirichlet models

Table 2: Mean cross-validation classification accuracy (in %) and standard deviation for the different datasets and methods. All the models were used, which results in a model of size 60k elements per fold. The results for the best value of Δt only are displayed.

Δt	A-Volute 2048	ESC-10 2048
Proposed method	**99.89** (0.01)	**99.18** (0.04)
GMM based [9]	77.81 (0.64)	70.99 (0.59)
DCNN based [5]	87.49 (0.80)	74.03 (1.72)
DMM based	87.94 (1.89)	80.79 (1.49)
Kernel based	45.65 (0.15)	34.00 (0.24)
Human	91.7	95.7

Table 3: Computation time per frame (ms) for the proposed method method on a A-Volute dataset fold ($n = 71000$) and a ESC-10 fold ($n = 137000$) using different materials: three CPU and two GPU.

Type	Time (ms)	
	A-Volute	ESC-10
CPU Intel Core i7 **@**2.20 GHz	649.9	996.1
CPU Intel Core i7 **@**2.70 GHz	687.9	842.1
CPU Intel Core i7 **@**3.3 GHz	457.8	779.0
GPU NVidia GTX 1080	14.5	23.0
GPU NVidia GTX TitanX	15.4	29.1

Table 4: Accuracy (%) and computation time (ms) for different reduction of the models for the A-Volute dataset. The best case ($\Delta t = 2048$ samples) was considered. An Intel Core i7 **@**3.3GHz was used for the computations.

r	Accuracy (%)	Computation time (ms)
1	99.73 (0.05)	457.8
2	99.70 (0.03)	233.7
10	99.30 (0.05)	51.0
50	97.04 (0.15)	10.2
100	95.16 (0.21)	6.6
400	89.47 (0.32)	1.7

Table 5: Illustration of the heuristic algorithm for efficient model reduction. The class-wise test accuracy threshold is set to $t_{\text{accuracy}} = 90\%$ and the initial guess is $\mathbf{N}'^{(0)} = [250, 350, 350, 200, 300]$. If the class-wise test accuracy is below t_{accuracy}, n'_z decreases by 20, and if above n'_z increases by 1.

	Iterations	Alarm $z=1$	Detonation $z=2$	Engine $z=3$	Step $z=4$	Voice $z=5$
n'_z	0	250	350	350	200	300
	1	230	351	351	180	301
	2	210	352	352	160	302
	3	190	353	353	140	303
Class-wise Accuracy (%)	0	88.58	92.39	92.33	87.49	90.77
	1	88.58	92.15	92.33	88.72	90.40
	2	88.58	91.91	92.19	88.96	90.40
	3	90.91	91.91	92.19	91.19	90.40

(87.94%). The neural network manages to perform relatively well (87.49%) as well as the mixture of Gaussians (77.81%) but the kernel method performs poorly (46.46%). The DMM works relatively well, but the kernel based algorithm performs poorly: the reason can be that the mode of each distribution is too flat to represent correctly the class (not flexible enough as said in Section 2), and this results in coin toss problem.

The computation time was measured using different materials for the proposed algorithm, reported in Table 3. Given that a frame lasts 46.4ms, the computation time on a standard CPU is too large to be considered as real-time (457.8ms for the best CPU); on a high-end GPU the computations are very fast (14.5ms) but this kind of material is not common.

Concerning the reduced dictionary, the results are displayed in Table 4. The results are displayed for a global reduction factor; however a reduction factor per class can be used, as illustrated in Table 5. This reduction factor allows to control the trade-off between accuracy and computation time.

5.3. Results on the polyphonic classification

The results for polyphonic classification using the full models are displayed in Table 6. Concerning the A-Volute dataset, since mixtures were not available, artificial test mixtures were created to test the method. The designed method performs well on the A-Volute dataset (F1 score of 89.12 and error rate of 10.38) compared to the CRNN (F1 score of 49.88 and error rate of 62.68): indeed since mixtures were not available the neural network has to learn artificial mixtures which was not (perhaps) discriminant enough for it. The proposed method performs relatively well on the Battlefield recording (F1 score of 97.84 and error rate of 2.90) where mixtures were available, whereas the CRNN has a F1 score of 64.21 (for an error rate of 45.02). On the TUT-SED 2017 dataset the proposed method performs less (F1 score of 62.89 and error rate of 42.54), compared to the CRNN (F1 score of 46.47 and error rate of 60.35).

Table 6: Mean cross-validation segment-based F1 score and error rate (in % detoned by E.R.) and standard deviation for the different datasets and methods.

	A-Volute		Battlefield		TUT-SED 2017	
	F1	E.R.	F1	E.R.	F1	E.R.
Our method	**89.12** (0.38)	**10.38** (0.36)	**97.84** (0.12)	**2.90** (0.16)	**62.89** (0.09)	**42.54** (0.07)
CRNN	49.88 (5.40)	62.68 (4.18)	64.21 (0.59)	45.02 (0.84)	46.47 (0.30)	60.35 (0.42)

6. Conclusion

This article dealt with a new method for monophonic and polyphonic real-time audio sources classification. The method uses the whole power spectrum instead of predefined audio descriptors, which is also useful for polyphonic events. The classification is based on a generative model of power spectra, which has the main advantage of being hypothesis free and allows to derive a temporal MAP to make the decision. Contrary to other methods like neural networks, this technique models both monophonic and polyphonic sources in a single framework. Moreover, a method for reducing the complexity is proposed that leads to an efficient computation time - accuracy trade-off. However, the presented method works only for monochannel streams. In practice, the streams are often stereo (2 channels), 5.1 (6 channels) or 7.1 (8 channels). The future works will extend the method to perform classification on multichannel streams.

Appendix A. Point-wise convergence of the approximated kernel

Consider $\mathbf{X}_{(i)} \in \mathbb{R}^B$ a random vector that sums to 1 related to the spectrum $\mathbf{x}_{(i)}$ in the learning set. $\mathbf{X}_{(i)}^{(q)}$ is defined as the closest integer random vector to $q\mathbf{X}_{(i)}$ by:

$$\mathbf{X}_{(i)}^{(q)} = \underset{\mathbf{X}\in\mathbb{N}^B}{\operatorname{argmin}} \left|\left|q\mathbf{X}_{(i)} - \mathbf{X}\right|\right|. \tag{A.1}$$

We have that $\frac{1}{q}\mathbf{X}_{(i)}^{(q)}$ converges in distribution to $\mathbf{X}_{(i)}$ as q goes to infinity. As a result the kernel defined by:

$$K^{(q)}\left(\cdot, \mathbf{X}_{(i)}^{(q)}\right) = \mathrm{Mult}_B\left(\cdot; q, \mathbf{p}_{(i)}^{(q)}\right), \tag{A.2}$$

with parameter $\mathbf{p}_{(i)}^{(q)} = \frac{1}{q}\mathbf{X}_{(i)}^{(q)}$ converges to the original kernel $K\left(\cdot, \mathbf{X}_{(i)}\right)$.

Appendix B. Derivation of the polyphonic spectrum decomposition

Eq. (18) is derived using the following arguments. A polyphonic sound is the sum in the time domain of several sound sources:

$$\mathbf{x}_{[t-\Delta T,t]}^{\text{time}} = \mathbf{x}_{1[t-\Delta T,t]}^{\text{time}} + \mathbf{x}_{2[t-\Delta T,t]}^{\text{time}}. \tag{B.1}$$

The framing operation is linear so that:

$$f_{\text{frame}}^{(\Delta t)}\left(\mathbf{x}_{[t-\Delta T,t]}^{\text{time}}\right) = f_{\text{frame}}^{(\Delta t)}\left(\mathbf{x}_{1[t-\Delta T,t]}^{\text{time}}\right) + f_{\text{frame}}^{(\Delta t)}\left(\mathbf{x}_{2[t-\Delta T,t]}^{\text{time}}\right). \tag{B.2}$$

For a given frame:

$$\mathbf{x}_{\tau t}^{\text{time}} = \mathbf{x}_{1\tau t}^{\text{time}} + \mathbf{x}_{2\tau t}^{\text{time}}. \tag{B.3}$$

By the linearity of the Fourier transform, if two signals are summed in the time domain they will be summed in the frequency domain. Denoting by $\mathbf{x}_{1\tau t}^{\text{freq}}$ and $\mathbf{x}_{2\tau t}^{\text{freq}}$ the complex spectra, the sum of these spectra $\mathbf{x}_{\tau t}^{\text{freq}}$ is:

$$\mathbf{x}_{\tau t}^{\text{freq}} = \mathbf{x}_{1\tau t}^{\text{freq}} + \mathbf{x}_{2\tau t}^{\text{freq}}. \tag{B.4}$$

The modeling disclosed in Section 2.4 requires to deal with a normalized version composed of the elements $|\mathbf{x}_{\tau t}^{\text{freq}}|^2$ as in Eq. (5). Two sources from different classes are assumed to be *uncorrelated signals*, meaning that the power spectrum of the sum is approximately the sum of the power spectra:

$$\begin{aligned} \left|\mathbf{x}_{\tau t}^{\text{freq}}\right|^2 &= \left|\mathbf{x}_{1\tau t}^{\text{freq}} + \mathbf{x}_{2\tau t}^{\text{freq}}\right|^2 \\ &\approx \left|\mathbf{x}_{1\tau t}^{\text{freq}}\right|^2 + \left|\mathbf{x}_{2\tau t}^{\text{freq}}\right|^2. \end{aligned} \tag{B.5}$$

The normalized power spectrum associated to $\mathbf{x}_{\tau t}^{\text{freq}}$ is $\mathbf{x}_{\tau t}$:

$$\mathbf{x}_{\tau t} = \frac{\left|\mathbf{x}_{1\tau t}^{\text{freq}}\right|^2 + \left|\mathbf{x}_{2\tau t}^{\text{freq}}\right|^2}{\left|\left|\mathbf{x}_{1\tau t}^{\text{freq}}\right|\right|^2 + \left|\left|\mathbf{x}_{2\tau t}^{\text{freq}}\right|\right|^2}. \tag{B.6}$$

Define $P_{1\tau t} = \left|\left|\mathbf{x}_{1\tau t}^{\text{freq}}\right|\right|^2$ and $P_{2\tau t} = \left|\left|\mathbf{x}_{2\tau t}^{\text{freq}}\right|\right|^2$ the powers of the two sources. Some calculi lead to the following result:

$$\mathbf{x}_{\tau t} = \mathbf{x}_{1\tau t}\frac{P_{1\tau t}}{P_{1\tau t} + P_{2\tau t}} + \mathbf{x}_{2\tau t}\frac{P_{2\tau t}}{P_{1\tau t} + P_{2\tau t}}, \tag{B.7}$$

where $\mathbf{x}_{1\tau t}$ and $\mathbf{x}_{2\tau t}$ are defined as in Eq. (5). Define the proportion $\phi_{\tau t}$ as:

$$\phi_{\tau t} = \frac{P_{1\tau t}}{P_{1\tau t} + P_{2\tau t}}. \tag{B.8}$$

The previous result becomes:

$$\mathbf{x}_{\tau t} = \phi_{\tau t}\mathbf{x}_{1\tau t} + (1 - \phi_{\tau t})\mathbf{x}_{2\tau t}. \tag{B.9}$$

References

[1] L. R. Rabiner, A Tutorial on hidden Markov Models and Selected Applications in Speech Recognition, Proceedings of the IEEE 77 (2) (1989) 257–286.

[2] Y. Qian, M. Bi, T. Tan, K. Yu, Very Deep Convolutional Neural Networks for Noise Robust Speech Recognition, IEEE/ACM Transactions on Audio, Speech, and Language Processing 24 (12) (2016) 2263–2276.

[3] G. Kour, N. Mehan, Music Genre Classification using MFCC, SVM and BPNN, International Journal of Computer Applications 112 (6) (2015) 12–14.

[4] C. Joder, S. Essid, G. Richard, Temporal Integration for Audio Classification With Application to Musical Instrument Classification, IEEE Transactions on Audio, Speech, and Language Processing 17 (1) (2009) 174–186.

[5] K. J. Piczak, Environmental sound classification with convolutional neural networks, in: 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), 2015, pp. 1–6.

[6] J. Flocon-Cholet, Classification audio sous contrainte de faible latence, Ph.D. thesis, Université de Rennes 1 (2016).

[7] G. Peeters, B. L. Giordano, P. Susini, N. Misdariis, S. McAdams, The Timbre Toolbox: extracting audio descriptors from musical signals, The Journal of the Acoustical Society of America 130 (5) (2011) 2902–2916.

[8] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, T. Virtanen, Detection and classification of acoustic scenes and events, Detection and Classification of Acoustic Scenes and Events 2017.

[9] C. Clavel, T. Ehrette, G. Richard, Events Detection for an Audio-Based Surveillance System, in: 2005 IEEE International Conference on Multimedia and Expo, 2005, pp. 1306–1309.

[10] R. Radhakrishnan, A. Divakaran, A. Smaragdis, Audio analysis for surveillance applications, in: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005., 2005, pp. 158–161.

[11] D. Istrate, M. Binet, S. Cheng, Real time sound analysis for medical remote monitoring, in: 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2008, pp. 4640–4643.

[12] J.-H. Choi, J.-H. Chang, On using acoustic environment classification for statistical model-based speech enhancement, Speech Communication 54 (3) (2012) 477–490.

[13] A. Bietti, F. Bach, A. Cont, An online em algorithm in hidden (semi-)Markov models for audio segmentation and clustering, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 1881–1885.

[14] T. Heittola, A. Mesaros, A. Eronen, T. Virtanen, Audio context recognition using audio event histograms, in: 18th European Signal Processing Conference, 2010, pp. 1272–1276.

[15] S. Lecomte, R. Lengell, C. Richard, F. Capman, B. Ravera, Abnormal events detection using unsupervised One-Class SVM - Application to audio surveillance and evaluation -, in: 2011 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS), 2011, pp. 124–129.

[16] S. Sameh, L. Zied, On the Use of Time-Frequency Reassignment and SVM-Based Classifier for Audio Surveillance Applications, International Journal of Image, Graphics and Signal Processing 6 (12) (2014) 17–25.

[17] R. Biondi, G. Dys, G. Ferone, T. Renard, M. Zysman, Low Cost Real Time Robust Identification of Impulsive Signals, International Journal of Computer, Electrical, Automation, Control and Information Engineering 8 (9) (2014) 1653–1656.

[18] C. P. Dadula, E. P. Dadios, Neural network classification for detecting abnormal events in a public transport vehicle, in: 2015 International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM), 2015, pp. 1–6.

[19] D. Palaz, M. M. Doss, R. Collobert, Convolutional Neural Networks-based continuous speech recognition using raw speech signal, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 4295–4299.

[20] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, T. Virtanen, Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection, IEEE/ACM Transactions on Audio, Speech, and Language Processing 25 (6) (2017) 1291–1303.

[21] P. Smaragdis, B. Raj, M. Shashanka, A probabilistic latent variable model for acoustic modeling, in: In Workshop on Advances in Models for Acoustic Processing at NIPS, 2006.

[22] P. Paatero, U. Tapper, Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values, in: Fourth International Conference on Statistical methods for the Environmental Sciences Environmetrics, Vol. 5, Espoo, Spain, 1994, pp. 111–126.

[23] E. Benetos, G. Lafay, M. Lagrange, M. D. Plumbley, Detection of overlapping acoustic events using a temporally-constrained probabilistic model, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 6450–6454.

[24] V. Bisot, S. Essid, G. Richard, Overlapping sound event detection with supervised Nonnegative Matrix Factorization, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 31–35.

[25] A. Mesaros, T. Heittola, O. Dikmen, T. Virtanen, Sound event detection in real life recordings using coupled matrix factorization of spectral representations and class activity annotations, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 151–155.

[26] T. Heittola, A. Mesaros, T. Virtanen, M. Gabbouj, Supervised model training for overlapping sound events based on unsupervised source separation, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 8677–8681.

[27] R. Alazaidah, F. Thabtah, Q. Al-Radaideh, A multi-label classification approach based on correlations among labels, International Journal of Advanced Computer Science and Applications 6 (2).

[28] M. Baelde, C. Biernacki, R. Greff, A mixture model-based real-time audio sources classification method, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 2427–2431.

[29] M. Baelde, C. Biernacki, R. Greff, Classification de signaux audio en temps-réel par un modèle de mélanges d'histogrammes, in: 49e Journées de Statistique, Avignon, France, 2017.

[30] E. Hellinger, Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen, J. Reine Angew. Math. 136 (1909) 210–271.

[31] J. H. J. Ward, Hierarchical Grouping to Optimize and Objective Function, Journal of the American Statistical Association 58 (301) (1963) 236–244.

[32] K. J. Piczak, ESC: Dataset for Environmental Sound Classification, ACM Press, 2015, pp. 1015–1018.

[33] A. Mesaros, T. Heittola, T. Virtanen, TUT database for acoustic scene classification and sound event detection, in: 24th European Signal Processing Conference, Vol. 2016, 2016.

[34] A. Mesaros, T. Heittola, T. Virtanen, Metrics for Polyphonic Sound Event Detection, Applied Sciences 6 (6) (2016) 162.

[35] Z. Ma, P. K. Rana, J. Taghia, M. Flierl, A. Leijon, Bayesian estimation of Dirichlet mixture model with variational inference, Pattern Recognition 47 (9) (2014) 3143–3157.

[36] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer Series in Statistics, Springer New York, New York, NY, 2009.

Maxime Baelde received M. D. degree in Applied Mathematics from Université de Lille in 2015. He is now a Ph. D student at Inria Lille and A-Volute. His research interests are focused on real-time audio source classification and separation, and generative models.

Christophe Biernacki is the Scientific Deputy at Inria Lille and Scientific Head of the MODAL team, and received his Ph. D degree from Université de Compiègne in 1997. His research interests are focused on model-based and model-free clustering of heterogeneous data.

Raphaël Greff is the R&D Director of A-Volute and received his Ph. D degree from Université Paris VI in 2008. His research interests are focused on spatial audio and real-time digital signal processing.