Real-Time Monophonic and Polyphonic Audio Classification from Power Spectra

Maxime Baelde, Christophe Biernacki, Raphaël Greff

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 normalized power spectrum (NPS) 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 the prototypes, revealing a particularly efficient computation time and classification accuracy trade-off. The proposed method, called RARE (for Real-time Audio Recognition Engine) reveals encouraging results both in monophonic and polyphonic classification tasks on benchmark and owned datasets, including also the targeted real-time situation. In particular, this method benefits from several advantages compared to the state-of-the-art methods including a reduced training time, no feature extraction, the ability to control the computation - accuracy trade-off and no training on already mixed sounds for polyphonic classification.

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

*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://mbaelde.lille.inria.fr (Maxime Baelde)

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 main topics. The most prominent topic is automatic speech recognition (ASR) whose goal is to identify speech (in particular phonemes) in audio recording [2, 3]. The next topic is music information retrieval aiming at analyzing musics and extracting relevant information such as the musical genre [4] (rock, classical, *etc.*) or the different instruments [5]. The last topic is environmental sound recognition which aims at recognizing sounds such as airplanes, dog coughs, trains, gunshots, *etc.* [6, 7, 8]. 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 [9]: *speed* and *latency*. First, speed is the time to make the decision and is related to how many time frames the system uses (past and future). 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 with a set of (expected) discriminant features. The usual audio features are known as *audio descriptors* [10], 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. It should be worth noting that temporal features can also be extracted by considering the temporal evolution of the spectral and cepstral features [5]. 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 [11].

Several supervised learning methods have been applied to monophonic SED using the previous features. What is called "monophonic" in the SED field of research corresponds to a multi-class single-label classification task. 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

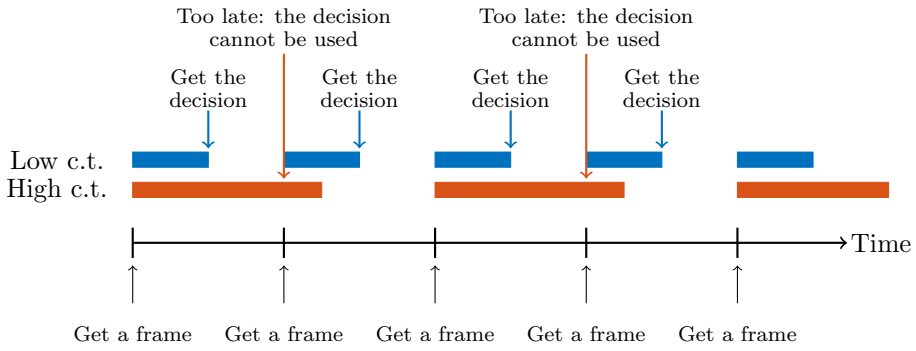

Figure 1: Illustration of the latency of a real-time audio classification system. A real-time process gets the frames at specific time clocks (black bottom arrows). If the computation time (c.t.) is low (Low c.t., blue filled rectangles), the decision will be used by the system because it will be available 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.

applied with MFCC for gunshot detection [12, 13] and real-time voice detection in medical application [14, 15]. Second, Hidden Markov Models (HMM) are used to model the temporal continuity of audio signals, alone or coupled with GMM. Bietti *et al.* modeled the normalized audio spectra using HMM in an online setting [16], whereas Heittola *et al.* fed a HMM with histograms of MFCC [17]. 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 [18] and [19] for surveillance application, and also in [20] for meeting room sounds. Finally, Neural Networks (NN) – and their deep variants such as Deep Convolutional/Recurrent Neural Networks (DC/RNN) – have recently attracted the attention of researchers. Biondi *et al.* used a normalized spectrum as input for a standard NN [21] and Dadula *et al.* considered the MFCC [22]. Palaz *et al.* [23] and Piczak [6] considered directly the raw audio signal and the spectrogram, respectively, using a deep architecture (DCNN). Other authors considered also Random Forest [24] using more contextual information.

Other algorithms are used in case of polyphonic SED. What is called "polyphonic"[1] in the SED field of research corresponds to a multi-class multi-label classification task. It is often cast to a multi-class single-label classification task but the output of the system is thresholded to predict the active classes. Çakır *et al.* [25] developed a Convolutional Recurrent Neural Network (CRNN), taking advantages of the two structures (convolutional and recurrent). However,

[1]Polyphonic SED must be distinguished from polyphonic music, the latter referring to playing different melodies simultaneously.

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 algorithms are considered: PLCA (Probabilistic Latent Component Analysis) [26] and NMF (Nonnegative Matrix Factorization) [27]. Benetos *et al.* [28] 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 [29] or as coupled training of sound spectra and class annotation [30]. Heittola *et al.* [31] 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 (which are 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 (BR, learn a classifier for each label individually: simple but does not learn correlations between labels) or Label Powerset (LP, consider multi-label output as a new single-label: complex and combinatorial) [32], which are far from optimal solutions. More advanced multi-label methods also exist such as RAKEL [33] (combination of LP classifiers) or Classifier Chains [34] (chain of BR classifiers) and their ensemble versions. 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 (and not audio descriptors) for the classification.

The method developed in this paper, called RARE (for Real-time Audio Recognition Engine) is based on a generative model of the whole normalized power spectrum (NPS), 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 time-varying MAP for the online classification. However, using the model as is, 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 trade-off between accuracy and computation time. The proposed method is a worthwhile extension of the one presented in our two previous conference papers [35] and [36], 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. **Features**: 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.

2. Monophonic modeling of the classes

The monophonic modeling is a mandatory step toward the polyphonic modeling, but the main goal of this work is to perform polyphonic classification. Indeed, the polyphonic modeling will be build upon the monophonic one in a straighforward manner (see Section 3).

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[2]). The classification uses only 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

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

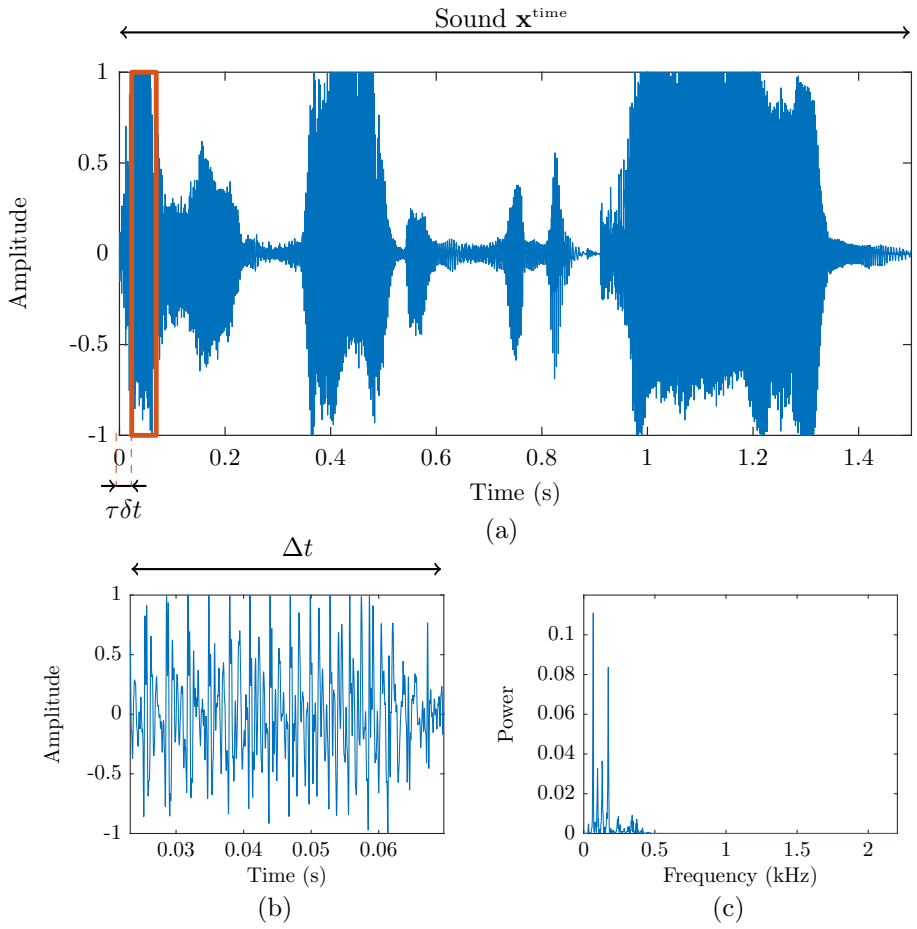

Figure 2: (a) Example with a real sound $\mathbf{x}^{\text{time}}$ containing a voice. This sound is split into time frames $\mathbf{x}_\tau^{\text{time}}$ (red rectangle) of size Δt and shifted by $\tau\delta t$ (b), and each time frame is converted into the normalized power spectrum $\mathbf{x}_\tau$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

$p\left(z\middle|f\left(\mathbf{x}^{\text{time}}_{[t-\Delta T,t]}\right),t\right)$ 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*. However the overall sound $\mathbf{x}^{\text{time}}$ can contain instances of difference classes.

Several constraints are considered for real-time classification. The first is the time related to the data acquisition which constrains 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 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 first of all 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 $\mathbf{x}_{\tau t}^{\text{time}}$ is thus a portion of $\mathbf{x}_{[t-\Delta T,t]}^{\text{time}}$ 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:

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

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

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

where $\mathbf{x}_{\tau t}^{\text{freq}} \in \mathbb{C}^B$ is the Fourier Transform of $\mathbf{x}_{\tau t}^{\text{time}}$ (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 model because a similar generative process will be used in the polyphonic case that simplifies the classification task. Indeed it considers only the monophonic dataset to construct the polyphonic one. This is a significant 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: $(\mathbf{x}_{1t},...,\mathbf{x}_{Nt}) = f\left(\mathbf{x}_{[t-\Delta T,t]}^{\text{time}}\right)$,

where $\text{Mult}_K(1, \mathbf{p})$ is a multinomial distribution over K categories and 1 draw with probabilities $\mathbf{p} = (p_k)_{k=1}^K \left(p_k > 0,\ \sum_{k=1}^K p_k = 1\right)$ which means that the probability of $z = k$ is p_k, 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:

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

The generative process models the joint distribution $p\left(\mathbf{x}_{1t},...,\mathbf{x}_{Nt},t,\mathbf{x}^{\text{time}},z\right)$, however only the conditional $p\left(\mathbf{x}_{1t},...,\mathbf{x}_{Nt}\middle|z,t\right)$ is useful for the monophonic classification task. The full signal $\mathbf{x}^{\text{time}}$ is redundant since the frames are extracted from it so that it will not be used in the remainder. 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 [35]) or as an independence assumption. It should worth noting that this assumption can be challenging since in practice acoustic signals are temporally correlated: however for computational purpose this hypothesis is quite convenient and used in some references like [12]. The second hypothesis is an hypothesis of *stationarity in time*, so 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 could be used: the Dirichlet kernel – though it is not flexible enough for our purpose –, a mixture of Dirichlet kernel – which is more flexible but in a real-time context the involved computational load is to high – and finally a very classical Gaussian mixture models like in [35]. We have tried these three kernels (results no reported here) but the best results were obtained with the very simple multinomial kernel that we present now. The *multinomial kernel* quantizes 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 later in Section 4, because only dot products are required to compute the distribution (the normalization constant is the same for each kernel). Consider $\mathbf{x}_{(i)}^{(q)} \in \mathbb{N}^B$ (where B is the number of frequency bins) 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 so-called approximated kernel by:

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

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 $p_{\mathrm{sound}}\left(\cdot|z\right)$ 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 is 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

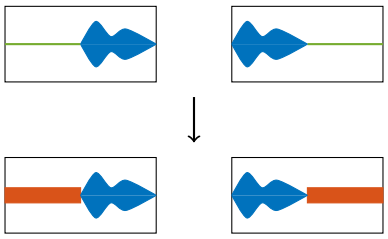

Figure 3: We illustrate the following phenomenon. In real-time, the sound (blue filled curve) neither begins at the beginning of the frame (top left) nor ends at the ending (top right). This is why we add (bottom left and right) Gaussian white noise (red rectangle) so as to fill the "silence" (green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

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 a clear advantage of the proposal. The case for a mixture of two sources is considered but it can be straighforwardly extended to a greated number of sources: 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:

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

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:

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

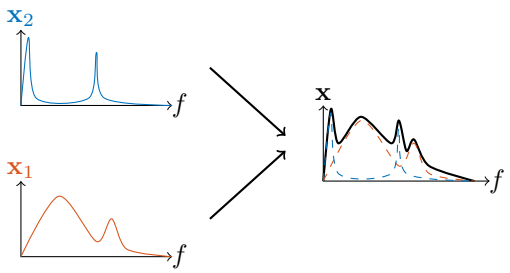

Figure 4: Consider that two power spectra $\mathbf{x}_1$ (red, left bottom curve) and $\mathbf{x}_2$ (left top blue curve) are mixed. The resulting spectrum $\mathbf{x} = \phi\mathbf{x}_1 + (1-\phi)\mathbf{x}_2$ (thick black right curve) will depend on the relative powers of the two spectra, represented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

where $\mathbf{x} \cdot \mathbf{y}$ is the element-wise multiplication between some vectors $\mathbf{x}$ 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:

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

where $\mathbf{x}_{1\tau t}^{\text{freq}}$ is defined as in the monophonic feature design section 2.2. A detailed explanation is available in Appendix B. As already said, this framework can thus 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 times: here, for the simplicity of the presentation, the underlying synchronization of the sounds is not presented but the proposed method actually takes it into account.

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 $z_1, z_2 \sim \text{Mult}_K(1, \mathbf{p})$,
- **Random**: Draw independently two different sounds: $\mathbf{x}_1^{\text{time}} \sim p_{\text{sound}}(\cdot|z_1)$ and $\mathbf{x}_2^{\text{time}} \sim p_{\text{sound}}(\cdot|z_2)$
- **Random**: Draw a time index: $t \sim \mathcal{U}([0, T])$
- **Deterministic**: Compute the features: $(\mathbf{x}_{1t}, ..., \mathbf{x}_{Nt}) = \boldsymbol{\phi}_t \cdot f\left(\mathbf{x}_{1[t-\Delta T,t]}^{\text{time}}\right) + (\mathbf{1} - \boldsymbol{\phi}_t) \cdot f\left(\mathbf{x}_{2[t-\Delta T,t]}^{\text{time}}\right)$.

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 pairwise normalized spectra are computed by weighting two normalized monophonic spectra by the corresponding proportion $\phi_{(i)}$ (defined in Eq. 19), with 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 $\phi_{(i)}$ 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 for both monophonic and polyphonic sounds. 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 fast to compute). 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 prototypes), and therefore the computation time can be 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 prototypes n.

4.2. Hierarchical clustering of the monophonic models

A model reduction technique was already considered in [30] 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.

The idea is to perform class-wise hierarchical clustering of the prototypes parameterized by the $\mathbf{p}_{(i)}^{(q)}$ 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 standard Hellinger distance [37] since the elements to cluster are (discrete) probability distributions. The elements are linked using the Ward criterion [38] – 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 prototypes n_z over the number of prototypes 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, $\mathcal{I}_{i'} = \left\{i : \mathbf{p}_{(i)}^{(q)} \in \mathcal{C}_{i'}\right\}$ denotes the set of the

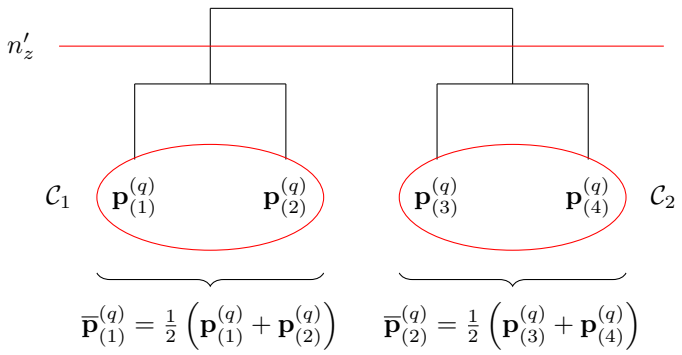

Figure 5: Illustration of the complexity reduction using hierarchical clustering. Consider that in a given class there are 4 models parameterized by $\mathbf{p}^{(q)}_{(1)}, ..., \mathbf{p}^{(q)}_{(4)}$. The hierarchical clustering results in a tree representing how the models are structured. Choose 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 $\overline{\mathbf{p}}^{(q)}_{(2)}$.

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 reduction method improves the results up to a certain reduction value and suggests that the procedure limits the influence of overfitting.

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 $r^{(0)}_z$. 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.

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 $\overline{\mathbf{p}}^{(q)}_{(i')}$ described in

Table 1: Summary of the datasets used for the experiments. The polyphonic complexity is the maximum number of classes mixed simultaneously: when this complexity equals 2 that means there are at most 2 different classes are mixed simultaneously.

Dataset name	Type	Number of classes	Polyphonic complexity
A-Volute	Monophonic	5	1
ESC-10	Monophonic	10	1
A-Volute	Polyphonic	5	2
Battlefield	Polyphonic	3	3
TUT-SED	Polyphonic	6	4

Section 4.2. The goal is to construct a new learning set containing relevant mixtures models. The method consists in computing pairwise mixture models from the different classes and reduces 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.

Remark: We emphasize on the hierarchical clustering as a means to reduce the computation load, however when this particular clustering is not suited (for large datasets for instance), a more usual clustering algorithm such as k-means can be used (as we will see in the experiments).

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 [39], composed of 400 sounds divided into 10 classes. For polyphonic classification, the mixtures from the A-Volute dataset (created using the protocole of Section 3.3) 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 [40] dataset is also considered, which contains real-life recordings in a street context. A summary of the datasets is available in Table 1.

Parameters tuning. All the sounds are resampled to $F = 44.1$kHz and centered. The considered frame size is $\Delta t = 2048$ samples (46.4ms) and the time shift is fixed to $\delta t = 512$ samples (11.6ms). Other values of Δt were tested to select the best (the results are not reported). The number of frequency bins $B = \Delta t/2 + 1$ (the highest half of the spectrum is discarded according to the Nyquist theorem). The quantization of the spectra is set to $q = B$: in practice, this seems to be an optimal number, but values above B are also good. The learning sets are divided using the v-fold scheme to perform cross-validation (with $v = 5$): precisely the split is performed on the sounds. Frame sequences

length is set to $N = 40$: this corresponds to approximately 500ms of audio data to make 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 (E.R.) and segment-based F1 score (F1) (see [41] for a detailed explanation of these metrics): a large F1 and a low E.R. mean that the system has very good performances. The F1 score is an extension of the accuracy since in a monophonic setting the F1 score becomes the accuracy (and the error rate becomes the misclassification rate).

Monophonic baseline systems. The proposed method, denoted by RARE, for monophonic classification is compared with several baseline systems. The first is a GMM with 20 and 30 components per classes using 20 MFCC and their first and second derivatives, inspired by [12]. The second is also a GMM with 20 and 30 components per classes but using the proposed normalized power spectrum (NPS) as a feature. The third is a multi-class Support Vector Machine trained using error-correcting ouput code with one-versus-one coding design. The fourth is a Deep Convolutional Neural Network (DCNN) using log-mel spectrograms inspired by [6], trained on 50 epochs. Human listening results were gathered for the two monophonic datasets: the results for ESC-10 are available in [39]. 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 RARE was also compared with a neural network based method, called CRNN (Convolutional Recurrent Neural Network) inspired by [25], trained on 100 epochs. 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. As this method usually uses dataset with already mixed sounds, we created artificial mixtures on the A-Volute dataset to train the network. Once the network is trained, a threshold th is used to predict the labels based on the output scores: the default value is 0.5 but we also tested other values (from 0.1 to 0.5).

Remark: polyphonic dataset creation. The polyphonic training data was created by considering a reduced monophonic training dataset. Indeed, the number of possible mixtures from the original dataset was too large to fit a standard computer memory, so that the mixtures are created using a reduced dataset. We define r_{mono} the reduction factor of the monophonic prototypes and r_{poly} the reduction factor of the polyphonic prototypes. For $r_{\text{mono}} = 100$, the monophonic dataset is reduced by a factor 100 and mixtures were created using this reduced dataset. Then the mixtures were classified using the reduced monophonic prototypes and the polyphonic ones. The CRNN has to be trained using a dataset containing mixed sounds, this is why artificial mixtures were created on the A-Volute dataset. The parameter N_{mixt} controls the number of frames from each class mixed together: if $N_{\text{mixt}} = 200$, there were 200 frames from the first class mixed with 200 frames from the second class.

Table 2: Monophonic classification task ($\Delta t = 2048$). Summary of the results for the A-Volute dataset in term of accuracy, training time (in seconds) and testing time per frame (in milliseconds) for different methods. RARE stands for our proposed method and NPS is the normalized power spectrum.

Method	Features	Param.	Accuracy (%)	Training t.	Testing t.
RARE	NPS	-	82.5 (6.0)	7.6×10^1	2.4×10^2
RARE	NPS	$r = 10$	83.1 (5.6)	1.0×10^3	2.5×10^1
		$r = 100$	79.9 (5.7)		2.7×10^0
GMM [12]	MFCC	20 comp.	86.6 (8.5)	1.3×10^2	3.0×10^0
		30 comp.	84.5 (7.1)	1.9×10^2	4.1×10^0
GMM [12]	NPS	20 comp.	77.2 (7.3)	7.3×10^2	2.8×10^3
		30 comp.	88.0 (7.5)	1.3×10^2	4.0×10^3
SVM	NPS	-	36.9 (9.2)	1.2×10^4	2.9×10^1
DCNN [6]	MFCC	-	61.6 (16.0)	1.4×10^3	3.1×10^0
CRNN [25]	MFCC	-	19.2 (1.6)	4.0×10^2	2.9×10^1
Human	-	-	91.7	-	-

5.2. Results on the monophonic classification

Proposed method without reduction. Tables 2 and 3 summarize the results for the A-Volute and ESC-10 datasets respectively for: the different methods (RARE, GMM, SVM and DCNN), the different features (NPS or MFCC) and the corresponding tuning parameter (if any). The results are the accuracy (in %), the training time (in seconds) and the testing time (in milliseconds). In the case of the GMM, the hyperparameter to tune is the number of components in the mixture. The reduction factor r in the proposed method is not a truly hyperparameter since it controls the trade-off between accuracy and the testing time (see the next paragraph). In term of accuracy, the best method is the GMM with a relatively few number of components but the proposed method is the second one and achieves good results compared to the other methods, in particular the DCNN and the SVM. We can see that on the A-Volute database, the GMM with NPS performs quite well compared to the GMM with MFCC, which indicates that the NPS is an effective space to describe sounds. The results on the ESC-10 dataset for the proposed method decrease because one class (over the 10) is not well recognized, but overall the results are similar than for the A-Volute dataset.

Proposed method with reduction. The model reduction algorithm seems to be very efficient because the number of prototypes can be reduced by a factor 100 without loosing too much accuracy, but it decreases the testing time dramatically (see last column of Tables 2 and 3). For a reduction factor of $r = 10$ the proposed method can be used in real-time because the testing time is lower than 50ms (which corresponds to the frame length). Moreover, for reduction factors above to 10, the results decrease: this may mean that the

Table 3: Monophonic classification task ($\Delta t = 2048$). Summary of the results for the ESC-10 dataset in term of accuracy, training time (in seconds) and testing time per frame (in milliseconds) for different methods. RARE stands for our proposed method and NPS is the normalized power spectrum.

Method	Features	Param.	Accuracy (%)	Training t.	Testing t.
RARE	NPS	-	64.7 (3.2)	8.2×10^1	3.7×10^2
RARE	NPS	$r = 10$	67.5 (4.4)	7.9×10^2	3.7×10^1
		$r = 100$	67.1 (5.1)		4.0×10^0
GMM [12]	MFCC	20 comp.	78.1 (4.0)	2.2×10^2	3.0×10^0
		30 comp.	78.3 (3.5)	3.0×10^2	3.5×10^0
GMM [12]	NPS	20 comp.	56.7 (3.5)	1.2×10^3	5.7×10^3
		30 comp.	56.1 (4.2)	2.0×10^3	8.0×10^3
SVM	NPS	-	46.3 (4.4)	2.2×10^4	4.7×10^1
DCNN [6]	MFCC	-	40.7 (6.0)	2.1×10^3	1.3×10^0
CRNN [25]	MFCC	-	10.0 (0.0)	6.0×10^2	2.6×10^1
Human	-	-	95.7	-	-

method overfits when there is no reduction and this reduction tends to improve the results. However for a large reduction factor there are less prototypes to compute the classification rule so that the results decrease.

Pros and Cons. The proposed method has several advantages compared to the state-of-the-art methods. One advantage is that there is no feature extraction. The monophonic proposed method is not the best method in term of accuracy: a fine tuned GMM can have better results. However, the GMM needs to have a number of components set beforehand or optimized using a model selection criterion (such as Bayesian Information Criterion or the classification accuracy), which increases the training time. The DCNN has a complex network architecture which is hard to fine tune and which is time consuming. The proposed method has this second advantage that the training time is small compared to the other methods. Finally, with the model preprocessing, the proposed method can be used in real-time. To summarize, the proposed method is the second best model (just before the fine tune GMM) and has the ability to control the trade-off between computation time and accuracy.

5.3. Results on the polyphonic classification

Proposed method without reduction. Table 4 summarizes the results on the A-Volute dataset for the polyphonic classification task in the same way as in the monophonic classification task. This table shows that the proposed method outperforms the CRNN in term of scores and testing time. When the reduction r_{mono} decreases, the scores are better since more prototypes are used to construct the polyphonic dataset, but the testing time increases dramatically.

We also see that the threshold th of the CRNN controls a trade-off between the F1 and E.R.: the default value 0.5 seems to be the optimal trade-off. However the results are far worst than our method.

Tables 5 and 6 summarize the results for the Battlefield and TUT-SED dataset like previously. The main difference between these datasets and the A-Volute dataset is that mixtures are already present in the datasets. This is why the proposed method uses only a reduction factor r_{poly} for the polyphonic classes. For the Battlefield dataset, the proposed method performs better both in term of scores (F1 and error rate) and times (training and testing). For the TUT-SED dataset, with no reduction, the proposed method does not reach the real-time objective and performs badly: since this database contains 6 classes with at most 4 classes mixed together at the same time, there are equivalently around 50 classes which are effectively present so that a completely random decision would have an accuracy of approximately 2%. It is then now interesting to express the following reduction technique to reach the real-time target.

Proposed method with reduction. For a smaller r_{mono}, the proposed method for the A-Volute dataset must use a polyphonic reduction step to work in real-time. Like the monophonic reduction, the polyphonic reduction increases the results so that without reduction there may be overfitting of the proposed method. The Battlefield dataset shows a similar behaviour: better scores for a reduction of a factor 10 and then a decrease for a large reduction factor. The TUT-SED dataset was reduced using a k-means algorithm since the hierarchical clustering was not able to reduce a large amount of prototypes (more than 100k): the results are better with the reduction, which shows again that the reduction brings accuracy benefits in our case.

Pros and cons. All the previous methods used for monophonic classification are not designed to work on polyphonic classification using a monophonic dataset, contrary to our proposed method (RARE), which is a major advantage. Indeed the proposed method does not have to learn the mixtures of sounds but only the individual sounds. Moreover, the reduction of the complexity has a regularizing effect which is very effective on the TUT-SED database for instance. Even with multiple reduction of the complexity, the training time of our proposed method is still lower than the CRNN, which is trained using iterative algorithms such as stochastic gradient descent.

6. Conclusion

This work dealt with a new method for monophonic and polyphonic real-time audio sources classification. The method used the whole power spectrum instead of predefined audio descriptors, which is also useful for polyphonic events thanks to the additivity of uncorrelated spectra. The classification was 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.

Table 4: Polyphonic classification task ($\Delta t = 2048$). Summary of the results for the A-Volute dataset in term of F1 score (F1), error rate (E.R.), training time (in seconds) and testing time per frame (in milliseconds) for different methods. r_{mono} is the reduction factor of the monophonic prototypes and r_{poly} is the reduction factor of the polyphonic prototypes. th is the threshold of the CRNN output. A large F1 and low E.R. means good performances. RARE stands for our proposed method.

Method	Tuning param.		F1	E.R.	Training t.	Testing t.
RARE	$r_{\text{mono}} = 400$	-	69.4 (3.2)	46.2 (3.2)	7.7×10^1	6.4×10^0
RARE	$r_{\text{mono}} = 100$	- $r_{\text{poly}} = 20$	72.3 (4.8) 71.4 (3.8)	43.1 (5.4) 41.3 (4.5)	1.9×10^4	7.9×10^2 4.2×10^1
RARE	$r_{\text{mono}} = 50$	-	74.5 (4.7)	40.6 (5.1)	7.7×10^1	3.0×10^3
CRNN [25]	$N_{\text{mixt}} = 200$	$th = 0.5$ $th = 0.2$ $th = 0.1$	39.8 (26.5) 56.4 (3.8) 55.9 (2.1)	70.3 (20.0) 114.4 (6.0) 141.7 (15.0)	4.8×10^3	2.8×10^1
CRNN [25]	$N_{\text{mixt}} = 300$	th $= 0.5$ $th = 0.2$ $th = 0.1$	53.1 (0.6) 56.2 (1.9) 57.1 (0.1)	60.3 (0.6) 141.1 (14.4) 149.1 (1.9)	1.3×10^4	3.1×10^1

Table 5: Polyphonic classification task ($\Delta t = 2048$). Summary of the results for the Battlefield dataset in term of F1 score (F1), error rate (E.R.), training time (in seconds) and testing time per frame (in milliseconds) for different methods. r_{poly} is the reduction factor of the polyphonic prototypes. th is the threshold of the CRNN output. A large F1 and low E.R. means good performances. RARE stands for our proposed method.

Method	Tuning param.	F1	E.R.	Training t.	Testing t.
RARE	-	66.0 (4.3)	44.5 (6.3)	5.1×10^1	9.4×10^1
RARE	$r_{\text{poly}} = 10$ $r_{\text{poly}} = 50$	69.2 (2.8) 67.9 (1.8)	40.9 (2.6) 42.3 (2.2)	1.2×10^2	1.0×10^1 2.1×10^0
CRNN [25]	$th = 0.5$ $th = 0.3$ $th = 0.2$ $th = 0.1$	61.8 (3.4) 63.5 (4.0) 65.2 (2.8) 56.1 (1.5)	54.4 (4.4) 77.9 (17.3) 85.1 (7.0) 156.7 (9.4)	1.8×10^2	2.6×10^1

Table 6: Polyphonic classification task ($\Delta t = 2048$). Summary of the results for the TUT-SED dataset in term of F1 score (F1), error rate (E.R.), training time (in seconds) and testing time per frame (in milliseconds) for different methods. r_{poly} is the reduction factor of the polyphonic prototypes. th is the threshold of the CRNN output. A large F1 and low E.R. means good performances. RARE stands for our proposed method.

Method	Tuning param.	F1	E.R.	Training t.	Testing t.
RARE	-	30.1 (4.1)	85.4 (9.8)	5.4×10^2	1.3×10^3
RARE	$r_{\text{poly}} = 100$	40.2 (9.8)	60.1 (9.7)	8.8×10^2	1.6×10^1
	$r_{\text{poly}} = 1000$	47.9 (11.9)	59.1 (11.3)	8.8×10^2	2.5×10^0
CRNN [25]	$th = 0.5$	0.0 (0.0)	100.0 (0.0)	2.3×10^3	2.9×10^1
	$th = 0.4$	7.5 (15.0)	100.0 (0.0)		
	$th = 0.3$	25.2 (16.9)	99.1 (2.5)		
	$th = 0.2$	39.1 (7.7)	137.4 (57.9)		
	$th = 0.1$	34.5 (5.2)	308.4 (122.1)		

Contrary to other methods like neural networks, this technique modeled both monophonic and polyphonic sources in a single framework.

As shown in the experiments, the polyphonic classification performed quite well on owned and benchmark datasets and outperforms the CRNN. Thanks to the reduction of the complexity, the method has a low computation time and can be used in real-time. This polyphonic classsification is built on monophonic sounds and does not have to learn from already mixed sounds, which is a major advantage compared to other methods like neural networks or Gaussian mixture models for instance. As mentionned in the experiments, the reduction of the complexity has a regularizing effect in addition to an efficient computation time - accuracy trade-off.

Since the NPS feature was essentially designed to handle the polyphonic classification, it is not surprising that the monophonic classification does not fully compete with usual monophonic classification algorithms such as a fine tuned GMM. Moreover, there may be overfitting of our method because of the nonparametric estimation using kernels. As the proposed method is flexible regarding the choice of the kernel, we also tested the Dirichlet kernel and the bGMM kernel (which is the GMM for binned data [35]). The Dirichlet kernel is constructed by considering that the NPS is the mode of the Dirichlet kernel. This kernel does not improve the results as the accuracy is 48.30% for a real-time application (after a reduction by 20). The bGMM is learned using the quantized NPS and by selecting the number of components using ICL [42] (Integrated Classification Likelihood). This kernel does not improve the results either since the accuracy is 82.7%. A future area of research would be to change the estimation of the condition distribution of Eq. 11 and 24, using a semiparametric estimation for instance. However it is worth noting that the considered nonparametric estimation using multinomial kernels allows to derive an efficient computation - accuracy trade-off.

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:

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

The framing operation is linear so that:

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

For a given frame:

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

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 $\mathbf{x}^{\mathrm{freq}}_{1\tau t}$ and $\mathbf{x}^{\mathrm{freq}}_{2\tau t}$ the complex spectra, the sum of these spectra $\mathbf{x}^{\mathrm{freq}}_{\tau t}$ is:

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

The modeling disclosed in Section 2.4 requires to deal with a normalized version composed of the elements $|\mathbf{x}^{\mathrm{freq}}_{\tau t}|^2$ as in Eq. (5). Two sources from different classes are assumed to be *uncorrelated signals* (a common assumption in signal separation [43]), meaning that the power spectrum of the sum is approximately the sum of the power spectra:

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

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

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

Define $P_{1\tau t} = ||\mathbf{x}_{1\tau t}^{\text{freq}}||^2$ and $P_{2\tau t} = ||\mathbf{x}_{2\tau t}^{\text{freq}}||^2$ 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] X. Liu, J. Geng, H. Ling, Y.-m. Cheung, Attention guided deep audio-face fusion for efficient speaker naming, Pattern Recognition 88 (2019) 557–568.

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

[5] 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.

[6] 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.

[7] P. Hu, W. Liu, W. Jiang, Z. Yang, Latent topic model for audio retrieval, Pattern Recognition 47 (3) (2014) 1138–1143.

[8] J. Beltrán, E. Chávez, J. Favela, Scalable identification of mixed environmental sounds, recorded from heterogeneous sources, Pattern Recognition Letters 68 (2015) 153–160.

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

[16] 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.

[17] 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.

[18] S. Lecomte, R. Lengellé, 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.

[19] 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.

[20] A. Temko, C. Nadeu, Classification of acoustic events using SVM-based clustering schemes, Pattern Recognition 39 (4) (2006) 682–694.

[21] 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.

[22] 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.

[23] 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.

[24] X. Xia, R. Togneri, F. Sohel, D. Huang, Random forest classification based acoustic event detection utilizing contextual-information and bottleneck features, Pattern Recognition 81 (2018) 1–13.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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).

[33] G. Tsoumakas, I. Vlahavas, Random k-labelsets: An ensemble method for multilabel classification, Machine Learning: ECML 2007 4701 (2007) 406–417.

[34] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning 85 (3) (2011) 333.

[35] 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.

[36] 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.

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

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

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

[40] 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.

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

[42] C. Biernacki, G. Celeux, G. Govaert, Assessing a Mixture Model for Clustering withthe Integrated Classification Likelihood, Tech. Rep. RR-3521, INRIA (1998).

[43] E. M. Grais, M. U. Sen, H. Erdogan, Deep neural networks for single channel source separation, in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 3734–3738.

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.