# Deep Belief Network based audio classification for construction sites monitoring

Michele Scarpiniti[a,*], Francesco Colasante[b], Simone Di Tanna[b], Marco Ciancia[b], Yong-Cheol Lee[c], Aurelio Uncini[a]

[a] *Department of Information Engineering, Electronics and Telecommunications (DIET), Sapienza University of Rome, Via Eudossiana 18, 00184 Rome, Italy*

[b] *Department of Computer, Control and Management Engineering (DIAG), Sapienza University of Rome, Italy, Via Ariosto 25, 00185 Rome, Italy*

[c] *Department of Construction Management, Louisiana State University, Baton Rouge, USA*

[*] Corresponding author. Phone: +39 06 44585869, Fax: +39 06 44585632.

*Email addresses:* `michele.scarpiniti@uniroma1.it` (Michele Scarpiniti), `colasante.1760608@studenti.uniroma1.it` (Francesco Colasante), `ditanna.1765243@studenti.uniroma1.it` (Simone Di Tanna), `ciancia.1741186@studenti.uniroma1.it` (Marco Ciancia), `yclee@lsu.edu` (Yong-Cheol Lee), `aurelio.uncini@uniroma1.it` (Aurelio Uncini)

## Abstract

In this paper, we propose a Deep Belief Network (DBN) based approach for the classification of audio signals to improve work activity identification and remote surveillance of construction projects. The aim of the work is to obtain an accurate and flexible tool for consistently executing and managing the unmanned monitoring of construction sites by using distributed acoustic sensors. In this paper, ten classes of multiple construction equipment and tools, frequently and broadly used in construction sites, have been collected and examined to conduct and validate the proposed approach. The input provided to the DBN consists in the concatenation of several statistics evaluated by a set of spectral features, like MFCCs and mel-scaled spectrogram. The proposed architecture, along with the preprocessing and the feature extraction steps, has been described in details while the effectiveness of the proposed idea has been demonstrated by some numerical results, evaluated by using real-world recordings. The final overall accuracy on the test set is up to 98% and is a significantly improved performance compared to other state-of-the-are approaches. A practical and real-time application of the presented method has been also proposed in order to apply the classification scheme to sound data recorded in different environmental scenarios.

*Keywords:* Deep learning, Deep Belief Network (DBN), Audio processing, Environmental sound classification, Construction monitoring.

## 1. Introduction

Environmental sound classification (ESC) (Piczak, 2015b) is a challenging field of research aiming at identifying a large variety of sounds related to natural sounds, like animals (dogs, birds, etc.), weather conditions (rain, wind, etc.), vehicles (cars, trucks, train, etc.), domestic noise (washing-machine, glass-breaking, clock-tick, etc.) and many others. However, often sounds are regarding urban environments, and hence sounds are usually related to various non-human events in normal day-to-day life. Although there exist lots of literature on the classification of speech and music (Fu et al., 2011), ESC is not yet a mature field of application. The main difference between speech or music signals and environmental sounds is that these last signals do not posses any common structure (Boddapati et al., 2017), making difficult the classification (Barchiesi et al., 2015).

Generally, due to the flexibility and cheapness of acoustic sensors, ESC can be a robust and adaptive approach for the environmental monitoring (Atrey et al., 2006). When we consider generic outdoor scenarios, an automatic monitoring system based on a microphone array would be an invaluable tool in assessing and controlling any type of situation occurring in the environment (Sallai et al., 2011; Scardapane et al., 2015). The general idea of this kind of approach is to capture the audio signal emitted from a particular direction by set of microphones, extract a set of peculiar and discriminative features from the recorded signals, and then apply some (audio) classification algorithms to identify the recorded sound (Abu-El-Quran, 2006).

A specific application of ESC can be for the classification of audio sounds of construction work and equipment operations (Navon & Sacks, 2007). According to the previous research studies, the steady and real-time monitoring of construction processes and tasks in the field reduces one of the most salient risks in a construction project's uncertainty (Cheng & Teizer, 2013). In order to reduce such uncertainty, several studies explored that the systematic monitoring of a construction project helps project managers by providing construction field information in a timely manner and enabling them to identify urgent issues and promptly respond to unexpected problems (Golparvar-Fard et al., 2015). At the moment, several methods have been suggested (Sherafat et al., 2020) and the video-based approaches are currently the preferable ones (Kim et al., 2019; Khosrowpour et al., 2014).

However, recently diverse studies investigated and showed the potentials of audio-based construction site monitoring, which can be a promising method for advanced unmanned field monitoring (Sherafat et al., 2020, 2019). It has been found that this new approach can obtain satisfactory performance and reliability (Cho et al., 2017; Sharan & Moir, 2016; Lee et al., 2020). In addition, an audio-based approach entails various opportunities to be analyzed with construction project data and other resources such as visuals to improve accurate project monitoring and progress surveillance. Since sounds generally consist of various discriminant features extracted from recorded signals, the identification of suitable and highly accurate classifiers that can accurately recognize and understand feature characteristics is of fundamental importance (Zhang et al., 2018).

Numerous methods have been developed for ESC (Ahmad et al., 2020) and acoustic environmental monitoring (Scardapane et al., 2015). Machine Learning (ML) approaches are very common for this kind of topic (Heittola et al., 2018; Chachada & Jay Kuo, 2013). Specifically, many works propose to use several methods like, to cite a few of them: Support Vector Machines (SVMs), *k*-Nearest Neighbors (*k*-NN), Artificial Neural Networks (ANNs), Hidden Markov Models (HMMs) and Gaussian Mixture Models (GMMs) (Sharan & Moir, 2016; Barchiesi et al., 2015).

Recently, Deep Learning (DL) (Goodfellow et al., 2016) provided very promising results overcoming those of traditional ML approaches (Gencoglu et al., 2014; Abeßer, 2020). Differently from the traditional shallow ML algorithms, like GMM, HMM and SVM, the deep models have many hidden layers of different abstract representations, which well fit in with the human acoustic perception that appears to use many layers of feature extractors and event detectors. The most used model in the field of DL is the Convolutional Neural Network (CNN) (Salamon & Bello, 2017; Piczak, 2015a; Abdoli et al., 2019) and related variants (Medhat et al., 2020; Li et al., 2018), followed by the Deep Recurrent Neural Network (DRNN) (Phan et al., 2017; Scarpiniti et al., 2020) or a combination of the two approaches (Sang et al., 2018). A comparison of several traditional and deep learning algorithms for construction activity monitoring can be found in (Lee et al., 2020).

A very interesting approach, not fully exploited in audio classification, consists in the use of Deep Belief Networks (DBNs) that show a powerful representation of audio signals (Mohamed et al., 2012; Hamel & Eck, 2010). A DBN is an architecture composed of multiple layers of latent variables, called hidden units, with connections between the layers but not between units within each layer (Goodfellow et al., 2016).

Although DBNs have demonstrated good results in audio classification, this kind of architecture has been used only for structured sound data as speech (Mohamed et al., 2012; Hinton et al., 2012) and music (Hamel & Eck, 2010; Xue & Su, 2015), but it has not been considered for other kind of data as in ESC. The main feature of DBN that makes it very suitable for the classification of audio data in construction sites, is that it presents a reasonable computational complexity compared to other DL methods, but, despite all this, providing very encouraging performance.

Motivated by these considerations, the main contributions of this paper are:

- we propose a DBN-based approach for the enhanced classification of audio signals captured in construction sites. We have chosen the DBN because it has a capability to improve the classification of audio signals with a superior classification accuracy. Although the training time is not negligible (even if less than other DL approaches), it presents an inference time comparable or lower than traditional ML approaches, but the accuracy is unquestionably greater. Moreover, DBN shows several advantages compared to other methods in that pre-training improves the model performance by avoiding overfitting and enhancing the model generalization (Pinaya et al., 2016). In time critical applications, like classification of work activities in construction sites, a fast, accurate, and reliable inference phase is of primary importance;

- we propose a statistically optimized set of features obtained as a series of statistics evaluated from mel-frequency cepstral coefficients (MFCCs) (Chu et al., 2009). In order to provide a robust set of features, our method assesses the minimum and maximum values, alongside with the mean, standard deviation, skewness and kurtosis of MFCCs, evaluated over different (possibly, overlapped) time windows in the same signal frame. The proposed set of features allows to obtain a reduced-dimensional input while keeping robust performance in terms of classification ability;

- we numerically evaluate the proposed architecture with existing state-of-the-art approaches by using real-world recordings. Ten classes of multiple vehicles and tools, normally employed in a construction sites, have been considered and used to validate the performance of the proposed method, which proves the reliability, feasibility, and applicability of the proposed method in a real industry. Interestingly enough, we expect that the carried out numerical performance comparisons support the conclusion that the proposed architecture outperforms both traditional ML and other DL approaches;

- we propose a possible practical and real-time application of the proposed method aiming at providing a reliable and prompt response. In order to mitigate the rapid degradation possibly caused by variations of an environmental condition in which audio sources are recorded, we adopted an approach executing majority voting over time windows that collects a certain number of adjacent frames. The decision is in turn made by the majority of the produced labels over each window, which is expected to maintain and enhance the performance of the proposed method under diverse situations and environments.

We expect that the proposed approach will provide a significant impact on advancing real-world data analyses of construction projects with advanced and consistent performance and accuracy of audio-based work activity identification.

The rest of the paper is organized as follows. Section 2 shows the related literature. Section 3 describes the proposed approach in terms of both used architecture and set of features. Section 4 introduces the experimental setup, while Section 5 shows the obtained numerical results. Section 6 is dedicated to describe a possible practical and real-time application of the proposed method. Finally, Section 7 concludes the paper and outlines some future works.

## 2. Related work

### 2.1. Audio classification

In this subsection, we present a literature review on the audio classification approaches. Specifically, three lines of research can be outlined: ESC, security monitoring and construction site monitoring.

*Environmental sound classification.* In environmental sound classification, so far results have been obtained by using several classic acoustic models such as hidden Markov models (HMM), Gaussian mixture models (GMM), support vector machines (SVM), and so on (Barchiesi et al., 2015; Chachada & Jay Kuo, 2013). HMM is a parametric representation of time-varying features that simulate the human language process. It needs a large number of samples for time-consuming training (Su et al., 2011). GMM is a probability density estimation model that can fit all probability distribution functions, but it has a strong dependence on data and demonstrates sensitive to data noise (Barchiesi et al., 2015; Chachada & Jay Kuo, 2013). On the other hand, SVM maps the feature vectors from input space to a high-dimensional Hilbert space by using kernel tricks and seeks an optimal hyperplane in the high-dimensional space to classify data at the price of a high computational complexity (Dhanalakshmi et al., 2009). But it cannot solve the problems of large-scale training samples that lead to a large or prohibitively huge kernel matrix (Sharan & Moir, 2016). With the rise of more powerful computers, a variety of new artificial neural networks (ANNs) have also been introduced for acoustic modeling (Chachada & Jay Kuo, 2013).

In the literature, it is possible to find several instances of successful applications in the field of ESC that make use of DL techniques (Goodfellow et al., 2016; Abeßer, 2020). Most of these approaches rely on the use of a convolutional neural network (CNN), particularly efficient in capturing spatial information (Abdoli et al., 2019). For example, in the work of Piczak (2015a), the author exploits a 2-layered CNN working on the spectrogram of the data to perform ESC, reaching an average accuracy of 70% over different datasets (Piczak, 2015b). Other approaches, instead of using handcrafted features such as the spectrogram, perform end-to-end environmental sound classification obtaining higher results with respect to the previous ones (Tokozume & Harada, 2017). The MelNet architecture described in Li et al. (2018) has been proven to be remarkably effective in environmental sound classification. This architecture uses a combination of two deep CNNs (DCNNs) to classify environmental sound data (like rain, dogs, cats, engines, trains, airplanes, etc.). Some approaches, like that in Boddapati et al. (2017), also exploit the intrinsic characteristic of CNNs to perform classification by processing the audio spectrogram as an image.

Another successful approach is based on recurrent neural networks (RNNs) that are good at dealing with time series information (Goodfellow et al., 2016; Schmidhuber, 2015), Specifically, RNNs are used together with CNNs to take into account the temporal structures of audio signals. Works in Phan et al. (2017) and Sang et al. (2018) show the potential suitability of such approach by obtaining high accuracy on real-world dataset. Often, in order to avoid exploding or vanishing gradient issues, long short-term memory (LSTM) networks, which are a special type of RNN with gating units, have been applied to the input audio sequence (Bae et al., 2016).

*Security monitoring.* Regarding the automatic security monitoring, audio-based approaches generally fall under the umbrella of the Computational Auditory Scene Analysis (CASA) (Wang & Brown, 2006), whose aim is to successfully analyze a stream of continuous audio to identify and isolate the sources of interest contained in it. Usually, the audio can be acquired by using a single microphone, (large) acoustic arrays (Abu-El-Quran, 2006; Scardapane et al., 2015), distributed sensors (Sallai et al., 2011) or sets of smart sensors (Maijala et al., 2018). The sound identification and classification for monitoring purposes are usually performed by machine learning and/or deep learning technique (Fu et al., 2011; Barchiesi et al., 2015; Abeßer, 2020). There are many practical applications of audio monitoring. For example, there exists a vast literature regarding speech discrimination (Maganti et al., 2007; Zhang et al., 2004), vehicle recognition (Duarte & Hu, 2004; Hsieh et al., 2006) and weapon classification (Jin et al., 2009; Sallai et al., 2011). In addition, due to the maturity of the field there exist several commercial and open-source products that perform these tasks in specific domains, such as airports control (Aldeman et al., 2016).

*Construction site monitoring.* Traditional approaches to the manual collection of on-site work data and human-based construction project monitoring are time-consuming, inaccurate, costly, and labor intensive (Navon & Sacks, 2007). With the evolution of information technology, the construction industry has been seeking state-of-art field data collection and analysis methods to enhance construction project monitoring and robust field management (Cho et al., 2017). The growing demand for improving real-time field data collection and site monitoring has led to a paradigm shift in new intelligent construction management (Taghaddos et al., 2016). Various field data collection methods have been studied and implemented in construction project management such as GPS, ultra-wide band (UWB), and sensors (Cheng & Teizer, 2013). Several recent studies (Golparvar-Fard et al., 2015; Seo et al., 2015) also have used construction field images such as daily construction photography to explore automatic image-based progress monitoring.

Studies on sound identification of construction site activities, involving signal processing and audio classification, have primarily focused on four main areas: signal analysis, feature extraction, model training, and model testing (Sharan & Moir, 2016). Some studies (Cheng & Teizer, 2013; Cheng et al., 2017; Cho et al., 2017; Zhang et al., 2018) have applied various algorithms such as the support vector machine (SVM) and the Hidden Markov model (HMM) to test and evaluate the audio-based classification of the activity types related to construction operational equipment. In addition, authors in Xie et al. (2019) adopt the *k*-NN classifier for sound classification based on extracted features of selectively retrieved sound data, assisted by a construction schedule in the XML format extracted from a construction scheduling software. A comprehensive comparison of the performance of several classifiers applied to several construction sites audio signals can be found in Lee et al. (2020).

More recently, DL techniques demonstrate their effectiveness by obtaining a high accuracy in classification of construction site activities (Sherafat et al., 2020). The work in Rashid & Louis (2019) exploits a RNN for analyzing time-series obtained by several sensors located on construction machines. Among other approaches, the most common use DCNNs applied to audio spectrograms. This kind of approach is exploited in Maccagno et al. (2021) whose aim is to develop an application able to recognize vehicles and tools used in construction sites, and classify them in terms of type and brand. This task has been tackled with a neural network approach, involving the use of a DCNN, which will be fed with the mel spectrogram of the audio source as input. However, the classification task presented in this paper is limited to only five classes extracted from audio files collected in several construction sites, containing in situ recordings of multiple vehicles and tools. Finally, Scarpiniti et al. (2020) propose a deep RNN (DRNN) approach, based on LSTM units (Goodfellow et al., 2016), for the classification of real-world data recorded in construction sites. Both these last methods provide very high accuracy in classifying the recorded audio data.

### *2.2. DBN approaches for audio*

In this subsection we present a literature review on the deep belief network (DBN), with a particular focus on the audio classification.

DBNs were the first deep network models that successfully worked in practice (Goodfellow et al., 2016). The main aim of DBNs is to learn a layer-wise and unsupervised abstract representation of the input data in a hierarchical model. It has been demonstrated that DBNs are compact universal approximators (Le Roux & Bengio, 2010; Montufar et al., 2011) and that they behave very well on small and medium size datasets (Bondarenko & Borisov, 2013).

Since their first appearance in 2006 (Hinton et al., 2006), DBNs have received much interest from researchers and have been widely used to solve problems in signal processing, speech recognition and many other fields (Bondarenko & Borisov, 2013; Hamel & Eck, 2010; Mohamed et al., 2012; Zhang & Wu, 2013). However, to the best of our knowledge, DBNs have been amply exploited for speech signals and related application (Hinton et al., 2012) but they were rarely used for recognizing and classifying environmental sounds that are relatively unstructured. Just authors in Xue & Su (2015) use the DBN model to recognize auditory scenes of typical indoor and outdoor scenarios (inside vehicle, beach, train station, street, restaurant, raining, etc.) by discovering unsupervised features and generating high-level descriptions of scene audio. Some recent extension of DBNs are related to their convolutional version (Lee et al., 2009; Li et al., 2019), but they are mainly used for image processing.

In these state-of-the-art works, a feature vector is used to set the states of the visible units on the lowest (input) layer, then the DBN is first pre-trained one layer at a time by using an unsupervised learning procedure. Once the pre-training is completed, the resulting feed-forward neural network is fine-tuned in a discriminative way by using the back-propagation algorithm to slightly adjusts the weights in every layer to optimize them for the classification (Xue & Su, 2015; Mohamed et al., 2012).

Differently from the literature reviewed above, in this paper we focus our attention on the challenging task of real-time monitoring of construction sites by exploiting the powerful representation provided by audio signals, which can be recorded, transmitted, saved and analyzed in a simpler way with respect video recordings. In addition, since we need of a high performance, we expect that DL techniques can guarantee a sufficient accuracy to be used in real-time monitoring systems, by trying to keep limited the inference time. Due to their nice trade-off between accuracy, training time, and inference time among other DL techniques, the attention of the paper is devoted towards the DBN architectures.

## 3. The proposed approach

The proposed approach consists in using a DBN, which is a learning model based on deep neural networks having capability of unsupervised pre-learning. DBN is composed of multiple modules stacked each other. After an unsupervised layer-wise learning, a DBN can be further fine-tuned by using a supervised learning (i.e., using class labels) in order to perform the final classification. The DBN is fed with a set of feature based on the MFCCs. Specifically, six statistics are evaluated from MFCCs over different (possibly, overlapped) time windows in the same signal frame. The proposed set of features makes the classification more robust with respect to environmental changes.

In the following subsections, we provide a detailed description of the feature extraction procedure and of each block of the used architecture.

### *3.1. Feature extraction*

In the process of audio classification, feature extraction is the most important step before the choice of a suitable classification technique (Mierswa & Morik, 2005). The assessment of several different features, extracted from audio samples or signals, has been discussed in the literature (Jiang et al., 2002). These features can regard both the time and the spectral domains (Lu et al., 2002; Scardapane et al., 2013). For example, (Hiyane, 2001) presented a signal processing-based system to classify five types of single impulsive sounds, whose features are based on peak and reverberation times. Often, basic statistics such as the mean and variance of the signal have been used as the main features for classification. As an effective alternative, many researchers started using the features such as Mel-Frequency Cepstral Coefficients (MFCCs) because they are time-invariant timbral descriptors and of reliable classification results (Zheng et al., 2001; Chu et al., 2009; Mendes da Silva et al., 2020). Several works used MFCC features alongside calculating mean and variance for each sound class (Chu et al., 2009).

Although, the use of mean and variance can significantly decrease the size of input dimension, it could not be a robust choice in a practical context due to the environmental variability introduced in the real-world recordings. This issue is further worsen by considering challenging environments, such as construction sites. Motivated by

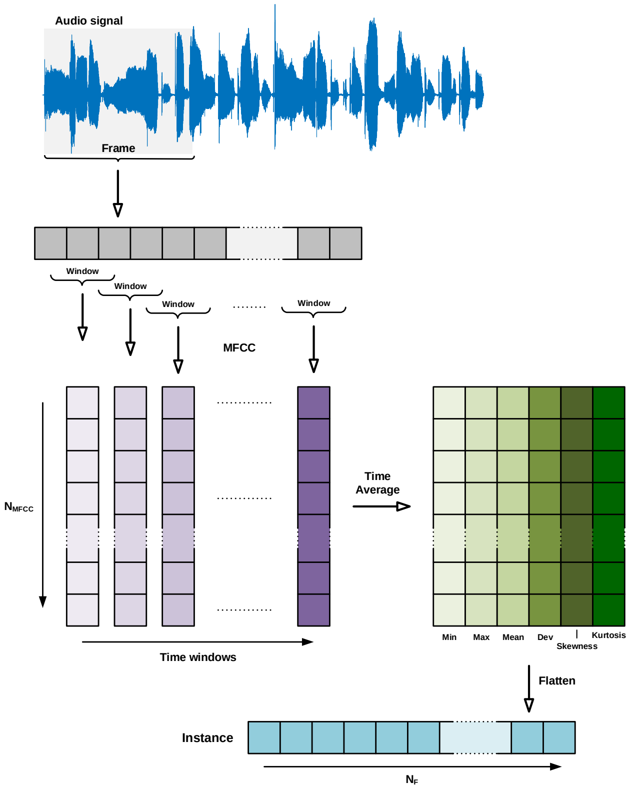


Figure 1: The whole process used for the feature extraction phase.

these considerations, in this work, we propose to take into consideration $N_s$ time statistics extracted from the MFCCs. Specifically, we consider the following six aggregate statistics: the minimum and maximum values of MFCCs, and the first, second, third and fourth moments (i.e., mean, standard deviation, skewness and kurtosis) of MFCCs. These statistics are evaluated over different (possibly, overlapped) time windows in the same signal frame.

In this paper, we consider a total of $N_{\text{mfcc}} = 64$ MFCC coefficients. The MFCCs are extracted by considering a window size of $L = 2048$ sample with an hop size of $H = 512$ samples. This implies that an audio frame of $T_F = 100$ ms, equivalent to 4410 samples when a sampling frequency $F_S = 44{,}100$ Hz is used, generates 9 vectors of 64 MFCCs. Hence, each frame of audio input generates a set of $64 \times 9$ parameters. Then the previous six statistics are evaluated along the time index, obtaining a total of $N_F = 6 \times 64 = 384$ features per frame. The list of these 384 features generates an instance of the dataset. The whole process of feature extraction is graphically illustrated in Fig. 1, while the used parameters in the process are listed in Table 1.

Table 1: Main parameters used in the feature extraction process.

| Description | Parameter | Value |
|---|---|---|
| Sampling frequency | $F_S$ | 44100 Hz |
| Frame duration | $T_F$ | 100 ms |
| Number of MFCCs | $N_{\text{mfcc}}$ | 64 |
| Number of statistics | $N_s$ | 6 |
| Number of features | $N_F$ | 384 |
| Analysis window size | $L$ | 2048 |
| Hope size | $H$ | 512 |

### *3.2. Deep Belief Network (DBN)*

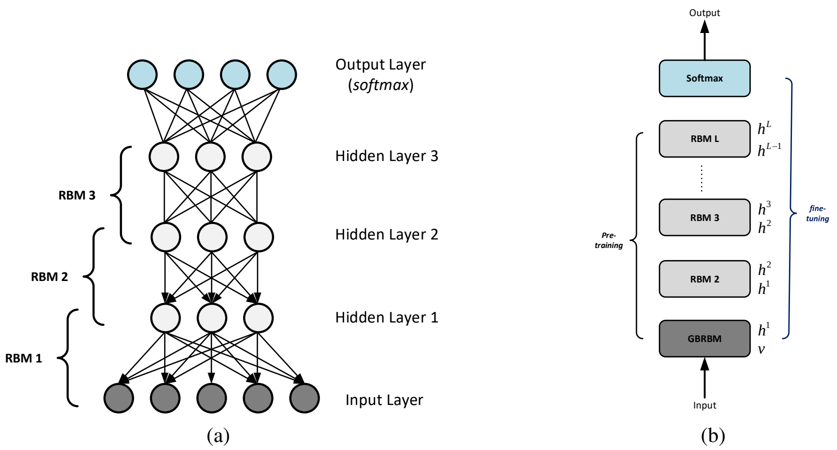
The extracted features will be used as input to a deep belief network (DBN). DBN is a particular deep architecture composed by stacking several computational layers formed by restricted Boltzmann machines (RBMs) one on the top of the other, as depicted in Fig. 2, and they incorporate both unsupervised pre-training and supervised fine-tuning. The unsupervised pre-training is used to obtain data distribution without the need of labels (Hinton et al., 2006). DBNs are also able to nicely scale on graphical processors for big data analytics (Raina et al., 2009; Chen & Lin, 2014). Although DBNs are not the most recent architecture, they show several advantages compared to other methods in that pre-training improves the model performance by avoiding overfitting and enhancing the model generalization such as in presence of background noise (Pinaya et al., 2016). This benefit is critical in construction sites given the limited number of samples available due to the difficulties in obtaining high quality recordings.

Each of single RBMs that compose the DBN is trained by an unsupervised layer-wise procedure by exploiting the *contrastive divergence* (CD) algorithm (Hinton et al., 2006; Fischer & Igel, 2014). The learned weights are then copied to a deep neural network (DNN). When used for the classification tasks, a top output layer is added to the DNN in order to make a prediction of the correct class label, i.e, the type of tool and equipment we are interested to classify on construction sites. Usually, this output layer uses a softmax function for the multi-class classification: the softmax is used to calculate a probability for every possible class (Mohamed et al., 2012). Note that the whole stack is a hybrid generative model whose top two layers are undirected (they form the final RBM in the stack) while the lower layers have top-down directed connections (Goodfellow et al., 2016), as shown in Fig. 2a

In the following subsections, we briefly describe the RBM, its unsupervised learning algorithms and the final fine-tuning through the back-propagation algorithm.

#### *3.2.1. Training of RBMs*

A RBM, also known as Bernoulli RBM, is a particular network composed by two layers of generally binary units (Goodfellow et al., 2016; Fischer & Igel, 2014), as shown in Fig. 3. Real-valued units are also possible, as shown next in this section. The first layer, called *visible layer*, represents the observed data and it is composed of $N_V$ units $v_i$, $i = 1, 2, \ldots, N_V$ directly fed by the data. In our case, the visible layers is



(a) (b)

Figure 2: (a) A schematic representation of a Deep Belief Network (DBN) with three hidden layers. (b) A Layer-wise block representation of a DBN for classification.

directly fed by the $N_F$ features extracted from the audio signals, i.e., $N_V = N_F$. The second layer, called *hidden layer*, is used, instead, to capture the dependencies between the observed data, and it is composed by $N_H$ units $h_j$, $j = 1, 2, \ldots, N_H$. Every unit in the visible layer is connected to all the units of the hidden layer and vice versa, but there are no connections between units in the same layer. The connections between the visible and hidden layers are of undirected type. Additionally, we denote with $b_i$ and $c_j$ the biases of the visible and hidden units, respectively.

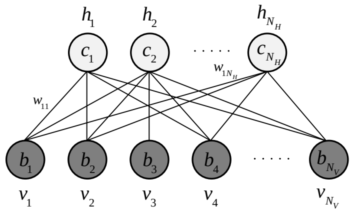


Figure 3: A Restricted Boltzmann Machine (RBM).

Let us denote with $\mathbf{v} = [v_1, v_2, \ldots, v_{N_V}]^T$ and $\mathbf{h} = [h_1, h_2, \ldots, h_{N_H}]^T$, the vectors collecting the visible and hidden units of a RBM, respectively. Hence, for a joint configuration, $(\mathbf{v}, \mathbf{h})$ of visible and hidden units it is possible to evaluate an energy function given by:

$$\mathcal{E}(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{N_V} b_i v_i - \sum_{j=1}^{N_H} c_j h_j - \sum_{i=1}^{N_V} \sum_{j=1}^{N_H} w_{ij} v_i h_j, \tag{1}$$

where $v_i$, $h_j$ are the binary states of visible unit $i$ and hidden unit $j$, while $b_i$, $c_j$ are their biases, and $w_{ij}$ is the weight of the link between them (see Fig. 3). By using a matrix notation, let us denote the coefficients of biases as the vectors $\mathbf{b} = \{b_i\}_{i=1}^{N_V}$, $\mathbf{c} = \left\{c_j\right\}_{j=1}^{N_H}$ and weights as the matrix $\mathbf{W} = \left\{w_{ij}\right\}$, for $i = 1, \ldots, N_V$ and $j = 1, \ldots, N_H$. In the following, we denote the set of all the RBM parameters in a compact way as $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$.

The RBM uses the energy function in (1) to assign a probability to every possible pair of a visible and a hidden vector. For example, the probability of a visible vector $\mathbf{v}$ is given by summing over all possible hidden vectors $\mathbf{h}$, as:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})}, \tag{2}$$

where $Z = \sum_{\mathbf{v},\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})}$ is a normalizing factor, called the partition function.

When both the visible and hidden units are binary, the sampling probabilities are very easy to sample. Specifically, we obtain:

$$p\left(h_j = 1 \mid \mathbf{v}, \boldsymbol{\theta}\right) = \sigma\left(b_i + \sum_{i=1}^{N_V} w_{ij} v_i\right) \tag{3}$$

and

$$p\left(v_i = 1 \mid \mathbf{h}, \boldsymbol{\theta}\right) = \sigma\left(c_i + \sum_{j=1}^{N_H} w_{ij} h_j\right), \tag{4}$$

where $\sigma(x) = 1/\left(1 + e^{-x}\right)$ denotes the sigmoid function.

RBMs are usually trained by the gradient descent algorithm on the log-likelihood function:

$$\mathcal{L}(\boldsymbol{\theta} \mid \mathbf{v}) = p(\mathbf{v} \mid \boldsymbol{\theta}) = \log \frac{1}{Z} \sum_{\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})} = \log \sum_{\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})} - \log \sum_{\mathbf{v},\mathbf{h}} e^{-\mathcal{E}(\mathbf{v},\mathbf{h})}. \tag{5}$$

The derivative of the log-likelihood (5) of a training set $\{\mathbf{v}_n\}_{n=1}^{N}$ with respect to the weights $w_{ij}$ is very simple (Fischer & Igel, 2014):

$$\frac{1}{N} \sum_{n=1}^{N} \frac{\partial \log p(\mathbf{v}_n)}{\partial w_{ij}} = \left\langle v_i h_j \right\rangle_{\text{data}} - \left\langle v_i h_j \right\rangle_{\text{model}} \tag{6}$$

where $\langle\cdot\rangle_{\text{data}}$ and $\langle\cdot\rangle_{\text{model}}$ are the expectations under the distributions defined by the data and the model, respectively.

An exact maximization of the log-likelihood function is infeasible in large RBMs because the term $\left\langle v_i h_j \right\rangle_{\text{model}}$ has an exponentially computational complexity for the derivative evaluation. However, there exist a very efficient approximate training procedure, known as *contrastive divergence* (CD) (Hinton, 2002). This procedure begins by setting the states of the visible units to a training vector. Subsequently, the states of the hidden units are computed all together by (3). Once binary states have been chosen for the hidden units, a "reconstruction" is produced by setting each new $\widetilde{v}_i$ to one with a probability given by (4) and, finally, the states of the hidden units are updated again with (3) obtaining $\widetilde{h}_j$. This forms a one step of the a Gibbs sampling procedure, which consists of an alternating updating of the hidden units by using (3) followed by updating the visible units by using (4) and so on more and more again (Fischer & Igel, 2014). Using the contrastive divergence, the update equation at the $t$-th iteration becomes:

$$w_{ij}(t+1) = w_t(t) + \eta \left( \left\langle v_i \, h_j \right\rangle_{\text{data}} - \left\langle \widetilde{v_i} \, \widetilde{h_j} \right\rangle_{\text{reconstruction}} \right). \tag{7}$$

In a similar way, the biases $b_i$ and $c_j$ can be updated as:

$$b_i(t+1) = b_i(t) + \eta \left( v_i - \widetilde{v_i} \right), \tag{8}$$

and

$$c_j(t+1) = c_j(t) + \eta \left( h_j - \widetilde{h_j} \right). \tag{9}$$

The learning algorithms in (7), (8) and (9) are iterated until the convergence, usually for a number $N_e^R$ of epochs.

When the input data is real-valued, such as the feature extracted by an audio signal, it is more natural to use the Gaussian-Bernoulli RBM, which considers input variables as linear with a Gaussian noise. In this case, the energy function in (1) is modified as follows:

$$\mathcal{E}(\mathbf{v}, \mathbf{h}) = \sum_{i=1}^{N_V} \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_{j=1}^{N_H} c_j h_j - \sum_{i=1}^{N_V} \sum_{j=1}^{N_H} \frac{v_i}{\sigma_i} h_j w_{ij}, \tag{10}$$

where $\sigma_i$ is the standard deviation of the Gaussian noise for the visible unit $i$.

The two conditional distributions in (3) and (4), for the GBRBM become:

$$p\left(h_j = 1 \mid \mathbf{v}, \boldsymbol{\theta}\right) = \sigma\left( b_i + \sum_{i=1}^{N_V} \frac{v_i}{\sigma_i} w_{ij} \right) \tag{11}$$

and

$$p\left(v_i = 1 \mid \mathbf{h}, \boldsymbol{\theta}\right) = \mathcal{N}\left( c_i + \sigma_i \sum_{j=1}^{N_H} h_j w_{ij}, \, \sigma_i^2 \right), \tag{12}$$

respectively, where $\mathcal{N}\left(\mu, \sigma_i^2\right)$ is a Gaussian function with mean $\mu$ and variance $\sigma_i^2$. For the training with the CD algorithm, the data are usually normalized to zero mean and unit variance, and hence the standard deviations are set to one. This simplifies the learning and the unique difference with respect the binary RBM is the use of (12) with $\sigma_i = 1$ in the first layer, instead of (4), while the rest remains unchanged.

### *3.2.2. Stacking RBMs*

A DBN is obtained by stacking $L$ RBMs on the top of each other, as shown in Fig. 2b. The hidden state of the $l$-the layer, for $l = 1, 2, \ldots, L$, is denoted as $\mathbf{h}^l = \left[ h_1^l, h_2^l, \ldots, h_{N_H^l}^l \right]^T$, where $N_H^l$ is the number of hidden units in the $l$-th layer.

The visible layer $\mathbf{v}$ of the first RBM is clamped directly with the features extracted from the input signal, and a hidden state $\mathbf{h}^1$ is produced and a set of weights $\boldsymbol{\theta}^1 = \left\{ \mathbf{W}^1, \mathbf{b}^1, \mathbf{c}^1 \right\}$ is learned. The hidden state $\mathbf{h}^1$ of the first layer is then used as input to the second RBM, in order to produce a hidden state $\mathbf{h}^2$ and a set of parameters $\boldsymbol{\theta}^2 = \left\{\mathbf{W}^2, \mathbf{b}^2, \mathbf{c}^2\right\}$. Again, the hidden state $\mathbf{h}^2$ of the second layer is then used as input to the third RBM and an estimate of the hidden state $\mathbf{h}^3$ and the set of parameters $\boldsymbol{\theta}^3 = \left\{\mathbf{W}^3, \mathbf{b}^3, \mathbf{c}^3\right\}$ is obtained. The process is continued until reaching the top layer $L$, generating the last hidden state $\mathbf{h}^L$.

Upon the $l$-th layer of RBM is learned, its parameters $\boldsymbol{\theta}^l$ are frozen, and we pass to train the $(l + 1)$-th layer. When we have learned all the $L$ layers, we obtain the directed generative model called a deep belief network (DBN) that has $L$ different parameter sets ($L$ weight matrices and $2L$ bias vectors) between the first lower layer and top $L$-th higher one. Since the feature are usually real-valued, the first layer is composed by a Gaussian-Bernoulli RBM, while all the other layers implement the simple Bernoulli (binary) RBM.

### *3.2.3. Supervised fine-tuning*

In order to use the DBN as a classifier, we then simply add a (top) final "softmax" layer of label units representing the possible $K$ class values $y_k$, $k = 1, 2, \ldots, K$, i.e., the number of tools and equipments we are interested to classify on construction sites. In summary, the generative DBN has been used for initializing all the detecting layers of a deterministic feed-forward DNN with parameters equal to $\boldsymbol{\theta}^l$, for $l = 1, 2, \ldots, L$. Hence, we have to train the whole DNN in a discriminative way, by using a back-propagation fine-tuning. After the fine-tuning, the outputs of each layer is not more binary sampled values, but they become real-valued.

For the final softmax layer, the probability of the $q$-th label $y_q$, given the real-valued activations of the final layer of features $\mathbf{h}^l$, is defined as:

$$p\left(y_q \mid \mathbf{h}^L\right) = \frac{\exp\left(c_q + \sum_i h_i^L w_{iq}\right)}{\sum_k \exp\left(c_k + \sum_i h_i^L w_{ik}\right)}, \tag{13}$$

where $c_q$ is the bias of the $q$-th label and $w_{iq}$ is the weight from hidden unit $i$ in layer $L$ to label $q$.

The discriminative training must learn the set of weights $w_{iq}$ from the last layer of features to the label units. However, this training has not to destroy the feature discovered by the previous unsupervised pre-training: it simply fine-tunes existing hidden layers. This fine-tuning is performed by the classical back-propagation algorithm applied to the corss-entropy loss function:

$$\mathcal{L} = -\sum_{k=1}^{K} d_k \log(y_k), \tag{14}$$

where $d_k$ is the $k$-th target label and $y_k$ the (predicted) $k$-th output label. Since the back-propagation is a gradient-based minimization techniques, a learning rate $\mu$ is used. The training is performed over all the training samples for a certain number of epochs $N_e$, or anyway until the network convergence.

The complete algorithm to train the proposed architecture is summarized in Algorithm 1. The output of the training algorithm, is the full set of the network parameters $\boldsymbol{\theta}^l$ and the set of weights $w_{iq}$ of the output layer.

**Algorithm 1** — DBN training algorithm

**Input:**

- the input data $\mathbf{x}$, i.e., the features extracted from audio signals;
- the number of layers $L$, input units $N_V$ and hidden units for layer $N_H^l$;
- the number of classes $K$;
- the learning rates $\eta$ and $\mu$;
- the number of RBM epochs $N_e^R$ and DBN epochs $N_e$.

**Output:**

- the trained set of parameters $\boldsymbol{\theta}^l$, for $l = 1, 2, \ldots, L$;
- the $N_H^L \times K$ output weights $w_{iq}$.

```
Initialize weights $\mathbf{W}^1$ and biases $\mathbf{b}^1$ and $\mathbf{c}^1$        ▷ (visible layer)
Initialize the iteration index $t = 0$
Set the visible state to the input $\mathbf{v} = \mathbf{x}$
while $t < N_e^R$ do
    Evaluate the hidden state $\mathbf{h}^1$ by Eq. (3)
    Perform Gibss sampling obtaining $\widetilde{\mathbf{v}}^1$ and $\widetilde{\mathbf{h}}^1$ with Eqs. (4) and (3)
    Update the weight matrix $\mathbf{W}^1$ with Eq. (7)
    Update the bias vectors $\mathbf{b}^1$ and $\mathbf{c}^1$ with Eqs. (8) and (9)
    $t = t + 1$
end while
for $l = 2 : L$ do        ▷ (hidden layers)
    Initialize the iteration index $t = 0$
    Initialize weights $\mathbf{W}^l$ and biases $\mathbf{b}^l$ and $\mathbf{c}^l$
    Set the visible state to the output of the previous layer $\mathbf{v} = \mathbf{h}^{l-1}$
    while $t < N_e^R$ do
        Evaluate the hidden state $\mathbf{h}^l$ by Eq. (3)
        Perform Gibss sampling obtaining $\widetilde{\mathbf{v}}^l$ and $\widetilde{\mathbf{h}}^l$ with Eqs. (4) and (3)
        Update the weight matrix $\mathbf{W}^l$ with Eq. (7)
        Update the bias vectors $\mathbf{b}^l$ and $\mathbf{c}^l$ with Eqs. (8) and (9)
        $t = t + 1$
    end while
end for
Stack the $L$ trained RBMs on the top of each other
Add a softmax layer        ▷ (output layer)
Initialize weights and biases of the output layer
for $ep = 1 : N_e$ do
    Perform fine-tuning by back-propagation of the whole architecture
end for
Evaluate the final output of the architecture
return: $\boldsymbol{\theta}^l$ and the set $w_{iq}$
```

## 4. Experimental setup

In this section, we describe the experimental setup. Specifically, we introduce the used dataset and its pre-processing.

### *4.1. Dataset*

Audio data of equipment operations has been collected in several construction sites consisting of diverse construction machines and equipments. The activities of these machines were observed during certain periods, and the audio signals generated were recorded accordingly. A Zoom H1 digital handy recorder has been used for data collection purposes. All files have been recorded by using a sample rate of $F_S = 44{,}100$ Hz and different files in WAV format for each machine are available. The different files are referring to a single machine per class recorded during these are in use and working in similar environmental conditions.

Unlike artificially built datasets, when working with real data different problems arise, such as noise due to weather conditions and/or workers talking among themselves. Classes which did not have enough usable audio (too short, excessive noise, low quality of the audio) were ignored for this work.

Thus, we focused our work on the classification of a reduced number of classes; specifically we consider 10 classes related to 5 different machinery (excavators of different size, bulldozers, compactors, concrete mixer and shovel) from 6 different manufactures. Details on such classes are shown in Table 2. For all of the ten classes, after pre-processing, approximately 97 minutes of clean audio are available and have been used to train, validate, and test the proposed architecture.

Table 2: Description of the used dataset and related 10 classes.

| N. | Class | Description |
|---|---|---|
| 1 | CAT320D | Hydraulic excavator Caterpillar 320D |
| 2 | CAT320E | Hydraulic excavator Caterpillar 320E |
| 3 | CATC5K | Dozer Caterpillar D5K |
| 4 | Hitachi50U | Compact excavator Hitachi ZX50U |
| 5 | IRCOM | Ingersoll Rand Compactor |
| 6 | JC3CX | Dozer JCB 3CX |
| 7 | JD50D | Compact excavator John Deere 50D |
| 8 | JD50G | Compact excavator John Deere 50G |
| 9 | KomatsuPC200 | Hydraulic excavator Komatsu PC200 |
| 10 | ConcreteMixer | Concrete mixer Mercedes-Benz Actros |

### *4.2. Preprocessing*

All files have been preemptively pre-processed and the silence segments (frames where the root mean square (RMS) is under the threshold of $-30$ dB) have been removed. In order to feed the network with enough and proper data, each audio file for each class is segmented into fixed length frames. In this work, we consider frames of $T_F = 100$ ms, corresponding to size of 4410 samples. A total of 58,204 frames have been extracted.

In addition, also a standardization procedure has been used. A standard scaler operation has been applied to the data matrix, in order to transform data to zero mean and unit variance.

In order to train, validate, and test our architecture, we split the original audio files into three parts: training samples (52% of the original dataset), test samples (25% of the original dataset), and validation samples (23% of the original dataset) used to find the optimal setting of the hyper-parameters. Details about the number of instances for every class in each set are provided in Table 3. This split is equivalent to about 50 minutes of audio data for training the architecture, 25 minutes for testing it, and 22 minutes for its validation. For the training process, the audio data is about 5 minutes for class, for the testing the length varies between 1.6 and 4 minutes, while for the validation the length is just over 2 minutes. After the partition in frames, we obtain 30,152 training, 13,200 validating, and 14,852 testing audio segments, all with 384 features per frame, as described in Section 3.1. We remark that data in each split (training/validation/test) belongs to different files, which are referring to the same machine and have been recorded on the same construction site. However, they have been recorded at different time (not always in the same day) and involve different operation tasks.

Table 3: Number of instances of the used dataset.

| N. | Class | Training | Validation | Test |
|---|---|---|---|---|
| 1 | CAT320D | 3,030 | 1,307 | 1,333 |
| 2 | CAT320E | 3,068 | 1,273 | 1,022 |
| 3 | CATC5K | 2,995 | 1,338 | 997 |
| 4 | Hitachi50U | 3,013 | 1,322 | 966 |
| 5 | IRCOM | 2,979 | 1,352 | 991 |
| 6 | JC3CX | 3,048 | 1,291 | 2,554 |
| 7 | JD50D | 3,028 | 1,309 | 1,033 |
| 8 | JD50G | 2,978 | 1,353 | 1,827 |
| 9 | KomatsuPC200 | 2,979 | 1,352 | 2,180 |
| 10 | ConcreteMixer | 3,034 | 1,303 | 1,949 |
|  | **Total** | 30,152 | 13,200 | 14,852 |

Using the Python library `librosa`[1] (McFee et al., 2015), we extract the waveform of the audio tracks from the audio samples and, using the same library, we generate the log-scaled Mel spectrogram of the signal and evaluate the features. Simulations have been carried out by using a computer equipped with an Intel® i7-8700K CPU @ 3.70GHz, with an INVIDIA GeForce GTX 1080.

A bi-dimensional projection of a sub-set of the training instances, by using the t-distributed stochastic neighbor embedding (t-SNE), is shown in Fig. 4. Specifically, t-SNE is a dimensionality reduction technique, which is particularly suited for the visualization of high-dimensional datasets (van der Maaten & Hinton, 2008). This method tries to keep similar instances close and dissimilar instances apart. Fig. 4 clearly shows that the considered classes are quite separated and bodes well that they will be nicely separable in the higher-dimensional feature space, confirming the validity of the proposed set of features.

[1] Available at: `https://librosa.github.io/librosa/`

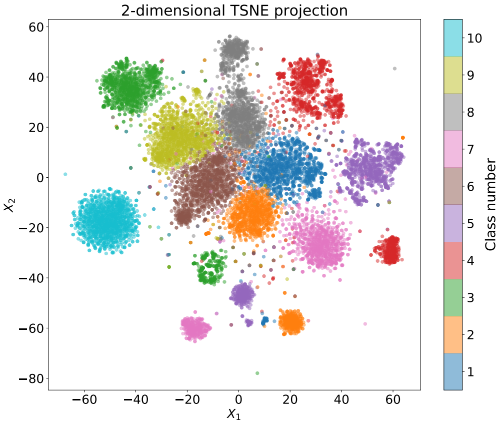


Figure 4: Bi-dimensional projection of the feature matrix using the t-SNE technique.

## 5. Numerical results

In this section we present some numerical results in order to demonstrate the effectiveness of the proposed approach and the suitability of the DBN for the classification of construction site tools and machinery. The results will be evaluated in terms of the overall accuracy (i.e., the proportion of instances that are correctly classified) and other well known metrics used in machine learning applications, specifically: precision, recall and F$_1$-score (Powers, 2011; Alpaydin, 2014). The confusion matrix will be also considered.

The training of the RBMs and the subsequent back-propagation fine-tuning have been performed by a mini-batch updated. A mini-batch is the amount of sample instance presented to the network at each learning step. The dimension $B$ of a mini-batch, also called batch size, should be accurately chosen since it represents a trade-off: small values provide fast convergence at the cost of a noisy gradient evaluation, while large values show slow convergence but accurate estimates of the error gradient. In this paper, we have selected the value $B = 64$ that provides good and accurate results. Inside a mini-batch, the instance are presented in a random order, to avoid bias due to the position order. Moreover, the final fine-tuning optimization is always performed by a dropout technique, with probability $p_{do} = 0.5$.

The main parameters used in simulation, obtained by considering the validation set, are reported in Table 4. Some of these parameters have been set heuristically by repeated trials. The rest have been selected by performing a grid search approach over suitable sets of values. To this purpose, Fig. 5 shows the results of the overall accuracy with respect the DBN learning rate $\mu$ and the number of back-propagation epochs $N_e$. The other parameters assume the values listed in Table 4. The figure clearly shows that the best results are obtained for $\mu = 0.02$ and $N_e = 200$, respectively. Hence, in the rest of this section we assume the latter values.

A first set of numerical results are related to the performance evaluation at varying the number $L$ of hidden layers in the DBN. Since each layer represents feature auto-

Table 4: Main parameters of the adopted approach and related meaning.

| **Description** | **Parameter** | **Value** |
|---|---|---|
| Mini-batch size | $B$ | 64 |
| Number of MFCC coefficients | $N_{\text{mfcc}}$ | 64 |
| Number of features | $N_F$ | 384 |
| Number of hidden layers | $L$ | 3 |
| Number of hidden units per layer | $N_H^l$ | 256/256/128 |
| RBM learning rate | $\eta$ | 0.0005 |
| RBM epochs | $N_e^R$ | 10 |
| DBN learning rate | $\mu$ | 0.02 |
| DBN epochs | $N_e$ | 200 |
| Dropout probability | $p_{do}$ | 0.5 |

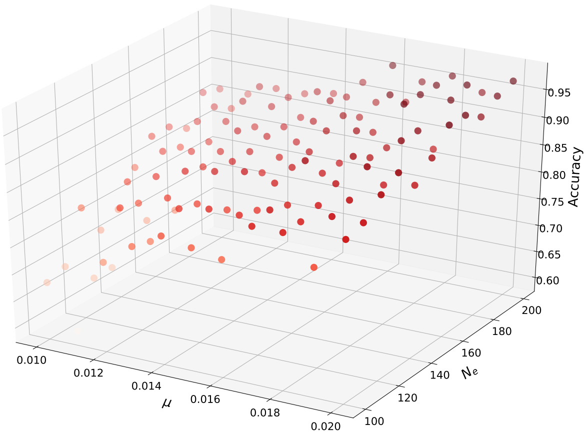


Figure 5: Overall accuracy vs. learning rate vs. number of epochs on the validation set.

matically extracted from the previous layer, a correct number of these hidden layers can largely help for a high accuracy in the classification process. Overall performance metrics and training time (in seconds) of the proposed approach, for different number $L$ of hidden layers, are shown in Table 5. This table clearly indicates that a suitable number of hidden layers is 3. In fact, if the number $L$ of layers is small the considered metrics are lower than the best ones, otherwise, if it is too large, the performance tends to rapidly and drastically drop off.

A second set of numerical results are related to the performance evaluation at varying the number $N_{\text{mfcc}}$ of MFCCs in the feature extraction process. Since this parameter controls the dimension $N_F$ of the feature space, the classification performance is strongly dependent of $N_{\text{mfcc}}$. Overall performance metrics and training time (in seconds) of the proposed approach for different number $N_{\text{mfcc}}$ of MFCCs are shown in Table 6. Performance, in terms of all the considered metrics, slightly decreases at diminishing and increasing of the number of MFCCs with respect to an optimal number

Table 5: Overall performance metrics and training time (in seconds) of the proposed approach for different number of hidden layers obtained on the validation set.

| L | Units per Layer | Accuracy | Precision | Recall | $F_1$ | Training time |
|---|---|---|---|---|---|---|
| 2 | [256, 128] | 0.9626 | 0.9624 | 0.9628 | 0.9624 | 1368 |
| 3 | [256, 256, 128] | 0.9820 | 0.9823 | 0.9820 | 0.9821 | 2837 |
| 4 | [256, 256, 256, 128] | 0.9160 | 0.9162 | 0.9140 | 0.9151 | 4795 |
| 5 | [256, 256, 256, 256, 128] | 0.7602 | 0.8562 | 0.7602 | 0.7936 | 7197 |

of coefficients. However, a larger number of MFCCs not only tends to excessively increase the computational load, but is also not beneficial since MFCCs represent increasing levels of spectral details in higher bands where the considered sounds have a limited contribution. This table clearly indicates that a suitable number of MFCCs is 64, providing $N_F = 384$.

Table 6: Overall performance metrics and training time (in seconds) of the proposed approach for different number of MFCCs obtained on the validation set.

| $N_{\text{mfcc}}$ | $N_F$ | Accuracy | Precision | Recall | $F_1$ | Training time |
|---|---|---|---|---|---|---|
| 96 | 576 | 0.9626 | 0.9629 | 0.9626 | 0.9626 | 3230 |
| 80 | 480 | 0.9740 | 0.9743 | 0.9740 | 0.9741 | 3072 |
| 64 | 384 | 0.9820 | 0.9823 | 0.9820 | 0.9821 | 2837 |
| 48 | 288 | 0.9712 | 0.9712 | 0.9712 | 0.9712 | 2556 |
| 32 | 192 | 0.9533 | 0.9530 | 0.9534 | 0.9532 | 2360 |

For comparison purpose, the same numerical test has been performed by using the whole set of MFCCs features, without the evaluation of the aggregate statistics. In this case, we experiment a diminishing of the accuracy and other performance. Specifically, the best accuracy for $N_{\text{mfcc}} = 64$ was 0.82. Also the training time is higher, of the order of 4200 seconds.

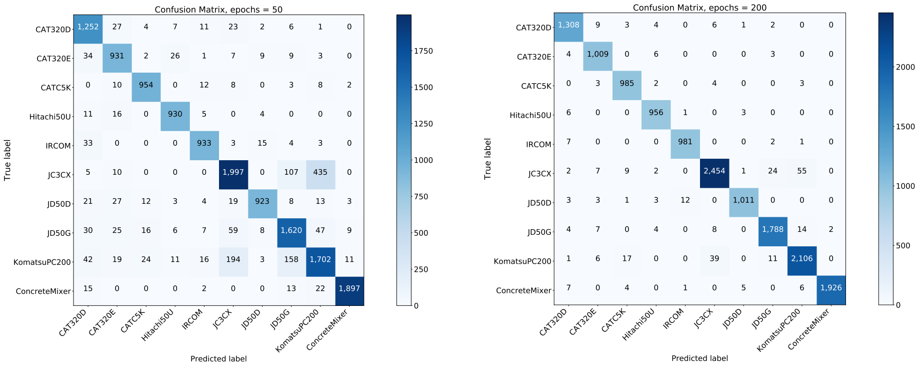
This last numerically setup (i.e., $L = 3$ layers, $\mu = 0.02$ and $N_e = 200$ epochs) with a number of MFCCs equal to $N_{\text{mfcc}} = 64$ has been chosen as the optimal set of hyper-parameters. Hence, the proposed architecture with this set of hyper-parameters has been evaluated on the test set: a detailed per-class metric evaluation is shown in Table 7, which corresponds to an overall accuracy of 97.79%. From Table 7, it is evident that the majority of classes are well classified. However, some classes show a lower accuracy. Specifically, also in accordance with the confusion matrix in Fig. 6b, class 6 (JC3CX) and class 9 (KomatsuPC200) provide the worst performance, in any case greater than 96% in terms of precision and recall.

In the following, in order to better understand the behavior of the DBN for the classification, especially for the couple of classes that provides a lower performance, we analyze the obtained confusion matrices. In particular, the confusion matrices after 50 and 200 epochs are shown in Figures 6a and 6b, respectively. From a careful examination of such matrices, we can see that the worst performing classes (JC3CX and KomatsuPC200) show the greater number of false positive between them, hence many

Table 7: Per-class performance metrics and their weighted average obtained on the test set.

| **Class** | **Precision** | **Recall** | **F$_1$-score** | **Support** |
|---|---|---|---|---|
| 1 | 0.9812 | 0.9747 | 0.9779 | 1333 |
| 2 | 0.9873 | 0.9665 | 0.9768 | 1022 |
| 3 | 0.9880 | 0.9666 | 0.9772 | 997 |
| 4 | 0.9896 | 0.9785 | 0.9840 | 966 |
| 5 | 0.9899 | 0.9859 | 0.9879 | 991 |
| 6 | 0.9608 | 0.9773 | 0.9690 | 2554 |
| 7 | 0.9787 | 0.9902 | 0.9844 | 1033 |
| 8 | 0.9787 | 0.9770 | 0.9778 | 1827 |
| 9 | 0.9661 | 0.9638 | 0.9649 | 2180 |
| 10 | 0.9882 | 0.9990 | 0.9936 | 1949 |
| **weighted avg** | 0.9779 | 0.9780 | 0.9779 | 14852 |

instances of the bulldozer JC3CX have been classified as the excavator KomatsuPC200 and vice-versa. Sometimes the bulldozer JC3CX has also been classified as the excavator JD50G. Similar behavior for the KomatsuPC200, at times confused also for the bulldozer CATC5K. The rest of the classes, instead, show very good results.



| True label \ Predicted label | CAT320D | CAT320E | CATC5K | Hitachi50U | IRCOM | JC3CX | JD50D | JD50G | KomatsuPC200 | ConcreteMixer |
|---|---|---|---|---|---|---|---|---|---|---|
| CAT320D | 1,252 | 27 | 4 | 7 | 11 | 23 | 2 | 6 | 1 | 0 |
| CAT320E | 34 | 931 | 2 | 26 | 1 | 7 | 9 | 9 | 3 | 0 |
| CATC5K | 0 | 10 | 954 | 0 | 12 | 8 | 0 | 3 | 8 | 2 |
| Hitachi50U | 11 | 16 | 0 | 930 | 5 | 0 | 4 | 0 | 0 | 0 |
| IRCOM | 33 | 0 | 0 | 0 | 933 | 3 | 15 | 4 | 3 | 0 |
| JC3CX | 5 | 10 | 0 | 0 | 0 | 1,997 | 0 | 107 | 435 | 0 |
| JD50D | 21 | 27 | 12 | 3 | 4 | 19 | 923 | 8 | 13 | 3 |
| JD50G | 30 | 25 | 16 | 6 | 7 | 59 | 8 | 1,620 | 47 | 9 |
| KomatsuPC200 | 42 | 19 | 24 | 11 | 16 | 194 | 3 | 158 | 1,702 | 11 |
| ConcreteMixer | 15 | 0 | 0 | 0 | 2 | 0 | 0 | 13 | 22 | 1,897 |

(a)

| True label \ Predicted label | CAT320D | CAT320E | CATC5K | Hitachi50U | IRCOM | JC3CX | JD50D | JD50G | KomatsuPC200 | ConcreteMixer |
|---|---|---|---|---|---|---|---|---|---|---|
| CAT320D | 1,308 | 9 | 3 | 4 | 0 | 6 | 1 | 2 | 0 | 0 |
| CAT320E | 4 | 1,009 | 0 | 6 | 0 | 0 | 0 | 3 | 0 | 0 |
| CATC5K | 0 | 3 | 985 | 2 | 0 | 4 | 0 | 0 | 3 | 0 |
| Hitachi50U | 6 | 0 | 0 | 956 | 1 | 0 | 3 | 0 | 0 | 0 |
| IRCOM | 7 | 0 | 0 | 0 | 981 | 0 | 0 | 2 | 1 | 0 |
| JC3CX | 2 | 7 | 9 | 2 | 0 | 2,454 | 1 | 24 | 55 | 0 |
| JD50D | 3 | 3 | 1 | 3 | 12 | 0 | 1,011 | 0 | 0 | 0 |
| JD50G | 4 | 7 | 0 | 4 | 0 | 8 | 0 | 1,788 | 14 | 2 |
| KomatsuPC200 | 1 | 6 | 17 | 0 | 0 | 39 | 0 | 11 | 2,106 | 0 |
| ConcreteMixer | 7 | 0 | 4 | 0 | 1 | 0 | 5 | 0 | 6 | 1,926 |

(b)

Figure 6: Confusion matrix after: (a) 50 epochs and (b) 200 epochs.

The convergence behavior can be analyzed by seeing the shape of the training loss functions. Specifically, Fig. 7 shows that the loss of the proposed network decreases quickly for the first 100 epochs and then continues to decrease gradually until reaching the convergence at 200 epochs. A quantitatively evaluation of the effect of the values of the loss function, can be observed in Fig. 6 that compares the confusion matrices of the DBN after 50 and 200 epochs, respectively. A comparison of this couple of matrices highlights the poorer performance of the architecture after 50 epochs (corresponding to an accuracy of 88.47%) with respect to the suggested number of 200 epochs (corresponding to a top accuracy of 97.79%). No more substantial improvements are obtained by further increasing the number of training epochs.

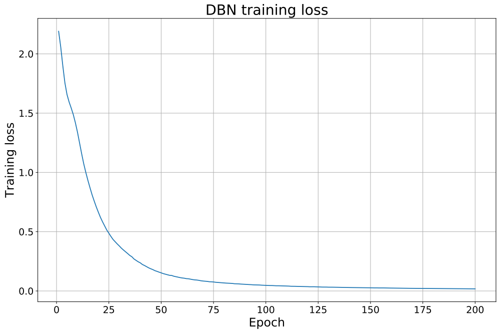


Figure 7: Training loss of the DBN for 200 epochs of learning.

Finally, we perform comparisons with other state-of-the-art approaches. Comparisons have been performed with both other Deep Learning methods and some standard Machine Learning approaches. Specifically, the proposed DBN approach has been compared with the Convolution Neural Network (CNN) in Maccagno et al. (2021) based on the spectrogram interpreted as image, and the Deep Recurrent Neural Network (DRNN) in Scarpiniti et al. (2020). Both these method showed high performance on the same dataset. Regarding the traditional ML approaches, results have been compared against the Multi-Layer Perceptron (MLP), the Support Vector Machine (SVM), $k$ Nearest Neighborhoods (kNN), Decision Trees (DT), Naive Bayes (NB), Quadratic Discriminant Analysis (QDA), and the two ensemble methods of Random Forests (RF) and AdaBoost (AB), with 500 and 100 base learners, respectively (Alpaydin, 2014). Specifically, the used hyper-parameters are summarized in the following. MLP consists in two hidden layers of 100 and 50 neurons with the ReLU activation function, while the learning is performed by 1000 iterations with the step size set to 0.001 and an $L_2$ penalty term. The SVM uses the default RBF kernel with a regularization parameter $C = 1$ and a kernel size $\gamma = 1/N_F$. The kNN uses a number $k = 3$ of neighborhoods. The DT uses the Gini impurity as information gain measure, with a minimum number of two samples to perform a split and no pruning. The NB classifier assumes that the likelihood of the features is Gaussian. QDA is a classifier with a quadratic decision boundary and no regularization has been used.

Table 8 summarizes the numerical results of the considered algorithms, in terms of the previous evaluation metrics. In addition, Table 8 also reports the training and inference time of the compared models. These times do not consider the time for feature extraction, but only the time needed to train the model and to predict the output of the whole test set, respectively. The training and inference time of the traditional ML approaches reported in Table 8 are referring to the implementation provided by the Scikit-learn library.

Table 8 shows that the proposed DBN provides the best accuracy of 97.79% and a similar behavior for precision, recall and F$_1$-score. Moreover, as can be seen from the table, from an accuracy point of view the other deep learning methods behave in

Table 8: Overall performance metrics and training and inference times (in seconds) of the compared DL and ML approaches.

| Approach | Accuracy | Precision | Recall | $F_1$ | Training time | Inference time |
|---|---|---|---|---|---|---|
| DBN | 0.9779 | 0.9780 | 0.9779 | 0.9779 | 2837 | 0.430 |
| CNN | 0.9708 | 0.9730 | 0.9734 | 0.9732 | 4300 | 0.786 |
| RNN | 0.9671 | 0.9569 | 0.9671 | 0.9620 | 6240 | 1.022 |
| MLP | 0.9599 | 0.9599 | 0.9599 | 0.9599 | 118 | 0.115 |
| SVM | 0.9602 | 0.9601 | 0.9602 | 0.9601 | 125 | 66.73 |
| kNN | 0.9431 | 0.9436 | 0.9431 | 0.9429 | 5 | 370.2 |
| DT | 0.8692 | 0.8692 | 0.8692 | 0.8692 | 34 | 0.022 |
| NB | 0.6770 | 0.7865 | 0.6770 | 0.7034 | 4 | 0.722 |
| QDA | 0.7302 | 0.7737 | 0.7302 | 0.7355 | 11 | 1.635 |
| RF | 0.9614 | 0.9613 | 0.9610 | 0.9611 | 288 | 3.158 |
| AB | 0.8355 | 0.8362 | 0.8355 | 0.8329 | 292 | 2.671 |

a similar way, obtaining up to 97% of accuracy. The traditional machine learning methods, instead, show results in a wider range. While some approaches (MLP, SVM, RF and kNN) are comparable to the deep learning ones performing with an accuracy in the range 94%–96%, the remaining others (DT, NB, QDA and AB) provided poorer results, with lower accuracy in the range 68%–87%. The worst performing approach is the NB, which makes too simple assumption and provides only the 68% of accuracy, even if it is very fast, while the best among the ML approaches is RF. Decision tree (DT), instead, provides an intermediate accuracy of 87%, however greater than that of AB, while running in a small amount of time.

Regarding the training time, Table 8 shows that DBN obviously needs a longer training with respect to the ML approaches, but its training time is considerably smaller than the other considered DL approaches (CNN and RNN). However, this higher training time corresponds to a greater accuracy. Interestingly enough, a careful examination of the last column in Table 8 shows that, despite its training time, the inference time of the proposed DBN is considerably limited compared to the majority of the considered approaches. Just DT and MLP present a lower inference time, but only MLP can compete in accuracy. It is difficult to compare different classifiers that presents different behavior for heterogeneous metrics (i.e., accuracy, training time and inference time) and decide which could be the best model. An attempt to represent these different metrics all together is provided in the scatterplot of Fig. 8. This figure shows each classifier with respect to its accuracy ($x$-axis), its normalized training time ($y$-axis) and its normalized inference time ($z$-axis). The normalization is performed with respect to the worst case. In order to have a decent classifier, it should be located in the in front bottom-right corner of the figure. This indicates that the classifier provides a high accuracy and low training and inference time. The proposed DBN, denoted with a red diamond, is located in the correct side of the figure and behaves significantly well comparing to the other approaches. In fact, despite its training time (not the higher one), it produces a high accuracy (the best one) with a sufficiently small inference time (one of the smallest). Since the training is usually performed offline, a fast inference (i.e., a prediction of a new class label) is of primary importance in real-time scenarios and represents a valuable characteristic of a classification model. From this trade-off point of view, we can consider the proposed DBN as the best one. We have to remark here, that we are searching for a fast, accurate, and reliable approach, which can be suitable for real-time critical applications, such as classification of work activities in a construction site. DBN possesses all these characteristics that make it a suitable approach for the enhanced classification of sounds generated in construction sites.

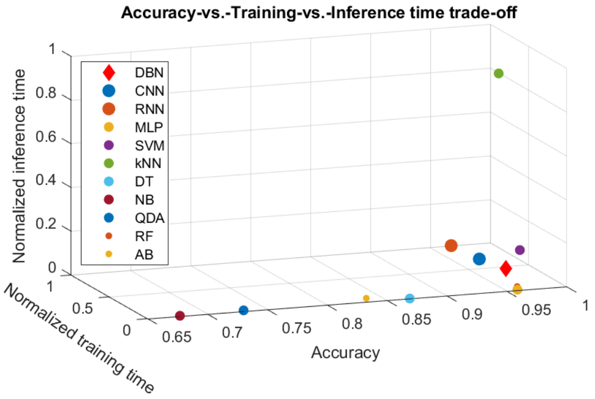


Figure 8: Scatterplot of the trade-off between accuracy vs. training time vs. inference time of the compared approaches.

## 6. Practical application

In this section, we aim at presenting a potential real-time application of the proposed architecture. Specifically, the inference results of a classification algorithm can rapidly degrade with the changing of the environmental condition in which audio sources are recorded. For example, the distance of the recording equipments, the number of different active tools and/or machinery, the closeness to a railway or other noisy sources, etc., can strongly reduce the obtained accuracy. Also the material on which the machinery is working can cause a performance degradation.

In order to overcome this issue, we propose a majority voting approach over time windows that collects a certain number of adjacent frames. Specifically, the recorded signal is analyzed in a longer window of duration $T_w$, for example, of 4–5 seconds or longer, if there is not an urgent real-time need. Since the feature for the inference are extracted for each frame of duration $T_F$, a total of $Q = T_w/T_F$ frames for windows are available, each of which produces an estimated class label. A decision can then be made by the majority of the produced labels over each window, as detailed in Fig. 9. This is similar to the ensemble majority voting approach, well known in machine learning (Alpaydin, 2014). Class labels predicted in each window, along with the associated information (like, e.g., ID of the related microphone, timestamps, etc.), can be easily saved into a database for future analytics and inferences.

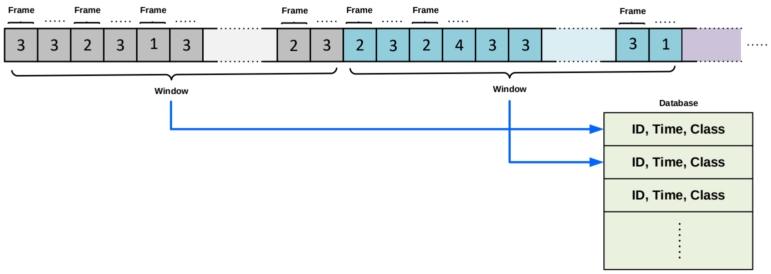


Figure 9: Basic idea of the proposed majority approach over a large window. In this example, the majority of frames over the first window produces the class 3 as the estimated label, even if some frames disagree since they estimate classes 1 or 2. A similar behavior is obtained for the second window and windows hereafter. The predicted class labels are then saved into a database along with related information.

In equation, the class label $\widehat{y}$ over a window can be estimated as:

$$\widehat{y} = \arg\max_{k=1}^{K} \sum_{i=1}^{Q} \widehat{y}_{i,k}, \tag{15}$$

where $\widehat{y}_{i,k}$ is equal to 1 if the predicted label in the $i$-th frame matches the $k$-th class, 0 otherwise. Hence, the effectiveness of the predicted label depends on a suitable choice of the number $Q$ of frames. We expect that a low frame count implies a non-robust classification, since the could happen that there is no a predominant class among the others. On the contrary, a high value of $Q$ causes the temporal resolution to be lost, which is instead an important prerogative in real-time applications.

As a practical example, we propose some numerical results related to a subset of the considered classes, for which the dataset provides recordings in different environmental conditions. Specifically, for class Hitachi50U it is available an on-board recording that strongly differs from the out-door one. On the contrary, for classes JD50D and JD50G, there are available other recordings made at different distances and with different environmental noise in background and ground material. A total of 5 minutes for each class have been recorded and tested. Results related to these classes, in terms of true positive rate, can be found in Table 9 that clearly shows the effectiveness of the proposed idea. Sounds recorded in different positions and with different background noise level are well classified by using a suitable number $Q$ of frames. A good trade-off between the obtained true positive rate and the temporal resolution is by considering a number $Q = 100$ of frames, for which the true positive rate is close to 97% for Hitachi50U and JD50D classes, and up to 91% for the JD50G one, and the window has a length of $T_w = Q \times T_F = 10$ s, appropriate for typical construction site activities. We observe in Table 9 that low values of $Q$ produce a not adequate true positive rate, which is increased by increasing $Q$ up to 100–120, then it decreases again for higher values of $Q$. In the interval 100–120, the true positive rate tends to remain constant at a value close to its maximum. In order to prefer an adequate time resolution, we have chosen $Q = 100$ frames.

Table 9: True positive rate (in %) obtained for a different number $Q$ of frames per window for the considered classes.

| **Class** | **Q** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 70 | 100 | 120 | 150 | 180 |
| Hitachi50U | 82.00 | 84.67 | 88.00 | 89.67 | 91.67 | 95.67 | 96.67 | 96.67 | 95.67 | 93.67 |
| JD50D | 84.33 | 90.00 | 95.00 | 96.33 | 96.67 | 96.67 | 96.67 | 96.67 | 95.67 | 94.33 |
| JD50G | 83.33 | 86.67 | 88.42 | 88.60 | 88.60 | 90.00 | 91.23 | 90.80 | 89.20 | 86.67 |

## 7. Conclusion

In this paper, we have presented a deep belief network (DBN) approach for the classification of audio data of construction work and equipment operations. The primary contribution of this study resides in that such architecture works with small audio frames and, for practical applications, the ability to perform a classification using very short samples with a low inference time can lead to the possibility to use such a network in time-critical applications in construction sites that require fast and reliable responses, such as hazard detection and activity monitoring. Up to now, the proposed architecture was tested on ten classes related to vehicles and tools, obtaining an overall accuracy up to 98%. The trade-off between accuracy, training time, and inference time overcomes results obtained by existing studies using other machine/deep learning algorithms and confirms that the proposed DBN can be a suitable approach for the classification of audio recorded on construction sites. The input to the DBN consists in the concatenation of six aggregate statistics extracted from the spectral features evaluated by the mel-spectrogram of each audio frame. A practical and real-time application of the proposed method has been also proposed in order to apply the classification scheme to sound data recorded in different environmental scenarios.

Future works will be conducted with an increased number of classes to cover diverse tools and vehicles employed in building sites, in order to lead to a more reliable and useful system. In addition, with the well-performed classification framework, the authors will build a real system that encompasses a wearable device for capturing and identifying sound data in real time and rapidly reflecting the results into physical environment. Moreover, the most interesting way to extend the work would be to test other deep learning architectures and try to combine different architectures in order to establish which kind of neural network approach can help the audio classification in construction sites.

## Acknowledgements

The authors would like to thank the associate editor and all anonymous reviewers for their precious and helpful comments and suggestions.

This work has been supported by the projects: “End-to-End Learning for 3D Acoustic Scene Analysis (ELeSA)” funded by Sapienza University of Rome Bando 2018; “SoFT: Fog of Social IoT”, funded by Sapienza University of Rome Bando 2018 and

2019; and, “DeepFog – Optimized distributed implementation of Deep Learning models over networked multitier Fog platforms for IoT stream applications” funded by Sapienza University of Rome Bando 2020.

## References

Abdoli, S., Cardinal, P., & Koerich, A. L. (2019). End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Systems with Applications*, *136*, 252–263. doi:10.1016/j.eswa.2019.06.040.

Abeßer, J. (2020). A review of deep learning based methods for acoustic scene classification. *Applied Sciences*, *10*, 1–16. doi:10.3390/app10062020.

Abu-El-Quran, A. (2006). Security monitoring using microphone arrays and audio classification. *IEEE Transactions on Instrumentation and Measurement*, *55*, 1025–1032. doi:10.1109/TIM.2006.876394.

Ahmad, S., Agrawal, S., Joshi, S., Taran, S., Bajaj, V., Demir, F., & Sengur, A. (2020). Environmental sound classification using optimum allocation sampling based empirical mode decomposition. *Physica A: Statistical Mechanics and its Applications*, *537*, 1–11. doi:10.1016/j.physa.2019.122613.

Aldeman, M., Bacchus, R., Chelliah, K., Patel, H., Raman, G., & Roberson, D. (2016). Aircraft noise monitoring using multiple passive data streams. *Noise & Vibration Worldwide*, *47*, 35–45. doi:10.1177/0957456516663329.

Alpaydin, E. (2014). *Introduction to Machine Learning*. (3rd ed.). Cambridge, MA: Mit Press.

Atrey, P. K., Maddage, N. C., & Kankanhalli, M. S. (2006). Audio based event detection for multimedia surveillance. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2006)* (pp. 1–5). volume 5. doi:10.1109/ICASSP.2006.1661400.

Bae, S. H., Choi, I., & Kim, N. S. (2016). Acoustic scene classification using parallel combination of LSTM and CNN. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE 2016)* (pp. 1–5). Budapest, Hungary.

Barchiesi, D., Giannoulis, D., Stowell, D., & Plumbley, M. D. (2015). Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, *32*, 16–34. doi:10.1109/MSP.2014.2326181.

Boddapati, V., Petef, A., Rasmusson, J., & Lundberg, L. (2017). Classifying environmental sounds using image recognition networks. *Procedia Computer Science*, *112*, 2048–2056. doi:10.1016/j.procs.2017.08.250.

Bondarenko, A., & Borisov, A. (2013). Research on the classification ability of deep belief networks on small and medium datasets. *International Journal of Information Technology and Management*, *16*, 60–65. doi:10.2478/itms-2013-0009.

Chachada, S., & Jay Kuo, C. C. (2013). Environmental sound recognition: A survey. In *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference* (pp. 1–9). doi:10.1109/APSIPA.2013.6694338.

Chen, X.-W., & Lin, X. (2014). Big data deep learning: Challenges and perspectives. *IEEE Access*, *2*, 514–525. doi:10.1109/ACCESS.2014.2325029.

Cheng, C.-F., Rashidi, A., Davenport, M. A., & Anderson, D. V. (2017). Activity analysis of construction equipment using audio signals and support vector machines. *Automation in Construction*, *81*, 240–253. doi:10.1016/j.autcon.2017.06.005.

Cheng, T., & Teizer, J. (2013). Real-time resource location data collection and visualization technology for construction safety and activity monitoring applications. *Automation in Construction*, *34*, 3–15. doi:10.1016/j.autcon.2012.10.017.

Cho, C., Lee, Y.-C., & Zhang, T. (2017). Sound recognition techniques for multi-layered construction activities and events. *Computing in Civil Engineering*, *2017*, 326–334. doi:10.1061/9780784480847.041.

Chu, S., Narayanan, S., & Kuo, C. C. J. (2009). Environmental sound recognition with time-frequency audio features. *IEEE Transactions on Audio, Speech and Language Processing*, *17*, 1142–1158. doi:10.1109/TASL.2009.2017438.

Dhanalakshmi, P., Palanivel, S., & Ramalingam, V. (2009). Classification of audio signals using SVM and RBFNN. *Expert Systems with Applications*, *36*, 6069–6075. doi:10.1016/j.eswa.2008.06.126.

Duarte, M. F., & Hu, Y. H. (2004). Vehicle classification in distributed sensor networks. *Journal of Parallel and Distributed Computing*, *64*, 826–838. doi:10.1016/j.jpdc.2004.03.020.

Fischer, A., & Igel, C. (2014). Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, *47*, 25–39. doi:10.1016/j.patcog.2013.05.025.

Fu, Z., Lu, G., Ting, K. M., & Zhang, D. (2011). A survey of audio-based music classification and annotation. *IEEE Transactions on Multimedia*, *13*, 303–319. doi:10.1109/TMM.2010.2098858.

Gencoglu, O., Virtanen, T., & Huttunen, H. (2014). Recognition of acoustic events using deep neural networks. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO 2014)* (pp. 506–510). Lisbon, Portugal.

Golparvar-Fard, M., Peña-Mora, F., & Savarese, S. (2015). Automated progress monitoring using unordered daily construction photographs and IFC-based building information models. *Journal of Computing in Civil Engineering*, *29*, 1–19. doi:10.1061/(ASCE)CP.1943-5487.0000205.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. Cambridge, MA: The MIT Press.

Hamel, P., & Eck, D. (2010). Learning features from music audio with deep belief networks. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR 2010)* (pp. 339–344).

Heittola, T., Cakir, E., & Virtanen, T. (2018). The machine learning approach for analysis of sound scenes and events. In T. Virtanen, M. D. Plumbley, & D. Ellis (Eds.), *Computational Analysis of Sound Scenes and Events* (pp. 13–40). Springer. doi:10.1007/978-3-319-63450-0_2.

Hinton, G. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*, 1771–1800. doi:10.1162/089976602760128018.

Hinton, G., Deng, L., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, *29*, 82–97. doi:10.1109/MSP.2012.2205597.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*, 1527–1554. doi:10.1162/neco.2006.18.7.1527.

Hiyane, J. I. K. (2001). Non-speech sound recognition with microphone array. In *International Workshop on Hands-Free Speech Communication (HSC 2001)* (pp. 107–110). Kyoto, Japan.

Hsieh, J.-W., Yu, S.-H., Chen, Y.-S., & Hu, W.-F. (2006). Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Transactions on Intelligent Transportation Systems*, *7*, 175–187. doi:10.1109/TITS.2006.874722.

Jiang, D.-N., Lu, L., Zhang, H.-J., Tao, J.-H., & Cai, L.-H. (2002). Music type classification by spectral contrast feature. In *2002 IEEE International Conference on Multimedia and Expo (ICME'02)* (pp. 113–116). Lausanne, Switzerland. doi:10.1109/ICME.2002.1035731.

Jin, X., Mukherjee, K., Gupta, S., Ray, A., Phoha, S., & Damarla, T. (2009). Asynchronous data-driven classification of weapon systems. *Measurement Science and Technology*, *20*, 123001. doi:10.1088/0957-0233/20/12/123001.

Khosrowpour, A., Niebles, J. C., & Golparvar-Fard, M. (2014). Vision-based workface assessment using depth images for activity analysis of interior construction operations. *Automation in Construction*, *48*, 74–87. doi:10.1016/j.autcon.2014.08.003.

Kim, H., Ham, Y., Kim, W., Park, S., & Kim, H. (2019). Vision-based nonintrusive context documentation for earthmoving productivity simulation. *Automation in Construction*, *102*, 135–147. doi:10.1016/j.autcon.2019.02.006.

Le Roux, N., & Bengio, Y. (2010). Deep belief networks are compact universal approximators. *Neural Computation*, *22*, 2192–2207. doi:10.1162/neco.2010.08-09-1081.

Lee, H., Largman, Y., Pham, P., & Ng, A. Y. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09)* (pp. 1096–1104).

Lee, Y.-C., Scarpiniti, M., & Uncini, A. (2020). Advanced sound classifiers and performance analyses for accurate audio-based construction project monitoring. *The ASCE Journal of Computing in Civil Engineering*, *34*, 1–11. doi:10.1061/(ASCE)CP.1943-5487.0000911. Paper 04020030.

Li, S., Yao, Y., Hu, J., Liu, G., Yao, X., & Hu, J. (2018). An ensemble stacked convolutional neural network model for environmental event sound recognition. *Applied Sciences*, *8*. doi:10.3390/app8071152.

Li, Z., Cai, X., Liu, Y., & Zhu, B. (2019). A novel Gaussian-Bernoulli based convolutional deep belief networks for image feature extraction. *Neural Processing Letters*, *49*, 305–319. doi:10.1007/s11063-017-9751-y.

Lu, L., Zhang, H.-J., & Jiang, H. (2002). Content analysis for audio classification and segmentation. *IEEE Transactions on Speech and Audio Processing*, *10*, 504–516. doi:10.1109/TSA.2002.804546.

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605.

Maccagno, A., Mastropietro, A., Mazziotta, U., Scarpiniti, M., Lee, Y.-C., & Uncini, A. (2021). A CNN approach for audio classification in construction sites. In A. Esposito, M. Faundez-Zanuy, F. C. Morabito, & E. Pasero (Eds.), *Progresses in Artificial Intelligence and Neural Systems* (pp. 371–381). Springer volume 184 of *Smart Innovation, Systems and Technologies*. doi:10.1007/978-981-15-5093-5_33.

Maganti, H. K., Motlicek, P., & Gatica-Perez, D. (2007). Unsupervised speech/nonspeech detection for automatic speech recognition in meeting rooms. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)* (pp. 1037–1040). volume 4. doi:10.1109/ICASSP.2007.367250.

Maijala, P., Shuyang, Z., Heittola, T., & Virtanen, T. (2018). Environmental noise monitoring using source classification in sensors. *Applied Acoustics*, *129*, 258–267. doi:10.1016/j.apacoust.2017.08.006.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference (SciPy 2015)* (pp. 18–24). volume 8. doi:10.25080/Majora-7b98e3ed-003.

Medhat, F., Chesmore, D., & Robinson, J. (2020). Masked conditional neural networks for sound classification. *Applied Soft Computing*, *90*. doi:10.1016/j.asoc.2020.106073.

Mierswa, I., & Morik, K. (2005). Automatic feature extraction for classifying audio data. *Machine Learning*, *58*, 127–149. doi:10.1007/s10994-005-5824-7.

Mohamed, A., Dahl, G., & Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech and Language Processing*, *20*, 14–22. doi:10.1109/TASL.2011.2109382.

Montufar, G., , & Ay, N. (2011). Refinements of universal approximation results for deep belief networks and restricted Boltzmann machines. *Neural Computation*, *23*, 1306–1319. doi:10.1162/NECO_a_00113.

Navon, R., & Sacks, R. (2007). Assessing research issues in automated project performance control (APPC). *Automation in Construction*, *16*, 474–484. doi:10.1016/j.autcon.2006.08.001.

Phan, H., Koch, P., Katzberg, F., Maass, M., Mazur, R., & Mertins, A. (2017). Audio scene classification with deep recurrent neural networks. In *Proc. of Interspeech 2017* (pp. 3043–3047). doi:10.21437/Interspeech.2017-101.

Piczak, K. J. (2015a). Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP 2015)* (pp. 1–6). Boston, MA, USA. doi:10.1109/MLSP.2015.7324337.

Piczak, K. J. (2015b). ESC: dataset for environmental sound classification. In *Proceedings of the 23rd ACM International Conference on Multimedia* (pp. 1015–1018). doi:10.1145/2733373.2806390.

Pinaya, W. H. L., Gadelha, A., Doyle, O. M., Noto, C., Zugman, A., Cordeiro, Q., Jackowski, A. P., Bressan, R. A., & Sato, J. R. (2016). Using deep belief network modelling to characterize differences in brain morphometry in schizophrenia. *Scientific Reports*, *6*, 1–9. doi:10.1038/srep38897. Paper 38897.

Powers, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, *2*, 37–63. doi:10.9735/2229-3981.

Raina, R., Madhavan, A., & Ng, A. (2009). Large-scale deep unsupervised learning using graphics processors. In *Proc. 26th International Conference on Machine Learning (ICML 2009)* (pp. 873–880). Montreal, QC, Canada. doi:10.1145/1553374.1553486.

Rashid, K. M., & Louis, J. (2019). Times-series data augmentation and deep learning for construction equipment activity recognition. *Advanced Engineering Informatics*, *42*, 100944. doi:10.1016/j.aei.2019.100944.

Salamon, J., & Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, *24*, 279–283. doi:10.1109/LSP.2017.2657381.

Sallai, J., Hedgecock, W., Volgyesi, P., Nadas, A., Balogh, G., & Ledeczi, A. (2011). Weapon classification and shooter localization using distributed multichannel acoustic sensors. *Journal of Systems Architecture*, *57*, 869–885. doi:10.1016/j.sysarc.2011.04.003.

Sang, J., Park, S., & Lee, J. (2018). Convolutional recurrent neural networks for urban sound classification using raw waveforms. In *26th European Signal Processing Conference (EUSIPCO 2018)* (pp. 2458–2462). doi:10.23919/EUSIPCO.2018.8553247.

Scardapane, S., Comminiello, D., Scarpiniti, M., & Uncini, A. (2013). Music classification using extreme learning machines. In *2013 8th International Symposium on Image and Signal Processing and Analysis (ISPA 2013)* (pp. 377–381). IEEE. doi:10.1109/ISPA.2013.6703770.

Scardapane, S., Scarpiniti, M., Bucciarelli, M., Colone, F., Mansueto, M. V., & Parisi, R. (2015). Microphone array based classification for security monitoring in unstructured environments. *AEÜ – International Journal of Electronics and Communications*, *69*, 1715–1723. doi:10.1016/j.aeue.2015.08.007.

Scarpiniti, M., Comminiello, D., Uncini, A., & Lee, Y.-C. (2020). Deep recurrent neural networks for audio classification in construction sites. In *28th European Signal Processing Conference (EUSIPCO 2020)* (pp. 810–814). Amsterdam, The Netherlands. doi:10.23919/Eusipco47968.2020.9287802.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. doi:10.1016/j.neunet.2014.09.003.

Seo, J., Han, S., Lee, S., & Kim, H. (2015). Computer vision techniques for construction safety and health monitoring. *Advanced Engineering Informatics*, *29*, 239–251. doi:10.1016/j.aei.2015.02.001.

Sharan, R. V., & Moir, T. J. (2016). An overview of applications and advancements in automatic sound recognition. *Neurocomputing*, *200*, 22–34. doi:10.1016/j.neucom.2016.03.020.

Sherafat, B., Ahn, C. R., Akhavian, R., Behzadan, A. H., Golparvar-Fard, M., Kim, H., Lee, Y.-C., Rashidi, A., & Azar, E. R. (2020). Automated methods for activity recognition of construction workers and equipment: State-of-the-art review. *Journal of Construction Engineering and Management*, *146*, 1–19. doi:10.1061/(ASCE)CO.1943-7862.0001843. Paper 0312000.

Sherafat, B., Rashidi, A., Lee, Y.-C., & Ahn, C. R. (2019). Hybrid kinematic-acoustic system for automated activity detection of construction equipment. *Sensors*, *19*, 1–21. doi:10.3390/s19194286.

Mendes da Silva, A. C., Coelho, M. A. N., & Fonseca Neto, R. (2020). A music classification model based on metric learning applied to MP3 audio files. *Expert Systems with Applications*, *144*, 1–13. doi:10.1016/j.eswa.2019.113071.

Su, F., Yang, L., Lu, T., & Wang, G. (2011). Environmental sound classification for scene recognition using local discriminant bases and HMM. In *Proceedings of the 19th ACM International Conference on Multimedia (MM'11)* (pp. 1389–1392). doi:10.1145/2072298.2072022.

Taghaddos, H., Mashayekhi, A., & Sherafat, B. (2016). Automation of construction quantity take-off: Using building information modeling (BIM). In *Construction Research Congress 2016* (pp. 2218 – 2227). doi:10.1061/9780784479827.221.

Tokozume, Y., & Harada, T. (2017). Learning environmental sounds with end-to-end convolutional neural network. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017)* (pp. 2721–2725). New Orleans, LA, USA. doi:10.1109/ICASSP.2017.7952651.

Wang, D., & Brown, G. J. (2006). *Computational auditory scene analysis: principles, algorithms, and applications*. Hoboken, NJ: Wiley–IEEE Press.

Xie, Y., Lee, Y.-C., Huther da Costa, T., Park, J., Jui, J. H., Choi, J. W., & Zhang, Z. (2019). Construction data-driven dynamic sound data training and hardware requirements for autonomous audio-based site monitoring. In *36th International Symposium on Automation and Robotics in Construction (ISARC 2019)* (pp. 1011–1017). doi:10.22260/ISARC2019/0135.

Xue, L., & Su, F. (2015). Auditory scene classification with deep belief network. In *International Conference on Multimedia Modeling (MMM 2015)* (pp. 348–359). doi:10.1007/978-3-319-14445-0_30.

Zhang, T., Lee, Y.-C., Scarpiniti, M., & Uncini, A. (2018). A supervised machine learning-based sound identification for construction activity monitoring and performance evaluation. In *Proc. of 2018 Construction Research Congress (CRC 2018)* (pp. 358–366). New Orleans, Louisiana, USA. doi:10.1061/9780784481264.035.

Zhang, X.-L., & Wu, J. (2013). Deep belief networks based voice activity detection. *IEEE Transactions on Audio, Speech, and Language Processing*, *21*, 697–710. doi:10.1109/TASL.2012.2229986.

Zhang, Z., Liu, Z., Sinclair, M., Acero, A., Deng, L., Droppo, J., Huang, X., & Zheng, Y. (2004). Multi-sensory microphones for robust speech detection, enhancement and recognition. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2004)* (pp. 781–784). volume 3. doi:10.1109/ICASSP.2004.1326661.

Zheng, F., Zhang, G., & Song, Z. (2001). Comparison of different implementations of MFCC. *Journal of Computer Science and Technology*, *16*, 582–589. doi:10.1007/BF02943243.