ONLINE LEARNING OF TIME-FREQUENCY PATTERNS

Jose F. Ruiz-Muñoz[1], *Raviv Raich*[2], *Mauricio Orozco-Alzate*[1] *and Xiaoli Z. Fern*[2]

[1]Universidad Nacional de Colombia - Sede Manizales - Signal Processing and Recognition Group, Manizales, 17004-Colombia
[2]School of EECS, Oregon State University, Corvallis, OR 97331-5505, USA

ABSTRACT

We present an online method to learn recurring time-frequency patterns from spectrograms. Our method relies on a convolutive decomposition that estimates sequences of spectra into time-frequency patterns and their corresponding activation signals. This method processes one spectrogram at a time such that in comparison with a batch method, the computational cost is reduced proportionally to the number of considered spectrograms. We use a first-order stochastic gradient descent and show that a monotonically decreasing learning-rate works appropriately. Furthermore, we suggest a framework to classify spectrograms based on the estimated set of time-frequency patterns. Results, on a set of synthetically generated spectrograms and a real-world dataset, show that our method finds meaningful time-frequency patterns and that it is suitable to handle a large amount of data.

Index Terms— dictionary learning, non-negative matrix factorization, online learning, classification.

1. Introduction

Learning time-frequency patterns is helpful for both supervised and unsupervised analyses of acoustic signals. For this purpose, the mathematical model known as dictionary learning (DL) has been used. Estimation of such a model is usually formulated as a constrained optimization problem that includes a data fit term between the signal and a combination of a set of patterns —called *dictionary*— and their corresponding coefficients for weighting those patterns —called *activations*.

Depending on the problem, a physical meaning can be attributed to patterns and coefficients [1]. For example, for bioacoustic signals, dictionary patterns can be associated with different sound sources, e.g., bird species vocalization, and coefficients can be related to the time when the vocalizations are emitted. For later analysis, a DL algorithm should appropriately recover the original signal and satisfy the constraints, e.g., norm-constraints or non-negativity. Nevertheless, those algorithms are usually computationally expensive; therefore, to scale up and allow handling a large amount of data, it is important to consider complexity and memory requirements [2].

One approach for DL, which has been widely applied in machine learning and digital signal processing, is based on nonnegative matrix factorization (NMF) [3]. Particularly, NMF allows extracting meaningful information from audio recordings that contain mixtures of sounds [4, 5]. In order to apply NMF, the audio signal is usually represented by its spectrogram [6–8]. NMF has been successfully applied to various audio applications including automatic transcription, music analyses and blind source separation [9, 10]. NMF is formulated as an optimization problem (sparsity constraints are often added) that minimizes the least-squares error or the generalized Kullback-Leibler divergence [11] between the measured signal and its decomposition.

Using NMF a spectrum is decomposed into a product of two matrices: one corresponding to a collection of 1-D spectra (which forms the dictionary) and another corresponding to their activations in time. An alternative model is the convolutive non-negative matrix factorization (cNMF) in which each pattern of the dictionary is a matrix that corresponds to a sequence of 1-D spectra (time-frequency pattern) [12, 13]. The resulting time-frequency patterns provide useful information related to relevant temporal structures contained in the recordings [14]. Nevertheless, when dealing with large data (e.g., in bioacoustics), traditional cNMF algorithms become computationally expensive and demand large memory resources. To reduce the computational complexity and memory consumption, low-rank approximations are applied [15]. However, this approach generally results in information loss. An alternative approach to alleviate the processing requirements is using online algorithms. For instance, in [16], an algorithm for learning 1-D patterns using stochastic gradient descent is proposed and in [17], an online version of the cNMF algorithm proposed in [18] is introduced.

In this paper, we propose an unsupervised online version of the algorithm originally presented in [19]. For this purpose, we use a first-order stochastic gradient descent approach. Our algorithm progressively updates the dictionary with each incoming spectrogram. Additionally, we propose a scheme for classifying audio signals based on features extracted from the convolutive decomposition of the spectrograms. We evaluate and compare the proposed approach on synthetic and real-world datasets.

2. Learning time-frequency patterns

2.1. Convolutive decomposition

We approximate a spectrogram $\boldsymbol{Y} \in \mathbb{R}^{F\times T}$ by the linear combination of K shifted time-frequency patterns $\boldsymbol{D}_k = [\boldsymbol{d}_{k1} \dots \boldsymbol{d}_{kF}]^\top \in \mathbb{R}^{F\times W}$ where $\boldsymbol{d}_{kf} \in \mathbb{R}^{W\times 1}$ is the k-th time

This work is partially supported by the National Science Foundation grants CCF-1254218, DBI-1356792 and IIS-1055113, and "Convocatoria 567 de 2012-Colciencias".

pattern at frequency f, and W is the length of each time-frequency pattern. This approximation is expressed by the discrete convolution operation[1] as follows

$$\boldsymbol{Y}(f,t) \approx \sum_{k=1}^{K} [\boldsymbol{a}_k * \boldsymbol{d}_{kf}](t) \tag{1}$$

where $\boldsymbol{Y}(f,t)$ is an entry of $\boldsymbol{Y} \in \mathbb{R}^{F\times T}$ at frequency $f \in [1,F]$ and time $t \in [1,T]$, and $\boldsymbol{a}_k = [\boldsymbol{a}_k(1)\dots\boldsymbol{a}_k(L)]^\top \in \mathbb{R}^{L\times 1}$ ($L = T+W-1$) is the activation signal corresponding to $\boldsymbol{D}_k$. The convolution is performed without zero-padded edges; therefore, the convolution between $\boldsymbol{a}_k \in \mathbb{R}^{L\times 1}$ and $\boldsymbol{d}_{kf} \in \mathbb{R}^{W\times 1}$ produces a vector of length T, i.e., $[\boldsymbol{a}_k * \boldsymbol{d}_{kf}] \in \mathbb{R}^{T\times 1}$. The full dictionary $\boldsymbol{D}$ is built by stacking all $\boldsymbol{D}_k$, such that $\boldsymbol{D} \in \mathbb{R}^{K\times F\times W}$. Similarly, the set of activation signals $\boldsymbol{a}_k$ forms the matrix $\boldsymbol{A} = [\boldsymbol{a}_1\dots\boldsymbol{a}_K] \in \mathbb{R}^{L\times K}$.

Dictionary and activations are estimated by solving an optimization problem that aims to minimize the least-squares error and the L_1-norm of the activations to induce sparsity:

$$\begin{aligned} &\min_{\boldsymbol{D},\boldsymbol{A}} \ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A}) \\ &\ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A}) := \frac{1}{2}\sum_{f=1}^{F}\sum_{t=1}^{T}\Big(\boldsymbol{Y}(f,t) - \sum_{k=1}^{K}[\boldsymbol{a}_k * \boldsymbol{d}_{kf}](t)\Big)^2 \\ &\qquad +\lambda\sum_{k=1}^{K}\sum_{t=1}^{L}|\boldsymbol{a}_k(t)| \\ &\text{subject to } \sum_{f=1}^{F}\sum_{t=1}^{W}(\boldsymbol{d}_{kf}(t))^2 \le 1,\ \forall\, 1\le k\le K. \end{aligned} \tag{2}$$

In [19], an iterative rule for updating the k-th time-frequency pattern is proposed (based on a convexification procedure by a surrogate loss function [20]) as follows:

$$\boldsymbol{D}_{k(p)} = \Pi(\boldsymbol{D}_{k(p-1)} + \eta_d \nabla_{\boldsymbol{D}_k}\ell(\boldsymbol{Y},\boldsymbol{D}_{k(p-1)},\boldsymbol{A})), \tag{3}$$

where (p) denotes the current iteration, the projection Π is defined as

$$\Pi(\boldsymbol{D}) = \begin{cases} \boldsymbol{D} \text{ if } \|\boldsymbol{D}\| \le 1 \\ \frac{\boldsymbol{D}}{\|\boldsymbol{D}\|} \text{ otherwise} \end{cases}, \forall \boldsymbol{D}$$

η_d is the step-size, and the gradient of the loss function wrt $\boldsymbol{D}_k$ is

$$\nabla_{\boldsymbol{D}_k}\ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A}) = [\boldsymbol{v}_{\boldsymbol{d}_{k1}}\dots\boldsymbol{v}_{\boldsymbol{d}_{kF}}]^\top \in \mathbb{R}^{F\times W} \tag{4}$$

where $\boldsymbol{v}_{\boldsymbol{d}_{kf}} = \boldsymbol{T}_{\boldsymbol{a}_k}^\top[\boldsymbol{y}_f - \sum_{k=1}^{K}\boldsymbol{T}_{\boldsymbol{a}_k}\boldsymbol{d}_{kf}^{(p-1)}] \in \mathbb{R}^{W\times 1}$, $\boldsymbol{y}_f = [\boldsymbol{Y}(f,1)\dots\ \boldsymbol{Y}(f,T)]^\top \in \mathbb{R}^{T\times 1}$ and $\boldsymbol{T}_{\boldsymbol{a}_k} = \text{TOEPLITZ}(\boldsymbol{a}_k, W, L, W) \in \mathbb{R}^{T\times W}$ (see Appendix A). A safe step-size $\eta_d = 1/\max_f \gamma_f$ where $\gamma_f = \lambda_{\max}([\boldsymbol{u}_1,\boldsymbol{u}_2]^\top\boldsymbol{T}_A^\top\boldsymbol{T}_A[\boldsymbol{u}_1,\boldsymbol{u}_2])$, $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue, $\boldsymbol{u}_1 = \boldsymbol{v}_{\boldsymbol{d}_{\cdot f}}/||\boldsymbol{v}_{\boldsymbol{d}_{\cdot f}}|| \in \mathbb{R}^{KW\times 1}$, $\boldsymbol{v}_{\boldsymbol{d}_{\cdot f}} = [\boldsymbol{v}_{\boldsymbol{d}_{1f}}^\top\dots\boldsymbol{v}_{\boldsymbol{d}_{Kf}}^\top]^\top \in \mathbb{R}^{KW\times 1}$, $\boldsymbol{u}_2 = \tilde{\boldsymbol{d}}_{\cdot f}/||\tilde{\boldsymbol{d}}_{\cdot f}|| \in \mathbb{R}^{KW\times 1}$, $\tilde{\boldsymbol{d}}_{\cdot f} = \boldsymbol{d}_{\cdot f} - (\boldsymbol{d}_{\cdot f}^\top\boldsymbol{u}_1)\boldsymbol{u}_1 \in \mathbb{R}^{KW\times 1}$, $\boldsymbol{d}_{\cdot f} = [\boldsymbol{d}_{1f}^\top\dots\boldsymbol{d}_{Kf}^\top]^\top \in \mathbb{R}^{KW\times 1}$, and $\boldsymbol{T}_A = [\boldsymbol{T}_{\boldsymbol{a}_1}\dots\boldsymbol{T}_{\boldsymbol{a}_K}] \in \mathbb{R}^{T\times KW}$. An update rule for activations is also given in [19]:

$$\boldsymbol{A}_{(p)} = \arg\min_{\boldsymbol{A}} \ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A}_{(p-1)}). \tag{5}$$

[1] Discrete convolution operation: $(\boldsymbol{u}*\boldsymbol{v})(n) = \sum_m u(n-m+1)v(m)$

2.2. Online dictionary learning

In order to learn a dictionary from a set of N stacked spectrograms $\{\boldsymbol{Y}^{(1)}\dots\boldsymbol{Y}^{(N)}\}$, in [19], the update rule of (3) is applied as follows:

$$\boldsymbol{D}_{k(p)} = \Pi(\boldsymbol{D}_{k(p-1)} + \frac{1}{N}\sum_{i=1}^{N}\eta_d^{(i)}\nabla_{\boldsymbol{D}_k}\ell(\boldsymbol{Y}^{(i)},\boldsymbol{D}_{(p-1)},\boldsymbol{A}^{(i)})). \tag{6}$$

Alternatively, we propose an online algorithm that updates the time-frequency patterns according to the current spectrogram and the ones observed in the past. Therefore, we define the following loss function:

$$g_N(\boldsymbol{D}) := \frac{1}{N}\sum_{i=1}^{N}\ell(\boldsymbol{Y}^{(i)},\boldsymbol{D},\boldsymbol{A}^{(i)})$$

where $\boldsymbol{A}^{(i)}$ is the estimated activation matrix that corresponds to the i-th spectrogram $\boldsymbol{Y}^{(i)}$. Hence, the dictionary learning task consists of minimizing the expected cost

$$g(\boldsymbol{D}) := \mathbb{E}_{\boldsymbol{Y}}[\ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A})] := \lim_{N\to\infty} g_N(\mathbf{D}).$$

For this purpose, we update $\boldsymbol{D}_k$ by using the first-order stochastic gradient descent algorithm [16, 21] as follows:

$$\boldsymbol{D}_{k(p)} = \Pi(\boldsymbol{D}_{k(p-1)} + \mu_p\eta_d\nabla_{\boldsymbol{D}_k}\ell(\boldsymbol{Y}^{(i)},\boldsymbol{D}_{(p-1)},\boldsymbol{A}^{(\boldsymbol{i})})) \tag{7}$$

where

$$i = \begin{cases} N & \text{if } \text{mod}(p,N) = 0 \\ \text{mod}(p,N) & \text{otherwise,} \end{cases}$$

and μ_p is the factor for scaling the gradient, also known as learning-rate. Notice that one iteration of (6) requires computing N times the gradient $\nabla_{\boldsymbol{D}_k}\ell(\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A})$ but (7) requires computing this gradient only once. According to [22], two learning-rate schedules commonly used in matrix factorization are:

- **Fixed Schedule (FS):** the learning rate $\mu_p = \alpha\ \forall p$ is fixed throughout the online learning process.
- **Monotonically Decreasing Schedule (MDS):** the learning rate monotonically decreases each time that a new spectrogram is observed. Two options are: i) MDS1: $\mu_p = \frac{\alpha}{p}$, and ii) MDS2: $\mu_p = \frac{\alpha}{\sqrt{p}}$.

Our online DL process, which aims to compute $\boldsymbol{D}$ and $\boldsymbol{A}$, alternatively updates both of them. Therefore, in the p-th iteration, activations are updated, as indicated in (5), by $\boldsymbol{A}_{(p)}^{(i)} = \arg\min_{\boldsymbol{A}^{(i)}} \ell(\boldsymbol{Y}^{(i)},\boldsymbol{D}_{(p-1)},\boldsymbol{A}_{(p-1)}^{(i)})\ \forall i$. Subsequently, the dictionary $\boldsymbol{D}_{(p-1)}$ is updated by (7). Note that due to the non-convex nature of the problem convergence to a global optimum is not guaranteed.

3. Classifying spectrograms

Our classification task consists of mapping the vector representation of a spectrogram $\boldsymbol{x} \in \mathbb{R}^K$ to a categorical (class) label $y \in \{-1,1\}$. The label in the binary classification setting indicates the presence ($y=1$) or absence ($y=-1$) of the target class in a given spectrogram.

We divide the experiments into two stages: training and test. In the training stage, the dictionary is estimated by

using the proposed online DL method, which receives a sequence of spectrograms. The estimated dictionary is used to extract the feature vector $\boldsymbol{x}_i = [x_{i1}, \ldots, x_{iK}] \in \mathbb{R}^K$ for the i-th spectrogram $\boldsymbol{Y}_i \in \mathbb{R}^{F \times T}$ in a training set, whose entries are computed as follows: $x_{ik} = \max_t |\sum_f \boldsymbol{h}_{kf}^{(i)}(t)|$ where $\boldsymbol{h}_{kf}^{(i)} = \overleftarrow{\boldsymbol{d}}_{kf} * \boldsymbol{y}_f^{(i)} \in \mathbb{R}^{(T+W-1)\times 1}$, and $\overleftarrow{\cdot}$ denotes the vector in reversed order.

A support vector machine (SVM) classifier is trained by using this representation, and the dictionary estimated in the training stage is used to compute the vector representation of the test set. Labels are assigned by the trained SVM.

4. Experiments

4.1. Experiments on an artificial dataset

Initially, we perform experiments in a collection of 1000 synthetically generated spectrograms containing some of six different time-frequency patterns. The three dimensional binary label vector of each spectrogram $\boldsymbol{Y} \in \mathbb{R}^{16\times 30}$ indicates the presence or absence of each class in the spectrogram. For each class, two types of time-frequency patterns of length 10 are considered. Therefore, the original dictionary is formed by six basic time-frequency patterns (see Fig. 3a). The free parameters in the proposed online DL method are: length of window W, number of dictionary words K, learning-rate μ, and ℓ_1-norm regularization parameter λ. We fix $W = 10$, since this parameter is known beforehand, and $K = 8$ (we over-estimate the size of the dictionary in order to avoid missing a time-frequency pattern). Estimation of the remaining parameters is described below.

We compare the schedules of μ described in Sec. 2.2 and tune the parameter α. Figure 1 contains the reconstruction error of a test set of 30 spectrograms and the actual sparsity of their activations (rate of non-zero entries) as a function of the number of observed spectrograms for a set of different values of α (for $\lambda = 0.1$). According to this experiment, the FS schedule works well for moderate values of α trading off initial instability at a large value of α with slow convergence for a small value of α.

Figure 2 shows the reconstruction error and the rate of non-zero entries in function of λ after observing 1000 spectrograms (the learning-rate is MDS1 for a set of different values of α). Results confirm the trade-off in the objective function between the reconstruction error and the ℓ_1-norm constraint. Figures 3a and 3b show the original set of time-frequency patterns and the estimated ones (with MDS1, $\alpha = 100$ and $\lambda = 0.1$), respectively.

4.2. Experiments on real-world datasets

To validate the proposed method, we perform experiments on the MLSP 2013 Bird Classification Challenge dataset,[2] which was collected in the H. J. Andrews (HJA) Long-Term Experimental Research Forest in Oregon (USA). Table 1 shows the number of recordings and classes of this dataset.

The classification experiments consider the following: i) for each class a binary (presence/absence) classification problem is considered; ii) the dataset is randomly divided into

[2] `https://www.kaggle.com/c/mlsp-2013-birds`

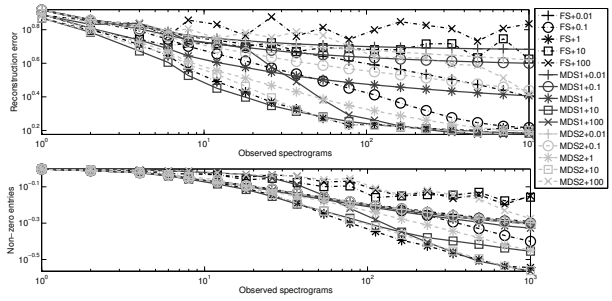


Fig. 1: Comparison of the studied learning-rate schedules, in a test set of 30 spectrograms, for a set of different values of α (FS+α, MDS1+α and MDS2+α) and $\lambda = 0.1$.

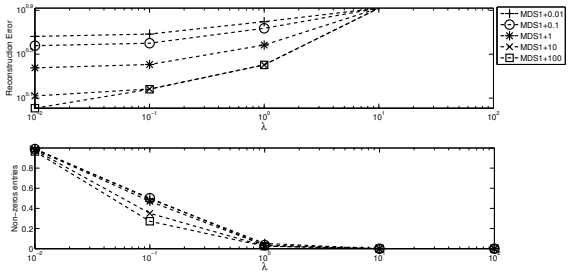
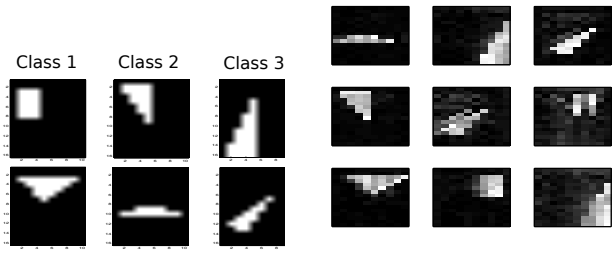


Fig. 2: Reconstruction error and rate of non-zero entries in function of λ after observing 1000 spectrograms (the learning-rate is MDS1 for a set of different values of α).

50% for training and 50% for test (with 20 repetitions); iii) spectrograms are computed with dimensions $F = 80$ and $T = 250$ (corresponding to 10 seconds); iv) the parameters of DL are: $W = 25$ (window length of 1 sec.), $K = 6$, λ is tuned for $\{0.01, 0.1, 1, 10\}$, learning-rate MDS1 where α is tuned for $\{1, 10, 100\}$, and 1000 iterations (due to the small size of the dataset, online DL cycles through the spectrograms several times to allow for a number of iterations that is greater than the number of spectrograms available in the dataset); v) feature representation as indicated in Sec. 3; vi) linear SVM classifier (the best regularization parameter C is searched using a cross-validation in the training set, such that $C \in [1 \times 10^{-2}, 1 \times 10^{2}]$); and, vii) performance is reported by the F-score.

Table 1 shows the classification performance of three methods: 1) *Wang et. al. (2013)* that considers the proposed classification framework but learns the dictionary by the online method proposed in [17] (using our own implementation); 2) *Online DL* that applies the proposed online DL method and classification framework; and, 3) *Frequency* that applies the proposed classification framework, but the feature representation is directly extracted from the spectrograms, i.e., the feature vector corresponds to the normalized average spectra. According to our results, the proposed *Online DL* outperforms the others in 12 of the 19 classes. *Frequency* outperforms the others when classifying `BRCR`, `VATH`, `BHGB`, and `MGWA`. Among these classes, the performance is remarkably high for `BHGB`, this suggests that the frequency band is enough to distinguish this species. *Wang et al. (2013)*



(a) Original dictionary (b) Estimated dictionary (with MDS1, $\alpha = 100$ and $\lambda = 0.1$)

Fig. 3: Original and estimated sets of time-frequency patterns.

Table 1: Number of 10-second recordings per species of MLSP2013 dataset. Size of the traing/test datasets. F-score performance of classification experiments (boldface indicates the highest result per species).

Label	# recordings	F-score performance		
		Wang et al.	Online DL	Frequency
BRCR	14	70.8 ± 7.5	92.8 ± 3.7	**94.1 ± 0.7**
PAWR	81	80.1 ± 3.4	**84.9 ± 1.3**	79.4 ± 2.4
PSFL	46	75.1 ± 5.3	**84.0 ± 3.4**	77.8 ± 2.4
RBNU	9	53.4 ± 10.8	**83.9 ± 8.7**	79.7 ± 7.6
DEJU	20	87.8 ± 4.3	**89.9 ± 4.7**	80.9 ± 2.4
OSFL	14	**90.0** ± 5.0	79.7 ± 7.3	88.6 ± 4.6
HETH	47	70.6 ± 5.5	**80.5 ± 5.1**	78.0 ± 2.5
CBCH	40	**83.9 ± 4.5**	74.7 ± 6.0	81.7 ± 1.5
VATH	61	74.7 ± 4.5	83.6 ± 3.0	**84.1 ± 0.6**
HEWA	53	75.5 ± 4.8	**80.7 ± 5.0**	77.7 ± 2.7
SWTH	103	70.0 ± 4.0	**82.4 ± 4.3**	77.2 ± 2.4
HAFL	28	81.6 ± 5.1	**85.8 ± 4.3**	74.1 ± 2.5
WETA	33	**88.1 ± 4.2**	75.3 ± 6.6	86.4 ± 0.8
BHGB	9	70.2 ± 9.1	67.5 ± 9.9	**95.5 ± 0.5**
GCKI	37	67.8 ± 5.6	**85.4 ± 6.0**	83.0 ± 1.4
WAVI	17	83.4 ± 4.9	**92.7 ± 6.8**	89.0 ± 1.1
MGWA	6	42.3 ± 10.6	77.3 ± 8.7	**86.3 ± 6.5**
STJA	10	86.7 ± 6.5	**94.3 ± 7.8**	93.6 ± 0.9
CONI	26	86.0 ± 3.5	**89.1 ± 4.6**	85.6 ± 1.4

produces the best performance when classifying OSFL, CBCH, and WETA.

4.3. Computational cost: Batch Learning vs Online Learning

In order to show the computational benefits of our method (online learning), we compare it against a batch learning approach. We call batch learning to the DL method that updates the time-frequency patterns by (6), which requires the whole set of spectrograms to estimate the gradient. These experiments were carried out on a CPU with Processor 2.20GHz × 8 and Memory 3.8 GB.

Figure 4 compares the time needed to reconstruct 20 (randomly selected) spectrograms from the MLSP 2013 dataset by batch learning and online learning. Note that since there are more than 20 iterations, the spectrograms are observed several times in the online case. We observe that the error is not monotonically decreasing at the beginning for online learning. However, the reconstruction error for both the online and batch methods converges to a similar value after several iterations. Furthermore, as expected, the online learning is faster than the batch learning by a factor of the number of spectrograms reconstructed at each iteration.

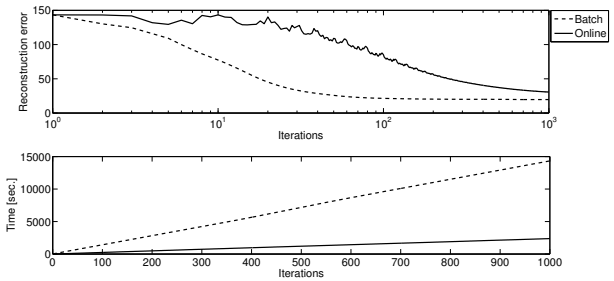


Fig. 4: Comparison of reconstruction error (top) and computational cost (bottom) between batch learning and online learning.

5. Conclusion

We described an online DL method based on stochastic gradient descent, which learns time-frequency patterns from large datasets of spectrograms. Our algorithm is based on a convolutive DL method with the additive update rule. The proposed method handles better the computational resources than its batch counterpart. Therefore, it could be preferred for analyzing large datasets. Experiments on an artificial dataset and a real-world dataset show that the method recovers appropriately the original spectrograms and finds meaningful time-frequency patterns for classification outperforming a state-of-the-art DL method and the classification based on the raw frequency information.

Appendix A. Toeplitz matrix

A Toeplitz matrix $\text{TOEPLITZ}(\boldsymbol{x}, \tau_c, \tau_1, \tau_2)$ is constructed as follows:

$$\begin{bmatrix} \phi(\boldsymbol{x},\tau_c) & \phi(\boldsymbol{x},\tau_c-1) & \cdots & \phi(\boldsymbol{x},\tau_c-\tau_2+1) \\ \phi(\boldsymbol{x},\tau_c+1) & \phi(\boldsymbol{x},\tau_c) & \cdots & \phi(\boldsymbol{x},\tau_c-\tau_2+2) \\ \vdots & \cdots & \cdots & \vdots \\ \phi(\boldsymbol{x},\tau_1) & \phi(\boldsymbol{x},\tau_1-1) & \cdots & \phi(\boldsymbol{x},\tau_1-\tau_2+1) \end{bmatrix}$$

where $\boldsymbol{x} \in \mathbb{R}^m$, $\tau_c, \tau_1, \tau_2 \in \mathbb{N}$ and

$$\phi(\boldsymbol{x},\tau) = \begin{cases} \boldsymbol{x}(\tau), & 1 \le \tau \le \dim(\boldsymbol{x}) \\ 0, & \text{otherwise.} \end{cases}$$

6. References

[1] Jingu Kim and Haesun Park, "Fast nonnegative matrix factorization: An active-set-like method and comparisons," *SIAM Journal on Scientific Computing*, vol. 33, no. 6, pp. 3261–3281, 2011.

[2] Maria G Jafari and Mark D Plumbley, "Fast dictionary learning for sparse representations of speech signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 1025–1031, 2011.

[3] Alexey Ozerov and Cédric Févotte, "Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 550–563, 2010.

[4] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*, John Wiley & Sons, 2009.

[5] Paris Smaragdis, Cedric Fevotte, Gautham J. Mysore, Nasser Mohammadiha, and Matthew Hoffman, "Static and Dynamic Source Separation Using Nonnegative Factorizations: A unified view," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 66–75, 2014.

[6] Roland Badeau and M Plumbley, "Multichannel high resolution NMF for modelling convolutive mixtures of non-stationary signals in the time-frequency domain," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 11, pp. 1670–1680, 2013.

[7] Cédric Févotte and Matthieu Kowalski, "Low-rank time-frequency synthesis," in *Advances in Neural Information Processing Systems*, 2014, pp. 3563–3571.

[8] Hideyuki Sawada, Hirokazu Kameoka, Shunsuke Araki, and Naonori Ueda, "Multichannel extensions of non-negative matrix factorization with complex-valued data," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 971–982, 2013.

[9] Andrzej Cichocki, Rafal Zdunek, and Shun-ichi Amari, "New algorithms for non-negative matrix factorization in applications to blind source separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings.* IEEE, 2006, vol. 5, pp. V–621–V–624.

[10] Qingju Liu, Wenwu Wang, Philip JB Jackson, Mark Barnard, Josef Kittler, and Jonathon Chambers, "Source separation of convolutive and noisy mixtures using audio-visual dictionary learning and probabilistic time-frequency masking," *IEEE Transactions on Signal Processing*, vol. 61, no. 22, pp. 5520–5535, 2013.

[11] Daniel D Lee and H Sebastian Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 556–562.

[12] Paul D. O'Grady and Barak A. Pearlmutter, "Discovering speech phones using convolutive non-negative matrix factorisation with a sparseness constraint," *Neurocomputing*, vol. 72, no. 1-3, pp. 88–101, 2008.

[13] Ruairí De Fréin and Scott T Rickard, "Learning speech features in the presence of noise: Sparse convolutive robust non-negative matrix factorization," in *16th International Conference on Digital Signal Processing.* IEEE, 2009, pp. 1–6.

[14] Paris Smaragdis, "Convolutive speech bases and their application to supervised speech separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 1–12, 2007.

[15] Guoxu Zhou, Andrzej Cichocki, and Shengli Xie, "Fast nonnegative matrix/tensor factorization based on low-rank approximation," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2928–2940, 2012.

[16] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro, "Online Learning for Matrix Factorization and Sparse Coding," *Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[17] Dong Wang, Ravichander Vipperla, Nicholas Evans, and Thomas Fang Zheng, "Online non-negative convolutive pattern learning for speech signals," *IEEE Transactions on Signal Processing*, vol. 61, no. 1, pp. 44–56, 2013.

[18] Wenwu Wang, "Convolutive non-negative sparse coding," in *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, pp. 3681–3684.

[19] J. F. Ruiz-Muñoz, Zeyu You, Raviv Raich, and Xiaoli Z. Fern, "Dictionary learning for bioacoustics monitoring with applications to species classification," *Journal of Signal Processing Systems*, pp. 1–15, 2016.

[20] Shai Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011.

[21] Michal Aharon and Michael Elad, "Sparse and Redundant Modeling of Image Content Using an Image-Signature-Dictionary," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 228–247, 2008.

[22] Wei Sheng Chin, Yong Zhuang, Yu Chin Juan, and Chih Jen Lin, "A learning-rate schedule for stochastic gradient methods to matrix factorization," in *Advances in Knowledge Discovery and Data Mining. 19th Pacific-Asia Conference, PAKDD 2015, Ho Chi Minh City, Vietnam, May 19-22, 2015, Proceedings, Part I*, 2015, vol. 9077, pp. 442–455.