Deep Convolutional Transform Learning

Jyoti Maggu, Angshul Majumdar, Emilie Chouzenoux, Giovanni Chierchia

Deep Convolutional Transform Learning*

Jyoti Maggu[1], Angshul Majumdar[1], Emilie Chouzenoux[2], and Giovanni Chierchia[3]

[1] Indraprastha Institute of Information Technology Delhi, India.
jyotim, angshul@iiitd.ac.in
[2] CVN, Inria Saclay, Univ. Paris-Saclay, CentraleSupélec, Gif-sur-Yvette, France.
emilie.chouzenoux@centralesupelec.fr
[3] LIGM, ESIEE Paris, Univ. Gustave Eiffel, Noisy-le-Grand, France.
giovanni.chierchia@esiee.fr

Abstract. This work introduces a new unsupervised representation learning technique called Deep Convolutional Transform Learning (DCTL). By stacking convolutional transforms, our approach is able to learn a set of independent kernels at different layers. The features extracted in an unsupervised manner can then be used to perform machine learning tasks, such as classification and clustering. The learning technique relies on a well-sounded alternating proximal minimization scheme with established convergence guarantees. Our experimental results show that the proposed DCTL technique outperforms its shallow version CTL, on several benchmark datasets.

Keywords: Transform Learning · Deep Learning · Convolutional Neural Networks · Classification · Clustering · Proximal Methods · Alternating Minimization

1 Introduction

Deep learning and more particularly convolutional neural networks (CNN) have penetrated almost every perceivable area of signal/image processing and machine learning. Its performance in traditional machine learning tasks encountered in computer vision, natural language processing and speech analysis are well assessed. CNNs are also being used with success in traditional signal processing domains, such as biomedical signal analysis [9], radars [14], astronomy [3] and inverse problems [22]. When large volumes of labeled data are available, CNNs can be trained efficiently using back-propagation methods and reach excellent performance [18]. However, training a CNN requires labeled data in a large quantity. The latter issue can be overcome by considering alternate learning paradigms, such as spiking neural network (SNN) [21] and the associated Hebbian learning [10], or alternate optimization strategies such as in [20]. However, none of those approaches can overcome the fundamental problem of neural networks, that is

* This work was supported by the CNRS-CEFIPRA project under grant NextGenBP PRC2017.

their limited capacity of learning in an unsupervised fashion. This explains the great recent interest in the machine learning community for investigating representation learning methods, that keep the best of both worlds, that is the performance of multi-layer convolutional representations and the unsupervised learning capacity [2, 19, 8, 12, 7].

In this work, we propose a deep version of the convolutional transform learning (CTL) approach introduced in [12], that we call deep convolutional transform learning (DCTL). A proximal alternating minimization scheme allows us to learn multiple layers of convolutional filters in an unsupervised fashion. Numerical experiments illustrate the ability of the method to learn representative features that lead to great performance on a set of classification and clustering problems.

The rest of the paper is organized into several sections. Section 2 introduces the transform learning paradigm and briefly reminds our previous CTL approach. The proposed DCTL formulation and the associated learning strategy are presented in Section 3. The experimental results are described in Section 4. The conclusion of this work is drawn in Section 5.

2 Transform learning

2.1 The transform learning paradigm

Traditional machine learning methods are limited in their ability to work with raw data. To perform any machine learning task, careful feature engineering is used, which in turn requires domain expertise. Using domain knowledge, a feature extractor is built to transform the raw data into a suitable internal representation. This internal representation is then fed to a learning subsystem, often a classifier to detect a pattern in the input data. Learning the weights between input and representation layer is a challenging task since both weights and output are unknown. This is called representation learning.

Transform learning, introduced in [17, 16], is a representation learning paradigm that can be viewed as the analysis equivalent of dictionary learning. In dictionary learning, a basis is learned such that it synthesizes the data from the learned coefficients. Transform learning analyzes the data by learning a basis to produce the coefficients. Mathematically this is expressed as $TX \approx Z$, where T is the analysis transform, X is the data, and Z the corresponding coefficient matrix. As proposed in [15], the matrices T and Z could be estimated by solving the following optimization problem

$$\underset{T,Z}{\text{minimize}} \ \frac{1}{2}\|TX - Z\|_F^2 + \lambda(\|T\|_F^2 - \log\det T) + \beta\|Z\|_1. \tag{1}$$

The logarithmic determinant (log det) term aims at imposing a full rank on the learned transform, and at preventing the degenerate solution $T = 0, Z = 0$. The additional quadratic penalty allows to limit scale indeterminacy. Both these additional penalties improve the conditioning of learnt transforms. Finally, the ℓ_1 term enforces a sparsity constraint on the coefficients. It is worthy to notice

that transform learning is more general than dictionary learning in its notion of compressibility. The learning process is also faster because the sparse coding step reads here as a simple step of thresholding, while in dictionary learning, it requires the resolution of a non trivial optimization problem.

2.2 Convolutional transform learning

We proposed in [12] the CTL approach, where a set of independent convolution filters are learnt to produce some data representations, in an unsupervised manner. The CTL strategy aims at generating unique and near to orthogonal filters, which in turn produces good features to be used for solving machine learning problems, as we illustrated in our experiments [12]. We present here a brief description of this approach, as its notation and concepts will serve as a basis for the deep CTL formulation introduced in this paper.

We consider a dataset $\left\{x^{(m)}\right\}_{1\leqslant m\leqslant M}$ with M entries in $\mathbb{R}^N$. The CTL formulation relies on the key assumption that the representation matrix T gathers a set of K kernels $t_1, \ldots, t_K$ with K entries, namely

$$T = [t_1 \mid \ldots \mid t_K] \in \mathbb{R}^{K\times K}. \tag{2}$$

This leads to a linear transform applied to the data to produce some features

$$(\forall m \in \{1, \ldots, M\}) \qquad Z_m \approx X^{(m)}T, \tag{3}$$

where $X^{(m)} \in \mathbb{R}^{N\times K}$ are Toeplitz matrices associated to $(x^{(m)})_{1\leqslant m\leqslant M}$ such that

$$\begin{aligned} X^{(m)}T &= \left[X^{(m)}t_1 \mid \ldots \mid X^{(m)}t_K\right] \\ &= \left[t_1 * x^{(m)} \mid \ldots \mid t_K * x^{(m)}\right], \end{aligned} \tag{4}$$

and $*$ is a discrete convolution operator with suitable padding. Let us denote

$$Z = \begin{bmatrix} Z_1 \\ \vdots \\ Z_M \end{bmatrix} \in \mathbb{R}^{NM\times K}. \tag{5}$$

The goal is then to estimate (T, Z) from $\left\{x^{(m)}\right\}_{1\leqslant m\leqslant M}$. To do so, we proposed in [12] a penalized formulation of the problem, introducing suitable conditioning constraints on the transforms, and sparsity constraint on the coefficients. The learning of (T, Z) was then performed using an alternating minimization scheme with sounded convergence guarantees. The aim of the present paper is to introduce a multi-layer formulation of the CTL, in order to learn deeper representations, with the aim of improving the representation power of the features.

3 Proposed Approach

3.1 Deep convolutional transform model

Starting from the CTL model, we propose to stack several layers of it to obtain a deep architecture. For every $\ell \in \{1, \ldots, L\}$, we will seek for the transform matrix

$$T_\ell = [t_{1,\ell} \mid \ldots \mid t_{K,\ell}] \in \mathbb{R}^{K\times K}, \tag{6}$$

where $t_{k,\ell} \in \mathbb{R}^K$ is the k-th kernel on the ℓ-th layer of the representation. The associated coefficients will be denoted as

$$Z_\ell = \begin{bmatrix} Z_{1,\ell} \\ \vdots \\ Z_{M,\ell} \end{bmatrix} \in \mathbb{R}^{NM \times K}, \tag{7}$$

with

$$(\forall m \in \{1, \dots, M\}) \qquad Z_{m,\ell} = \left[z_1^{(m,\ell)} \mid \dots \mid z_K^{(m,\ell)} \right] \in \mathbb{R}^{N \times K}. \tag{8}$$

The learning of $(T_\ell)_{1 \leqslant \ell \leqslant L}$ and $(Z_\ell)_{1 \leqslant \ell \leqslant L}$ will be performed by solving

$$\underset{(T_\ell)_{1 \leqslant \ell \leqslant L}, (Z_\ell)_{1 \leqslant \ell \leqslant L}}{\text{minimize}} \quad F(T_1, \dots, T_L, Z_1, \dots, Z_L) \tag{9}$$

where

$$F(T_1, \dots, T_L, Z_1, \dots, Z_L) = \sum_{\ell=1}^{L} \left(\frac{1}{2} \sum_{m=1}^{M} ||\mathcal{Z}_{m,\ell-1} T_\ell - Z_{m,\ell}||_F^2 + \mu ||T_\ell||_F^2 \right.$$
$$\left. - \lambda \log\det(T_\ell) + \beta ||Z_\ell||_1 + \iota_+(Z_\ell) \right), \tag{10}$$

Here, we denote ι_+ the indicator function of the positive orthant, equals to 0 if all entries of its input have non negative elements, and $+\infty$ otherwise. Moreover, by a slight abuse of notation, we denote as $\log\det$ the sum of logarithms of the singular values of a squared matrix, taking infinity value as soon as one of those is non positive. The first layer follows the CTL strategy, that is $\mathcal{Z}_{m,0} \equiv X^{(m)}$. Moreover, for every $\ell \in \{2, \dots, L\}$, we introduced the linear operator $\mathcal{Z}_{m,\ell-1}$ so as to obtain the compact notation for the multi-channel convolution product:

$$\mathcal{Z}_{m,\ell-1} T_\ell = [\mathcal{Z}_{m,\ell-1} t_{1,\ell} | \dots | \mathcal{Z}_{m,\ell-1} t_{K,\ell}] \tag{11}$$
$$= \left[t_{1,\ell} * z_1^{(m,\ell-1)} | \dots | t_{K,\ell} * z_K^{(m,\ell-1)} \right]. \tag{12}$$

3.2 Minimization algorithm

Problem (9) is non-convex. However it presents a particular multi-convex structure, that allows us to make use of an alternating proximal minimization algorithm to solve it [1, 4]. The proximity operator [5] of a proper, lower semi-continuous, convex function $\psi : \mathcal{H} \mapsto]-\infty, +\infty]$, with $(\mathcal{H}, \|\cdot\|)$ a normed Hilbert space, is defined as[4]

$$(\forall \widetilde{X} \in \mathcal{H}) \quad \text{prox}_\psi(\widetilde{X}) = \underset{X \in \mathcal{H}}{\text{argmin}} \ \psi(X) + \frac{1}{2} \|X - \widetilde{X}\|^2. \tag{13}$$

The alternating proximal minimization algorithm then consists in performing iteratively proximity updates, on the transform matrix, and on the coefficients.

[4] See also http://proximity-operator.net/

The iterates are guaranteed to ensure the monotonical decrease of the loss function F. Convergence to a local minimizer of F can also be ensured, under mild technical assumptions. The algorithm reads as follows:

$$\begin{array}{l} \text{For} \quad i = 0, 1, \dots \\ \left| \begin{array}{l} \text{For} \quad \ell = 1, \dots, L \\ \left| \begin{array}{l} T_\ell^{[i+1]} = \operatorname{prox}_{\gamma_1 F(T_1^{[i+1]}, \dots, T_L^{[i]}, Z_1^{[i+1]}, \dots, Z_L^{[i]})} \left(T_\ell^{[i]} \right) \\ Z_\ell^{[i+1]} = \operatorname{prox}_{\gamma_2 F(T_1^{[i+1]}, \dots, T_L^{[i]}, Z_1^{[i+1]}, \dots, Z_L^{[i]})} \left(Z_\ell^{[i]} \right) \end{array} \right. \end{array} \right. \end{array} \tag{14}$$

with $T_\ell^{[0]} \in \mathbb{R}^{K \times K}$, $Z_\ell^{[0]} \in \mathbb{R}^{NM \times K}$, and γ_1 and γ_2 some positive constants. We provide hereafter the expression of the proximity operators involved in the algorithm, whose proof are provided in the appendix.

Update of the transform matrix: Let $i \in \mathbb{N}$ and $\ell \in \{1, \dots, L\}$. Then

$$\begin{aligned} T_\ell^{[i+1]} &= \operatorname{prox}_{\gamma_1 F(T_1^{[i+1]}, \dots, T_L^{[i]}, Z_1^{[i+1]}, \dots, Z_L^{[i]})} \left(T_\ell^{[i]} \right), \\ &= \underset{T_\ell \in \mathbb{R}^{K \times K}}{\operatorname{argmin}} \ \frac{1}{2\gamma_1} ||T_\ell - T_\ell^{[i]}||_F^2 \\ &\qquad + \frac{1}{2} \sum_{m=1}^{M} ||\mathcal{Z}_{m,\ell-1}^{[i+1]} T_\ell - Z_{m,\ell}^{[i]}||_F^2 + \mu ||T_\ell||_F^2 - \lambda \log\det(T_\ell) \\ &= \frac{1}{2} \Lambda^{-1} V \left(\Sigma + \left(\Sigma^2 + 2\lambda \mathrm{Id} \right)^{1/2} \right) U^\top, \end{aligned} \tag{15}$$

with

$$\Lambda^\top \Lambda = \sum_{m=1}^{M} (\mathcal{Z}_{m,\ell-1}^{[i+1]})^\top (\mathcal{Z}_{m,\ell-1}^{[i+1]}) + (\gamma_1^{-1} + 2\mu)\mathrm{Id}. \tag{16}$$

Hereabove, we considered the singular value decomposition:

$$U \Sigma V^\top = \left(\sum_{m=1}^{M} (Z_{m,\ell}^{[i]})^\top (\mathcal{Z}_{m,\ell-1}^{[i+1]}) + \gamma_1^{-1} T_\ell^{[i]} \right) \Lambda^{-1}. \tag{17}$$

Update of the coefficient matrix: Let $i \in \mathbb{N}$. We first consider the case when $\ell \in \{1, \dots, L-1\}$ (recall that $\mathcal{Z}_{m,0} = X^{(m)}$ when $\ell = 1$). Then

$$\begin{aligned} Z_\ell^{[i+1]} &= \operatorname{prox}_{\gamma_2 F(T_1^{[i+1]}, \dots, T_L^{[i]}, Z_1^{[i+1]}, \dots, Z_L^{[i]})} \left(Z_\ell^{[i]} \right), \\ &= \underset{Z_\ell \in \mathbb{R}^{MN \times K}}{\operatorname{argmin}} \ \frac{1}{2\gamma_2} ||Z_\ell - Z_\ell^{[i]}||_F^2 \\ &\qquad + \frac{1}{2} \sum_{m=1}^{M} ||\mathcal{Z}_{m,\ell-1}^{[i+1]} T_\ell^{[i+1]} - Z_{m,\ell}||_F^2 \\ &\qquad + \frac{1}{2} \sum_{m=1}^{M} ||\mathcal{Z}_{m,\ell} T_{\ell+1}^{[i+1]} - Z_{m,\ell+1}^{[i]}||_F^2 \\ &\qquad + \beta ||Z_\ell||_1 + \iota_+(Z_\ell). \end{aligned} \tag{18}$$

Although the above minimization does not have a closed-form expression, it can be efficiently carried out with the projected Newton method. In the case when $\ell = L$, the second term is dropped, yielding

$$\begin{aligned} Z_L^{[i+1]} &= \text{prox}_{\gamma_2 F(T_1^{[i+1]},\ldots,T_L^{[i+1]},Z_1^{[i+1]},\ldots,Z_{L-1}^{[i+1]},\cdot)}\left(Z_L^{[i]}\right) \\ &= \underset{Z_L \in \mathbb{R}^{MN\times K}}{\text{argmin}} \ \frac{1}{2\gamma_2}\|Z_L - Z_L^{[i]}\|_F^2 \\ &\qquad + \frac{1}{2}\sum_{m=1}^{M} \|\mathcal{Z}_{m,L-1}^{[i+1]} T_L^{[i+1]} - Z_{m,L}\|_F^2 + \beta\|Z_L\|_1 + \iota_+(Z_L). \end{aligned} \tag{19}$$

Hereagain, the projected Newton method can be employed for the minimization.

4 Numerical results

4.1 Datasets

To assess the performance of the proposed approach, we considered the following image datasets[5] of small-to-medium size.

1. *YALE [6]:* The Yale dataset contains 165 images of 15 individuals, downscaled to 32-by-32 pixels. There are 11 images per subject, one per different facial expression or configuration. For our experiments, we shuffled all the samples, and took 70% for training and 30% for testing. Moreover, we full-size YALE images of size 150-by-150 pixels.
2. *E-YALE-B [11]:* The extended Yale B database contains 2432 images with 38 subjects under 64 illumination conditions. Each image is cropped to 192-by-168 pixels and downscaled to 48-by-42 pixels. For our experiments, we shuffled all the samples, took 70% for training and 30% for testing.
3. *AR-Face [13]:* This database contains more than 4000 images of 126 different subjects (70 male and 56 female). The images have various facial expressions, the lighting varies, and some of the images are partially occluded by sunglasses and scarves. For our experiments, we selected 2600 images of 100 individuals (50 males and 50 females), that is 26 different images for each subject. Train set contains 2000 images, and 600 images are kept in the test set. Each image has 540 features.

4.2 Numerical results

We experiment on the YALE, EYALEB and AR faces datasets; these are well known benchmarking face datasets. In the first set of experiments, we want to show that the accuracy of deep transform learning indeed improves when one goes deeper. Going deep beyond three layers makes performance degrade as the model tends to overfit for the small training set. To elucidate, we have used a

[5] http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html

Table 1: Accuracy on SVM with layers

Dataset	CTL	DCTL-2	DCTL-3	DCTL-4
YALE 150 × 150	94.00	94.28	**96.00**	92.21
YALE 32 × 32	88.00	89.11	**90.00**	87.73
E-YALE-B	97.38	97.00	**98.00**	94.44
AR-Faces	88.87	92.22	**97.67**	82.21

Table 2: Classification Accuracy using KNN

Dataset	Raw Features	CTL	DCTL
YALE 150 × 150	78.00	70.00	**80.00**
YALE 32 × 32	**60.00**	58.85	**60.00**
E-YALE-B	71.03	84.00	**85.00**
AR-Faces	55.00	**56.00**	58.00

Table 3: Classification Accuracy using SVM

Dataset	Raw Features	CTL	DCTL
YALE 150 × 150	93.00	94.00	**96.00**
YALE 32 × 32	68.00	88.00	**90.00**
E-YALE-B	93.24	97.38	**98.00**
AR-Faces	87.33	88.87	**97.67**

Table 4: Convolutional Transformed Clustering: ARI

YALEB/Method	Raw Features	DCTL-2	DCTL-3
K-means	0.785	0.734	**0.788**
Random	0.733	0.718	**0.738**
PCA-based	0.734	**0.791**	0.777

Table 5: Clustering time in sec

YALEB/Method	Raw Features	DCTL-2	DCTL-3
K-means	2.28	0.45	**0.14**
Random	1.95	0.33	**0.08**
PCA-based	0.36	0.09	**0.03**

simple support vector machine (SVM) classifier. The results are shown in Table 1 for levels 1, 2, 3 and 4. It has already been shown in [12] that the single layer CTL yielded better results than other single layer representation learning tools, including dictionary learning and transform Learning. Therefore it is expected that by going deeper, we will improve upon their deeper counterparts. We do not repeat those baseline experiments here, by lack of space. We also skip comparison with CNNs because of its supervised nature, whereas the proposed technique is unsupervised. We only show comparison of our proposed technique with raw features and with CTL. We take extracted features from the proposed DCTL and perform classification using two classifiers, namely KNN and SVM. The classification accuracy is reported in table 2 and table 3. Then we perform clustering on the extracted features of DCTL and report the comparison of Adjusted Rank Index (ARI) in table 4. We also report clustering time on extracted features in table 5. It is worthy to remark that the time to cluster extracted features from the proposed methodology is comparatively less than others.

5 Conclusion

This paper introduces a deep representation learning technique, named Deep Convolutional Transform Learning. Numerical comparisons are performed with the shallow convolutional transform learning formulations on image classification and clustering tasks. In the future, we plan to compare with several other deep representation learning techniques, namely stacked autoencoder and its convolutional version, restricted Boltzmann machine and its convolutional version, discriminative variants of deep dictionary and transform Learning.

6 Appendix: Proofs of the proximity updates

6.1 Update of $\boldsymbol{T}$

Let us consider $M = 1$ for simplicity, but note that all the calculations hold for M greater than 1. We want to minimize a function of the form:

$$\Phi(T) = \frac{1}{2}||XT - Z||_F^2 + \mu||T||_F^2 - \lambda \log\det(T) + \frac{1}{2\gamma_1}||T - T^{[n]}||_F^2. \tag{20}$$

Using some linear algebra, We can easily prove that:

$$\Phi(T) = \frac{1}{2}||W^{1/2}T - Y||_F^2 - \lambda \log\det(T) + c \tag{21}$$

with c a constant with respect to T,

$$W = X^\top X + (2\mu + \frac{1}{\gamma_1})\mathrm{Id} \tag{22}$$

and

$$Y = W^{-1/2}(Z^\top X + \frac{1}{\gamma_1}T^{[n]}). \tag{23}$$

Since W is invertible, one can perform the change of variable $\tilde{T} = W^{1/2}T$, that is $T = W^{-1/2}\tilde{T}$. Thus,

$$\operatorname{argmin}_T \Phi(T) = W^{-1/2} \operatorname{argmin}_T \tilde{T}\Phi(W^{-1/2}\tilde{T}), \tag{24}$$

with

$$\Phi(W^{-1/2}\tilde{T}) = \frac{1}{2}||\tilde{T} - Y||_F^2 - \lambda \log\det(W^{-1/2}\tilde{T}) + c. \tag{25}$$

Moreover,

$$\log\det(W^{-1/2}\tilde{T}) = \log\det(\tilde{T}). \tag{26}$$

Thus,

$$\operatorname{argmin}_{\tilde{T}} \Phi(W^{-1/2}\tilde{T}) = \operatorname{prox}_{\lambda \log\det(\tilde{T})}(Y), \tag{27}$$

which maps with the proximity operator of the logarithmic determinant function with weight λ. We can then apply [5, Example 24.66] and [5, Proposition 24.68] to conclude the proof.

6.2 Update of Z

We have to solve:

$$\operatorname{argmin}_Z \frac{1}{2} \sum_{m=1}^{M} \|X^{(m)}T^{[n+1]} - Z_m\|_F^2 + \beta||Z||_1 + \iota_+(Z) + \frac{1}{2\gamma_2}||Z - Z^{[n]}||_F^2. \tag{28}$$

The function in (28) is fully separable, i.e. it can be written as a sum over all the entries of matrix Z. Due to the separability property of the proximity operator, it is sufficient to resonate on the minimization of scalar function with respect to $Z_{p,q,r}$:

$$\frac{1}{2}([X^m T^{i+1}]_{p,q} - Z_{p,q,r})^2 + \beta|Z_{p,q,r}| + \iota_+(Z_{p,q,r}) + \frac{1}{2}\gamma_2(Z_{p,q,r} - Z^i_{p,q,r})^2. \tag{29}$$

One can conclude, noticing that the term $\beta|\cdot| + \iota_+$ corresponds to case 'ix' of in [5, Table 10.2], and by applying case 'iv' of [5, table 10.1] to process the final quadratic term.

References

1. Attouch, H., Bolte, J., Svaiter, B.F.: Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. Mathematical Programming **137**, 91–129 (Feb 2011)
2. Chabiron, O., Malgouyres, F., Tourneret, J.: Toward fast transform learning. International Journal on Computer Vision (114), 195–216 (2015)
3. Chan, M.C., Stott, J.P.: Deep-cee i: fishing for galaxy clusters with deep neural nets. Monthly Notices of the Royal Astronomical Society **490**(4), 5770–5787 (2019)

4. Chouzenoux, E., Pesquet, J.C., Repetti, A.: A block coordinate variable metric forward-backward algorithm. Journal on Global Optimization **66**(3), 457–485 (2016)
5. Combettes, P.L., Pesquet, J.C.: Proximal splitting methods in signal processing. In: Fixed-Point Algorithms for Inverse Problems in Science and Engineering, pp. 185–212. Springer-Verlag, New York (2010)
6. D.J.: The yale face database. URL: http://cvc. yale. edu/projects/yalefaces/yalefaces. html **1**(2), 4 (1997)
7. El Gheche, M., Chierchia, G., Frossard, P.: Multilayer network data clustering. IEEE Transactions on Signal and Information Processing over Networks **6**(1), 13–23 (Dec 2020)
8. Fagot, D., Wendt, H., Févotte, C., Smaragdis, P.: Majorization-minimization algorithms for convolutive NMF with the beta-divergence. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019). pp. 8202–8206 (2019)
9. Hannun, A.Y., Rajpurkar, P., Haghpanahi, M., Tison, G.H., Bourn, C., Turakhia, M.P., Ng, A.Y.: Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nature medicine **25**(1), 65 (2019)
10. Kempter, R., Gerstner, W., Van Hemmen, J.L.: Hebbian learning and spiking neurons. Physical Review E **59**(4), 4498 (1999)
11. Lee, K., Ho, J., Kriegman, D.: Acquiring linear subspaces for face recognition under variable lighting. IEEE Transactions on Pattern Analysis & Machine Intelligence (5), 684–698 (2005)
12. Maggu, J., Chouzenoux, E., Chierchia, G., Majumdar, A.: Convolutional transform learning. In: Proceedings of the International Conference on Neural Information Processing (ICONIP 2018). pp. 162–174. Springer (2018)
13. Martinez, A.M.: The ar face database. CVC Technical Report24 (1998)
14. Mason, E., Yonel, B., Yazici, B.: Deep learning for radar. In: 2017 IEEE Radar Conference (RadarConf). pp. 1703–1708. IEEE (2017)
15. Ravishankar, S., Bresler, Y.: Learning sparsifying transforms. IEEE Trans. Signal Process. **61**(5), 1072–1086 (2013)
16. Ravishankar, S., Bresler, Y.: Online sparsifying transform learning - Part II. IEEE J. Sel. Topics Signal Process. **9**(4), 637–646 (2015)
17. Ravishankar, S., Wen, B., Bresler, Y.: Online sparsifying transform learning - Part I. IEEE J. Sel. Topics Signal Process. **9**(4), 625–636 (2015)
18. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. nature **323**(6088), 533–536 (1986)
19. Tang, W., Chouzenoux, E., Pesquet, J., Krim, H.: Deep transform and metric learning network: Wedding deep dictionary learning and neural networks. Tech. rep. (2020), https://arxiv.org/pdf/2002.07898.pdf
20. Taylor, G., Burmeister, R., Xu, Z., Singh, B., Patel, A., Goldstein, T.: Training neural networks without gradients: A scalable admm approach. In: International conference on machine learning. pp. 2722–2731 (2016)
21. Van Gerven, M., Bohte, S.: Artificial neural networks as models of neural information processing. Frontiers in Computational Neuroscience **11**, 114 (2017)
22. Ye, J.C., Han, Y., Cha, E.: Deep convolutional framelets: A general deep learning framework for inverse problems. SIAM Journal on Imaging Sciences **11**(2), 991–1048 (2018)