HAL open science

Deep Transform and Metric Learning Network: Wedding Deep Dictionary Learning and Neural Network

Wen Tang, Emilie Chouzenoux, Jean-Christophe Pesquet, Hamid Krim

To cite this version:

Wen Tang, Emilie Chouzenoux, Jean-Christophe Pesquet, Hamid Krim. Deep Transform and Metric Learning Network: Wedding Deep Dictionary Learning and Neural Network. Neurocomputing, 2022, 509, pp.244-256. hal-03762444

HAL Id: hal-03762444

https://hal.science/hal-03762444v1

Submitted on 27 Aug 2022

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

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

Deep Transform and Metric Learning Network: Wedding Deep Dictionary Learning and Neural Network

Wen Tang[a,*], Emilie Chouzenoux[b], Jean-Christophe Pesquet[b] and Hamid Krim[a]

[a]Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, 27606, NC, USA
[b]Center for Visual Computing, Inria, Université Paris-Saclay, CentraleSupélec, Gif sur Yvette, 91190, Paris, France

ARTICLE INFO

Keywords:
Deep Dictionary Learning
Deep Neural Network
Metric Learning
Transform Learning
Proximal Operator
Differentiable Programming

ABSTRACT

On account of its many successes in inference tasks and imaging applications, Dictionary Learning (DL) and its related sparse optimization problems have garnered a lot of research interest. In DL area, most solutions are focused on single-layer dictionaries, whose reliance on handcrafted features achieves a somewhat limited performance. With the rapid development of deep learning, improved DL methods called Deep DL (DDL), have been recently proposed an end-to-end flexible inference solution with a much higher performance. The proposed DDL techniques have, however, also fallen short on a number of issues, namely, computational cost and the difficulties in gradient updating and initialization. While a few differential programming solutions have been proposed to speed-up the single-layer DL, none of them could ensure an efficient, scalable, and robust solution for DDL methods. To that end, we propose herein, a novel differentiable programming approach, which yields an efficient, competitive and reliable DDL solution. The novel DDL method jointly learns deep transforms and deep metrics, where each DL layer is theoretically reformulated as a combination of one linear layer and a Recurrent Neural Network (RNN). The RNN is also shown to flexibly account for the layer-associated approximation together with a learnable metric. Additionally, our proposed work unveils new insights into Neural Network (NN) and DDL, bridging the combinations of linear and RNN layers with DDL methods. Extensive experiments on image classification problems are carried out to demonstrate that the proposed method can not only outperform existing DDL several counts including, efficiency, scaling and discrimination, but also achieve better accuracy and increased robustness against adversarial perturbations than CNNs.

1. Introduction

Dictionary Learning / Sparse Coding has demonstrated its high potential in exploring the semantic information embedded in high dimensional noisy data. It has been successfully applied for solving different inference tasks, such as image denoising Elad and Aharon (2006), image restoration Xu, Jia, Pickering and Plaza (2016), image super-resolution Zhong (2012); Skau, Wohlberg, Krim and Dai (2016), audio processing Grosse, Raina, Kwong and Ng (2007) and image classification Zhang, Liu, Zhang, Zhang, Wang and Jing (2016).

Dictionary Learning can be decomposed into two principal themes of learning: Synthesis Dictionary Learning (SDL) and Analysis Dictionary Learning (ADL) / Transform Learning. While SDL has been greatly investigated and widely used Mairal, Ponce, Sapiro, Zisserman and Bach (2009); Aharon, Elad and Bruckstein (2006); Ramirez, Sprechmann and Sapiro (2010a); Yang, Zhang, Feng and Zhang (2014); Wang, Yang, Nasrabadi and Huang (2013), the ADL /Transform Learning, as a dual problem, has been getting greater attention for its robustness property among others Rubinstein, Peleg and Elad (2013); Bian, Krim, Bronstein and Dai (2016); Tang, Otero, Krim and Dai (2016). Conventional DL based methods have primarily focused on learning a single-layer dictionary and its associated sparse representation. Other variations on the classification theme have also been appearing with a goal of addressing some recognized limitations, such as task-driven dictionary learning Mairal et al. (2009), first introduced to jointly learn the dictionary, its sparse representation, and its classification objective. In Aharon et al. (2006), a label consistent term is additionally considered. Class-specific dictionary learning has been recently shown to improve the discrimination in Ramirez et al. (2010a); Yang et al. (2014); Wang et al. (2013) at the expense of a higher complexity. On the ADL side, more and more efficient classifiers Guo, Guo, Kong, Zhang and He (2016); Wang, Guo, Guo, Luo and Kong (2017); Wang, Guo, Guo and Kong (2018); Tang, Panahi, Krim and Dai (2018, 2019a) have resulted from numerous research efforts, and have yielded to an outperformance of SDL in both training and testing phases Tang, Panahi, Krim and Dai (2019b). All the aforementioned SDL and ADL methods are, however, single-layer-based. They thus possibly have limited learning capacity to address the complex and diverse data in an end-to-end manner. With the fast development of deep learning, Deep Dictionary Learning (DDL) methods Tariyal, Majumdar, Singh and Vatsa (2016); Mahdizadehaghdam, Dai, Krim, Skau and Wang (2017) have thus come into play to achieve the best performance in various image tasks, such as super-resolution Huang and Dragotti (2018), image classification Mahdizadehaghdam, Panahi, Krim and Dai (2019) and unsupervised learning Maggu and Majumdar (2018); Gupta, Maggu, Majumdar, Chouzenoux and Chierchia (2020).

*Corresponding author
wtang6@ncsu.edu (W. Tang); emilie.chouzenoux@centralesupelec.fr (E. Chouzenoux); jean-christophe.pesquet@centralesupelec.fr (J. Pesquet); ahk@ncsu.edu (H. Krim)
ORCID(s): 0000-0003-0283-5493 (W. Tang)

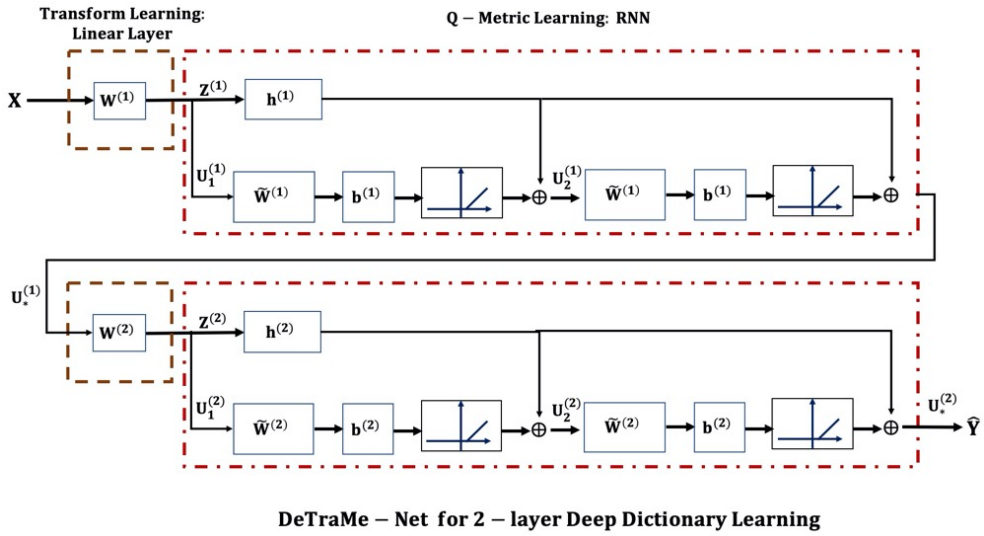


Figure 1: 2-layer DeTraMe-Net model. Each layer solves a DL problem, which is transformed into the combination of Transforming Learning (*i.e.*, linear layer in brown dashed lines) and Q-Metric Learning (*i.e.*, RNN in red dashed dot lines). A truncated 2-iterations RNN is unfolded. Sparsity is imposed by shifted-ReLU functions. In the forward pass, we first use a linear layer to learn the new representation $\mathbf{Z}^{(1)}$ for input data $\mathbf{X}$. The RNN is then used to iteratively learn the optimal sparse representation $\mathbf{U}_*^{(1)}$. For the second layer, the sparse representation $\mathbf{U}_*^{(1)}$ is used as input to learn the second layer sparse representation $\mathbf{U}_*^{(2)}$. Finally, a cross-entropy loss based on $\mathbf{U}_*^{(2)}$ and ground truth $\mathbf{Y}$ is used. The parameters $\mathbf{W}^{(i)}$, $\widetilde{\mathbf{W}}^{(i)}$, $\mathbf{h}^{(i)}$ and $\mathbf{b}^{(i)}$, $i = 1, 2$, in the linear layer and RNN parts are learned by back-propagation.

The conventional single-layer DL methods with their associated sparse representations, present significant computational challenges addressed by different techniques, including K-SVD Aharon et al. (2006); Rubinstein et al. (2013), SNS-ADL Bian et al. (2016) and Fast Iterative Shrinkage-thresholding Algorithm (FISTA) Beck and Teboulle (2009). Meant to provide a practically faster solution, the alternating minimization of FISTA still exhibited limitations and a relatively high computational cost. Apparently, the DDL training procedure is much more difficult than that of single-layer DL. In addition to the extensive time cost, DDL training also suffers from its initialization sensitivity and gradient propagation issues. All these existing DDL models have, to the best of our knowledge, fallen short on delivering fast, robust and viable solutions.

To address such a computational difficulty of conventional DL methods, differentiable programming solutions Gregor and LeCun (2010); Zhou, Di, Du, Peng, Yang, Pan, Tsang, Liu, Qin and Goh (2018) have thus been developed. They take advantage of the efficiency of neural networks to reduce the learning time. For example, LISTA Gregor and LeCun (2010) was first proposed to unfold iterative hard-thresholding into an RNN format, thus speeding up SDL. Sparse LSTM (SLSTM) Zhou et al. (2018) adapts LISTA to a Long Short Term Memory (LSTM) structure to automatically learn the dimension of the sparse representation. Although these approaches successfully resolve the computational limitation of the single-layer DL methods, none of them offers a fast, scalable and reliable solution for DDL methods.

We thus propose herein, such an efficient end-to-end DDL solution, with an improved discriminative capability. Specifically, we propose a novel differentiable programming method for DDL, which jointly learns deep metrics together with associated deep transforms. Cascading these canonical paired structures of metric and transform learning allows us to build a deep network with a stronger learning capacity, and yields what we refer to as a Deep Transform and Metric Learning Network (DeTraMe-Net). This newly proposed approach not only increases the discrimination capabilities of both single-layer DL and DDL, but also affords a flexibility of constructing different DDL or Deep Neural Network (DNN) architectures, as well as ensuring robustness against adversarial attacks.

In this paper, we first theoretically reformulate DDL as a deep cascaded paired structures of metric and transform learning. We subsequently show that a differentiable programming training solution can be realized through a combination of linear layers and RNNs in the neural network architecture. We also introduce additional flexibility and improve the power of DDL by decoupling the metric and the dual frame operator (pseudo-inverse of dictionary) into two independent variables. Our experiments demonstrate that these reformulations not only improve the discriminative ability of the single-layer DL, but also ensure that our proposed DeTraMe-Net is an efficient, discriminative, scalable and robust DDL solution. Additionally, the reformulations also yield independence between the linear and RNN layers, and thus, induce more flexibility when constructing different network architectures. As a result, different new DeTraMe networks are obtained by integrating the RNN part into various CNN architectures such as Plain CNN Springenberg, Dosovitskiy, Brox and Riedmiller (2014) and ResNets He, Zhang, Ren and Sun

(2016). In our experiments, the resulting DeTraMe-Nets-based architectures are also demonstrated to be more discriminative and robust than original CNN models.

To further simplify and clarify our discussion, we first show a two-layer DeTreMe-Net in Figure 1. In each layer of DeTraMe-Net, the DL problem is decomposed into a transform learning component, *i.e.* a linear layer stage cascaded with a nonlinear component as a learned metric. The latter, referred to as Q-Metric Learning, is realized by an RNN. The learnable Q-Metric stage may also be viewed as a non-separable weighted activation function for better feature representations. The DeTraMe-Net training is then hence similar to other neural network with the forward and backward passes.

Consequently, our main contributions are summarized below:

- We theoretically transform one-layer dictionary learning into transform learning and Q-Metric learning, and deduce how to convert DDL into DeTraMe-Net.
- Such joint transform learning and Q-Metric learning are successfully and easily implemented as a tandem of a linear layer and an RNN. To the best of our knowledge, this is the first work which makes an insightful bridge between DDL methods and the combination of linear layers and RNNs, with the associated performance gains.
- The transform and Q-Metric learning uses two independent variables, one for the dictionary and the other for the dual frame operator of the dictionary. This bridges the current work to conventional SDL while introducing more discriminative power, and allowing the use of faster learning procedures than the original DL.
- The Q-Metric can also be viewed as a parametric non-separable nonlinear activation function, while in current neural network architectures, very few non-separable nonlinear operators are used (softmax, max pooling, average pooling). As a component of a neural network, it can be flexibly inserted into any network architecture to easily construct a DL layer.
- The proposed DeTraMe-Net is demonstrated to not only greatly reduce the training and testing time, but also improve the discrimination power of DDL and its scalability.
- By easily integrating the Q-Metric in any CNN architecture, the resulting DeTraMe-Net-based CNN architectures achieve better accuracy than the original CNNs with stronger robustness.

The paper is organized as follows: We first highlight the difference between our work and the existing literature in Section 2. Then, in Section 3, we introduce the required background material. We derive the theoretical basis for our novel approach in Section 4. Its algorithmic solution is investigated in Section 5. Substantiating experimental results and evaluations are presented in Section 6. Finally, we provide some concluding remarks in Section 7.

2. Related Work

2.1. Deep Dictionary Learning

To improve the performance of conventional DL methods, deep DL methods Tariyal et al. (2016); Mahdizadehaghdam et al. (2017); Huang and Dragotti (2018); Mahdizadehaghdam et al. (2019); Maggu and Majumdar (2018); Gupta et al. (2020) have been increasing research attention. Tariyal et al. (2016); Mahdizadehaghdam et al. (2017) first develop the early multiple-layers DL methods, such as 2- or 3-layers, but they are not as deep as the later developed methods Huang and Dragotti (2018); Mahdizadehaghdam et al. (2019); Maggu and Majumdar (2018); Gupta et al. (2020). In Huang and Dragotti (2018), a deep model for ADL followed by a SDL is developed for image super-resolution. Also, Mahdizadehaghdam et al. (2019) deeply stacks SDLs to classify images to achieve promising and robust results. Moreover, unsupervised DDL approaches have also been proposed with promising results in Maggu and Majumdar (2018); Gupta et al. (2020). Our proposed DeTraMe-Net is a deep SDL supervised learning method. The merit of DeTraMe-Net is multifold, starting with the adoption of differentiable programming for a neural network framework, with other solutions Huang and Dragotti (2018); Mahdizadehaghdam et al. (2019); Maggu and Majumdar (2018); Gupta et al. (2020) relying on the alternating optimization of various variables. Specifically in contrast to the SDL-based classifier proposed in Mahdizadehaghdam et al. (2019), DeTraMe-Net further decouples the dependency between the dictionaries and their pseudo-inverse, while Mahdizadehaghdam et al. (2019) updates its dictionaries as a composite function of the dictionaries and their associated pseudo-inverses. It is also worth noting that the layer reaches 23 with no significant gain in performance beyond that in Mahdizadehaghdam et al. (2019), while our architecture can reach up to 110 layers.

2.2. Differentiable Programming for DL

Differentiable programming is a numerical optimization solution using automatic differentiation to address difficult and large scale computational problems. In general, the training of neural networks is always addressed by differentiable programming due to its complex computational graph. In the DL area, an additional difficulty is to handle sparsity measures. FISTA Beck and Teboulle (2009) is a popular iterative solution for classical optimization but it is notorious for its computational time complexity. To that end, LISTA Gregor and LeCun (2010) was the first proposed differentiable programming approach for recasting a FISTA-like solver into an RNN format. Unlike conventional solutions for solving optimization problems, LISTA uses the forward and backward passes to simultaneously update the sparse representation and dictionary in an efficient manner. SLSTM Zhou et al. (2018) learns the dimension of the sparse

representation by changing the RNN format in LISTA with a LSTM structure. Meanwhile, in Wang, Liu, Yang, Han and Huang (2015) and Liu, Chen, Chen and Wassell (2018), the authors used a CNN followed by an RNN for respectively solving super-resolution and scene recognition tasks. They directly used LISTA Gregor and LeCun (2010) in their model to jointly learn the sparse representations. Our method solves a problem similar to LISTA Gregor and LeCun (2010), in the sense that it computes both the dictionary and its sparse coefficients in a SDL context, rather than just enhancing the sparsity as Wang et al. (2015) and Liu et al. (2018) did. A single RNN format is used in LISTA Gregor and LeCun (2010). In contrast, a linear layer followed by a specific RNN is used in our method, which leads to a more discriminative DL approach than a conventional DL. Moreover, LISTA is grounded on the L_0 pseudo-norm as a sparsity measure, while ours is based on the L_1 norm. In addition, we also formally derive the linear and RNN-based layer structure from DDL, thus providing a theoretical justification and a rationale for such approaches. This may also open an avenue to new and more creative and performing alternatives.

In Hasannasab, Hertrich, Neumayer, Plonka, Setzer and Steidl (2020), an L_1 norm transformation is also used in conjunction with the proximal operator Bach, Jenatton, Mairal and Obozinski (2011); Parikh and Boyd (2014); Combettes and Pesquet (2021) in a neural network framework, which is also captured by the more general variational framework developed in Combettes and Pesquet (2020). But it is worth noting that in Hasannasab et al. (2020), there is no separation of the dictionary and the pseudo-inverse into two independent variables to learn the weighted operator as we propose here.

2.3. Notation

Symbols	Descriptions
$\mathbf{A}$, $(\mathbf{a}_i)$, $(a_{i,j})$	A Matrix
$\mathbf{A}^\top$, $\mathbf{A}^{(-1)}$	The transpose and inverse of matrices
$\mathbf{I}$	The Identity Matrix
$a_{i,j}$	The ith row and j^{th} column element of a matrix $\mathbf{A}$
$\mathbf{a}$, a_i	A Vector and its ith element
$\mathcal{A}$	An Operator

3. Preliminaries

3.1. Dictionary Learning for Classification

In task-driven dictionary learning Mairal et al. (2009), the common method for single-layer dictionary learning classifier is to jointly learn the dictionary matrix $\mathbf{D}$, the sparse representation $\mathbf{a}$ of a given vector $\mathbf{x}$, and the classifier parameter $\mathbf{C}$. Let $(\mathbf{x}_j)_{1\leq j\leq N}$ be the data and $(\mathbf{y}_j)_{1\leq j\leq N}$ the associated labels. Task-driven DL can be expressed as a solution to,

$$\underset{\mathbf{D},(\mathbf{a_j})_{1\leq j\leq N},\mathbf{C}}{\operatorname{argmin}} \sum_{j=1}^{N} f(\mathbf{x}_j,\mathbf{D},\mathbf{a}_j) + g(\mathbf{x}_j,\mathbf{y}_j,\mathbf{D},\mathbf{a}_j,\mathbf{C}). \quad (1)$$

In SDL, we learn the composition of a dictionary and a sparse reconstruction in order to reconstruct or synthesize the data, hence yielding the standard formulation,

$$f(\mathbf{x},\mathbf{D},\mathbf{a}) = \frac{\mathbf{1}}{\mathbf{2}}\|\mathbf{x}-\mathbf{D}\mathbf{a}\|^{\mathbf{2}} + \lambda\|\mathbf{a}\|_{\mathbf{1}}, \quad \lambda\in(0,+\infty). \quad (2)$$

Alternatively, in ADL, we directly operate on the data using a dictionary, leading to,

$$f(\mathbf{x},\mathbf{D},\mathbf{a}) = \frac{\mathbf{1}}{\mathbf{2}}\|\mathbf{a}-\mathbf{D}\mathbf{x}\|^{\mathbf{2}} + \lambda\|\mathbf{a}\|_{\mathbf{1}}, \quad \lambda\in(0,+\infty). \quad (3)$$

The term $g(\mathbf{x},\mathbf{y},\mathbf{D},\mathbf{a},\mathbf{C})$ may correspond to various kinds of loss functions, such as least-squares, cross-entropy, or hinge loss.

3.2. Deep Dictionary Learning for Classification

An efficient DDL approach Mahdizadehaghdam et al. (2019) consists of computing

$$\hat{\mathbf{y}} = \varphi(\mathbf{C}\mathbf{x}^{(\mathbf{s})}), \quad (4)$$

where $\hat{\mathbf{y}}$ denotes the estimated label, $\mathbf{C}$ is the classifier matrix, φ is a nonlinear function, and

$$\begin{aligned}\mathbf{x}^{(s)} =& \mathcal{P}^{(s)}\circ\mathcal{M}_{\mathbf{D}^{(s)}}\circ\mathcal{P}^{(s-1)}\circ\mathcal{M}_{\mathbf{D}^{(s-1)}}\circ \\ &\ldots\circ\mathcal{P}^{(1)}\circ\mathcal{M}_{\mathbf{D}^{(1)}}(\mathbf{x}^{(0)}),\end{aligned} \quad (5)$$

where $\circ$ denotes the composition of operators. For every layer $r\in\{1,\ldots,s\}$, $\mathcal{P}^{(r)}$ is a reshaping operator, which is a tall matrix. Moreover, $\mathcal{M}_{\mathbf{D}^{(r)}}$ is a nonlinear operator computing a sparse representation within a synthesis dictionary matrix $\mathbf{D}^{(r)}$. More precisely, for a given matrix $\mathbf{D}^{(r)}\in\mathbb{R}^{m_r\times k_r}$,

$$\begin{aligned}\mathcal{M}_{\mathbf{D}^{(r)}} :& \mathbb{R}^{m_r}\to\mathbb{R}^{k_r} \\ \mathbf{x}\mapsto& \underset{\mathbf{a}\in\mathbb{R}^{k_r}}{\operatorname{argmin}} \quad \mathcal{L}^R(\mathbf{D}^{(r)},\mathbf{a},\mathbf{x}),\end{aligned} \quad (6)$$

with

$$\mathcal{L}^R(\mathbf{D}^{(r)},\mathbf{a},\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{D}^{(r)}\mathbf{a}\|_F^2 + \lambda\psi_r(\mathbf{a}) + \frac{\alpha}{2}\|\mathbf{a}\|_2^2 + (\mathbf{d}^{(r)})^\top\mathbf{a}, \quad (7)$$

where $(\lambda,\alpha)\in(0,+\infty)^2$, and ψ_r is a function in $\Gamma_0(\mathbb{R}^{k_r})$, the class of proper lower semicontinuous convex functions from $\mathbb{R}^{k_r}$ to $(-\infty,+\infty]$. $\mathbf{d}^{(r)}\in\mathbb{R}^{k_r}$ is a vector for the general linear penalty in (7) to balance fidelity and classification. For conventional DL, a simple choice consists in setting $\mathbf{d}^{(r)}$ to zero, while adopting the following specific form for ψ_r;

$$\psi_r = \|\cdot\|_1 + \iota_{[0,+\infty)^{k_r}}, \quad (8)$$

where ι_S denotes the indicator function of a set S (equal to zero in S and $+\infty$ otherwise). Note that Eq. (6) corresponds to the minimization of a strongly convex function, which thus admits a unique minimizer, so making the operator $\mathcal{M}_{\mathbf{D}^{(r)}}$ properly defined.

4. Joint Deep Metric and Transform Learning

4.1. Proximal interpretation

Our goal here is to establish an equivalent but more insightful solution for $\mathcal{M}_{\mathbf{D}}$ in each layer.
To simplify notation, we omit the superscript which denotes the layer in Eq. (6) which, in turn, aims at finding the sparse representation $\mathbf{a}$.

Claim 1: *$\mathcal{M}_{\mathbf{D}}$ can be solved by a proximal operator Bach et al. (2011); Parikh and Boyd (2014); Combettes and Pesquet (2021) related to transform learning with a metric $\mathbf{Q}$:*

$$\mathcal{M}_{\mathbf{D}}(\mathbf{x}) = \operatorname{prox}_{\lambda\psi}^{\mathbf{Q}}(\mathbf{F}\mathbf{x} - \mathbf{c}) \tag{9}$$

with

$$\mathbf{Q} = \mathbf{D}^\top\mathbf{D} + \alpha\mathbf{I},\ \mathbf{F} = \mathbf{Q}^{-1}\mathbf{D}^\top,\ \mathbf{c} = \mathbf{Q}^{-1}\mathbf{d}. \tag{10}$$

Indeed, for every $\mathbf{D} \in \mathbb{R}^{m\times k}$, $\mathbf{a} \in \mathbb{R}^k$, and $\mathbf{x} \in \mathbb{R}^m$, Eq. (7) can be re-expressed as follows:

$$\begin{aligned}\mathcal{L}^R(\mathbf{D},\mathbf{a},\mathbf{x}) &= \frac{1}{2}\big(\|\mathbf{x}\|^2 - 2\mathbf{x}^\top\mathbf{D}\mathbf{a} + \mathbf{a}^\top(\mathbf{D}^\top\mathbf{D} + \alpha\mathbf{I})\mathbf{a}\big)\\ &\quad + \lambda\psi(\mathbf{a}) + \mathbf{d}^\top\mathbf{a}\\ &= \tilde{\mathcal{L}}^R(\mathbf{D},\mathbf{a},\mathbf{x}) + \frac{1}{2}(\|\mathbf{x}\|^2 - \|\mathbf{F}\mathbf{x}\|_{\mathbf{Q}}^2 - \|\mathbf{c}\|_{\mathbf{Q}}^2)\\ &\quad + \mathbf{x}^\top\mathbf{D}\mathbf{c},\end{aligned} \tag{11}$$

where

$$\tilde{\mathcal{L}}^R(\mathbf{D},\mathbf{a},\mathbf{x}) = \frac{1}{2}\|\mathbf{a} - \mathbf{F}\mathbf{x} + \mathbf{c}\|_{\mathbf{Q}}^2 + \lambda\psi(\mathbf{a}), \tag{12}$$

and $\|\cdot\|_{\mathbf{Q}} = \sqrt{(\cdot)^\top\mathbf{Q}(\cdot)}$ denotes the weighted Euclidean norm induced by $\mathbf{Q}$. Determining the optimal sparse representation $\mathbf{a}$ of $\mathbf{x} \in \mathbb{R}^m$ is therefore, equivalent to computing the proximity operator in Eq. (12), that is Eq. (9):

$$\mathcal{M}_{\mathbf{D}}(\mathbf{x}) = \underset{\mathbf{a}\in\mathbb{R}^k}{\operatorname{argmin}}\ \tilde{\mathcal{L}}^R(\mathbf{D},\mathbf{a},\mathbf{x}) = \operatorname{prox}_{\lambda\psi}^{\mathbf{Q}}(\mathbf{F}\mathbf{x} - \mathbf{c}). \tag{13}$$

This thus establishes a re-expression of the solution of the representation procedure as the proximity operator of $\lambda\psi$ within the metric induced by the symmetric positive definite matrix $\mathbf{Q}$ Combettes and Pesquet (2010); Chouzenoux, Pesquet and Repetti (2014). Furthermore, it shows that the SDL can be equivalently viewed as an ADL formulation involving the dictionary matrix $\mathbf{F}$, provided that a proper metric is chosen.

4.2. Multilayer representation

Consequently, by substituting Eq. (13) in Eqs. (4) and (5), the DDL model can be re-expressed in a more concise and comprehensive form as

$$\begin{aligned}\hat{\mathbf{y}} =& \varphi\circ\mathcal{A}^{(s+1)}\circ\operatorname{prox}_{\lambda\psi_s}^{\mathbf{Q}^{(s)}}\circ\mathcal{A}^{(s)}\circ\operatorname{prox}_{\lambda\psi_{s-1}}^{\mathbf{Q}^{(s-1)}}\circ\dots\\ &\circ\operatorname{prox}_{\lambda\psi_1}^{\mathbf{Q}^{(1)}}\circ\mathcal{A}^{(1)}(\mathbf{x}^{(0)}),\end{aligned} \tag{14}$$

where, for $1 \le r \le s$, the affine operators $\mathcal{A}^{(r)}$ mapping $\mathbf{z}^{(r-1)} \in \mathbb{R}^{k_{r-1}}$ to $\mathbf{z}^{(r)} \in \mathbb{R}^{k_r}$ by an analysis transform $\mathbf{W}^{(r)}$ and a shift term $\mathbf{c}^{(r)}$, and explicitly as,

$$\begin{aligned}\forall r \in \{1,\dots,s\}, \mathcal{A}^{(r)} :& \mathbb{R}^{k_{r-1}} \to \mathbb{R}^{k_r}\\ & \mathbf{z}^{(r-1)} \mapsto \mathbf{W}^{(r)}\mathbf{z}^{(r)} - \mathbf{c}^{(r)}\end{aligned} \tag{15}$$

with $k_0 = m_1$ and

$$\begin{aligned}&\mathbf{W}^{(1)} = \mathbf{F}^{(1)},\\ &\forall r \in \{2,\dots,s\},\\ &\qquad \mathbf{W}^{(r)} = \mathbf{F}^{(r)}\mathcal{P}^{(r-1)},\\ &\qquad \mathbf{W}^{(s+1)} = \mathbf{C}\mathcal{P}^{(s)}\\ &\forall r \in \{1,\dots,s\},\\ &\qquad \mathbf{Q}^{(r)} = (\mathbf{D}^{(r)})^\top\mathbf{D}^{(r)} + \alpha\mathbf{I},\\ &\qquad \mathbf{F}^{(r)} = (\mathbf{Q}^{(r)})^{-1}(\mathbf{D}^{(r)})^\top,\\ &\qquad \mathbf{c}^{(r)} = (\mathbf{Q}^{(r)})^{-1}\mathbf{d}^{(r)}.\end{aligned} \tag{16}$$

Eq. (15) shows that, for each layer r, we obtain a structure similar to a linear layer by treating $\mathbf{W}^{(r)}$ as the weight operator and $\mathbf{c}^{(r)}$ as the bias parameter, which are referred as the Transform learning part in DeTraMe method. In standard Forward Neural Networks (FNNs), the activation functions can be interpreted as proximity operators of convex functions Combettes and Pesquet (2020). Eq. (14) attests that our model is more general, in the sense that different metrics are introduced for these operators. In the next section, we propose an efficient method to learn these metrics in a supervised manner.

5. Q-Metric Learning

5.1. Prox computation

Reformulation (14) has the great advantage to allow us to benefit from algorithmic frameworks developed for FNNs, provided that we are able to compute efficiently

$$\operatorname{prox}_{\lambda\psi}^{\mathbf{Q}}(\mathbf{Z}) = \underset{\mathbf{U}\in\mathbb{R}^{k\times N}}{\operatorname{argmin}}\ \frac{1}{2}\|\mathbf{U} - \mathbf{Z}\|_{F,\mathbf{Q}}^2 + \lambda\psi(\mathbf{U}), \tag{17}$$

where $\|\cdot\|_{F,\mathbf{Q}} = \sqrt{\operatorname{tr}((\cdot)\mathbf{Q}(\cdot)^\top)}$ is the $\mathbf{Q}$-weighted Frobenius norm. Hereabove, $\mathbf{Z}$ is a matrix where the N samples associated with the training set have been stacked columnwise. A similar convention is used to construct $\mathbf{X}$ and $\mathbf{Y}$ from $(\mathbf{x}_j)_{1\le j\le N}$ and $(\mathbf{y}_j)_{1\le j\le N}$.

An elastic-net like regularization is chosen by setting $\psi = \|\cdot\|_1 + \iota_{[0,+\infty)^{k\times N}} + \frac{\beta}{2\lambda}\|\cdot\|_F^2$ with $\beta \in (0,+\infty)$. We have, in particular, observed that the last quadratic term has a positive influence in increasing stability and avoiding overfitting. As $\mathbf{Q} = \mathbf{D}^\top\mathbf{D} + \alpha\mathbf{I}$ in Eq. (10), Eq. (17) is actually equivalent to solving the following optimization problem:

$$\begin{aligned}\underset{\mathbf{U}\in[0,+\infty)^{k\times N}}{\operatorname{minimize}}\ & \frac{1}{2}\|\mathbf{D}(\mathbf{U} - \mathbf{Z})\|_F^2 + \frac{\alpha}{2}\|\mathbf{U} - \mathbf{Z}\|_F^2\\ &+ \frac{\beta}{2}\|\mathbf{U}\|_F^2 + \lambda\|\mathbf{U}\|_1.\end{aligned} \tag{18}$$

Various iterative splitting methods could be used to find the unique minimizer of the above optimized convex function Boyd and Vandenberghe (2004); Komodakis and Pesquet (2014). Our purpose is to develop an algorithmic solution for which classical NN learning techniques can be applied in a fast and convenient manner. We show next the following property.

Claim 2: *The solution of Eq. (18) is obtained as an iteration of the form:*

$$\mathbf{U}_{t+1} = \mathrm{ReLU}\left((\mathbf{h}\mathbf{1}^\top) \odot \mathbf{Z} + \widetilde{\mathbf{W}}(\mathbf{U}_t - \mathbf{Z}) - \mathbf{b}\mathbf{1}^\top\right), \quad (19)$$

where $\widetilde{\mathbf{W}}$ is a symmetric $k \times k$ matrix, $\mathbf{h} \in [0,1]^k$, $\mathbf{b} \in [0,+\infty)^k$, and $\mathbf{1} = [1,\dots,1]^\top \in \mathbb{R}^N$.

By subdifferential calculus, the solution $\mathbf{U}$ to the problem (18) satisfies the following optimality condition:

$$\mathbf{0} \in \mathbf{Q}(\mathbf{U} - \mathbf{Z}) + \beta\mathbf{U} + \lambda\partial\widetilde{\psi}(\mathbf{U}), \quad (20)$$

where $\widetilde{\psi} = \|\cdot\|_1 + \iota_{[0,+\infty)^{k\times N}}$. Element-wise rewriting of Eq. (20) yields, for every $i \in \{1,\dots,k\}$, and $j \in \{1,\dots,N\}$,

$$0 \in \sum_{\ell=1}^{k} q_{i,\ell}(u_{\ell,j} - z_{\ell,j}) + \beta u_{i,j} + \begin{cases} (-\infty,\lambda] & \text{if } u_{i,j} = 0 \\ \lambda & \text{if } u_{i,j} > 0 \\ \varnothing & \text{if } u_{i,j} < 0 \end{cases}. \quad (21)$$

Let us adopt a block-coordinate approach and update the i-th row of $\mathbf{U}$ by fixing all the other ones. As $\mathbf{Q}$ is a positive definite matrix, $q_{i,i} > 0$ and Eq. (21) implies that

$$u_{i,j} = \begin{cases} \frac{q_{i,i}}{q_{i,i}+\beta} z_{i,j} - v_{i,j} & \text{if } q_{i,i} z_{i,j} > (q_{i,i}+\beta) v_{i,j} \\ 0 & \text{otherwise,} \end{cases} \quad (22)$$

where $v_{i,j} = \frac{\lambda + \sum_{\ell=1,\ell\neq i}^{k} q_{i,\ell}(u_{\ell,j} - z_{\ell,j})}{q_{i,i}+\beta}$. And let

$$\begin{aligned} \widetilde{\mathbf{W}} &= -\left(\frac{q_{i,\ell}}{q_{i,i}+\beta}(1-\delta_{i-\ell})\right)_{1\le i,\ell\le k}, \\ \mathbf{h} &= \left(\frac{q_{i,i}}{q_{i,i}+\beta}\right)_{1\le i\le k} \in [0,1]^k, \\ \mathbf{b} &= \left(\frac{\lambda}{q_{i,i}+\beta}\right)_{1\le i\le k} \in [0,+\infty)^k, \end{aligned} \quad (23)$$

where $(\delta_\ell)_{\ell\in\mathbb{Z}}$ is the Kronecker sequence (equal to 1 when $\ell = 0$ and 0 otherwise). Then, Eq. (22) suggests that the elements of $\mathbf{U}$ can be globally updated, at iteration t, as shown in Eq. (19):

$$\mathbf{U}_{t+1} = \mathrm{ReLU}\left((\mathbf{h}\mathbf{1}^\top) \odot \mathbf{Z} + \widetilde{\mathbf{W}}(\mathbf{U}_t - \mathbf{Z}) - \mathbf{b}\mathbf{1}^\top\right),$$

with $\odot$ denoting the Hadamard (element-wise) product. Note that a similar expression can be derived by applying a preconditioned forward-backward algorithm Chouzenoux et al. (2014) to Eq. (18), where the preconditioning matrix is $\mathrm{Diag}(q_{1,1},\dots,q_{k,k})$, which has been detailed in the Appendix A. The implementation of the method allowing us to compute the proximity operator in (17) is summarized below:

Algorithm 1 Q-Metric ReLU Computation

Input: matrix $\widetilde{\mathbf{W}}$ and $\mathbf{Z}$, vectors $\mathbf{h}$ and $\mathbf{b}$, and maximum iteration number $t_{\max}$
Output: Sparse Representation $\mathbf{U}^*$

```
Initialize U_0 as the null matrix and set t = 0
while not converged and t < t_max do
    Update U_{t+1} according to Eq. (19)
    t ← t + 1
end while
```

5.2. RNN implementation

Although $\widetilde{\mathbf{W}}$, $\mathbf{h}$, and $\mathbf{b}$ are defined on the basis of matrix $\mathbf{Q}$, for more flexibility, we will treat them as decoupled variables. Then, given independent $\widetilde{\mathbf{W}}$, $\mathbf{h}$, and $\mathbf{b}$, Alg. (1) can be viewed as an RNN structure for which $\mathbf{U}_t$ is the hidden variable and $\mathbf{Z}$ is a constant input over time. By taking advantage of existing gradient back-propagation techniques for RNNs, $(\widetilde{\mathbf{W}}, \mathbf{h}, \mathbf{b})$ can thus be directly computed in order to minimize the global loss $\mathcal{L}$. This shows that, thanks to the re-parameterization in Eq. (23), Q-Metric Learning has been recast as the training of a specific RNN.

Note that $\mathbf{Q}$ is a $k\times k$ symmetric matrix. In order to reduce the number of parameters and ease of optimizing them, we choose a block-diagonal structure for $\mathbf{Q}$. In addition, for each of the blocks, either an arbitrary or convolutive structure can be adopted. Since the structure of $\mathbf{Q}$ is reflected by the structure of $\widetilde{\mathbf{W}}$, this leads in Eq. (19) to fully connected or convolutional layers where the channel outputs are linked to non overlapping blocks of the inputs. In our experiments on images, Convolutional-RNNs have been preferred for practical efficiency.

5.3. Training procedure

We have finally transformed our DDL approach in an alternation of linear layers and specific RNNs. This not only simplifies the implementation of the resulting DeTraMe-Net by making use of standard NN tools, but also allows us to employ well-established stochastic gradient-based learning strategies. Let $\rho_t > 0$ be the learning rate at iteration t, the simplified form of a training method for DeTraMe-Nets is provided in Alg. 2.

The constraints on the parameters of the RNNs have been imposed by projections. In Alg. 2, $\mathsf{P}_{\mathcal{S}}$ denotes the projection onto a nonempty closed convex set $\mathcal{S}$ and $\mathcal{D}_0$ is the vector space of $k \times k$ matrices with diagonal terms equal to 0.

6. Experiments and Results

In this section, our single-layer DeTraMe-Net is evaluated on four well-known small datasets: Extended YaleB

Algorithm 2 Deep Transform and Metric Learning Network

Initialization:

for $r = 1, \ldots, s+1$ **do**
  Randomly initialize $\mathbf{W}_0^{(r)}$, $\mathbf{c}_0^{(r)}$, $\widetilde{\mathbf{W}}_0^{(r)}$, $\mathbf{h}_0^{(r)}$, and $\mathbf{b}_0^{(r)}$.
end for
Set $t = 0$.
while not converged and $t < t_{\max}$ **do**
  Forward pass:
  $\mathbf{U}_t^{(0)} = \mathbf{X}$
  for $r = 1, \ldots, s+1$ **do**
    $\mathbf{Z}_t^{(r)} = \mathbf{W}_t^{(r)}\mathbf{U}_t^{(r-1)} - \mathbf{c}_t^{(r)}$
    if $r \leq s$ **then**
      $\mathbf{U}_t^{(r)} = \mathrm{prox}_{\lambda\psi_r}^{\mathbf{Q}_t^{(r)}}(\mathbf{Z}_t^{(r)})$ by Alg. 1
    end if
  end for
  $\hat{\mathbf{Y}}_t = \varphi(\mathbf{Z}_t^{(s+1)})$
  Loss: $\mathcal{L}'(\theta_t) = \mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}_t)$, θ_t: vector of all parameters

  Backward pass:
  for $r = 1, \ldots, s+1$ **do**
    $\mathbf{W}_{t+1}^{(r)} = \mathbf{W}_t^{(r)} - \rho_t \frac{\partial \mathcal{L}'}{\partial \mathbf{W}^{(r)}}(\theta_t)$
    $\mathbf{c}_{t+1}^{(r)} = \mathbf{c}_t^{(r)} - \rho_t \frac{\partial \mathcal{L}'}{\partial \mathbf{c}^{(r)}}(\theta_t)$
  end for
  for $r = 1, \ldots, s$ **do**
    $\widetilde{\mathbf{W}}_{t+1}^{(r)} = \mathsf{P}_{\mathcal{D}_0}\left(\widetilde{\mathbf{W}}_t^{(r)} - \rho_t \frac{\partial \mathcal{L}'}{\partial \widetilde{\mathbf{W}}^{(r)}}(\theta_t)\right)$
    $\mathbf{h}_{t+1}^{(r)} = \mathsf{P}_{[0,1]^k}\left(\mathbf{h}_t^{(r)} - \rho_t \frac{\partial \mathcal{L}'}{\partial \mathbf{h}^{(r)}}(\theta_t)\right)$
    $\mathbf{b}_{t+1}^{(r)} = \mathsf{P}_{[0,+\infty)^k}\left(\mathbf{b}_t^{(r)} - \rho_t \frac{\partial \mathcal{L}'}{\partial \mathbf{b}^{(r)}}(\theta_t)\right)$
  end for
  $t \leftarrow t + 1$
end while

Georghiades, Belhumeur and Kriegman (2001), AR Martinez and Benavente (June 1998), Caltech101 Fei-Fei, Fergus and Perona (2007) and Scene15 Lazebnik, Schmid and Ponce (2006). The deep DeTraMe-Net method is evaluated on three popular datasets, namely CIFAR10 Krizhevsky, Hinton et al. (2009), CIFAR100 Krizhevsky et al. (2009) and Street View House Numbers (SVHN) Netzer, Wang, Coates, Bissacco, Wu and Ng (2011). Since the common NN architectures are plain networks such as ALL-CNN Springenberg et al. (2014) and residual ones, such as ResNet He et al. (2016) and WideResNet Zagoruyko and Komodakis (2016), we compare DeTraMe-Net with these three respective state-of-the-art architectures. All the experiments of the state-of-the-arts and our method are re-implemented and repeated over 5 runs for the large datasets, and repeated over 10 runs for the small datasets.

6.1. Architectures

Since we break SDL into two independent linear layer and RNN parts, RNNs can be flexibly inserted into any nonlinear layer of a neural network. To comprehensively compare our DeTraMe-Net and other various DL and DDL methods, we compose a single-layer DeTraMe-Net and two different deep DeTraMe-Net architectures with inserted RNNs into Plain Networks and residual blocks.

A linear forward layer followed by a linear Q-Metric learning is used to construct a single-layer DeTraMe-Net. The convolutional Q-Metric ReLU is employed in the deep DeTraMe-Nets. For the two different deep DeTraMe-Nets, one solution is to replace all the ReLU activation layers in PlainNet with Q-Metric ReLU, leading to DeTraMe-PlainNet. Another one is to replace the ReLU layer inside the block in ResNet by Q-Metric ReLU, giving rise to DeTraMe-ResNet. When replacing all the ReLU layers, DeTraMe-PlainNet becomes equivalent to DDL as explained in Section 5. When only replacing a single ReLU layer in the ResNet architecture, a new DeTraMe-ResNet structure is built. The detailed architectures are illustrated in the Appendix B.

For the PlainNet, we use a 9 layer architecture similar to ALL-CNN Springenberg et al. (2014) with dropouts, as listed in Table 1. For the ResNet architecture, we follow the setting in He et al. (2016), the first layer is a 3×3 convolutional layer with 16 filters. 3 residual blocks with output map size of 32, 16, and 8 are then used with 16, 32 and 64 filters for each block. The network ends up with a global average pooling and a fully-connected layer. The parameters listed in Table 2 are respectively chosen equal to $n = 1, 3, 9, 18, 27$ for ResNet 8, 20, 56, 110 and 164-layer networks, and we respectively use $n = 2, q = 4$ and $n = 2, q = 8$ for WideResNet 16-4 and WideResNet 16-8 networks as suggested in Zagoruyko and Komodakis (2016).

For a single-layer DeTraMe-Net, the conventional RNN is used with 3 iterations in all our experiments, as chosen by cross validation. For deep DeTraMe-Net, as CNNs settings have demonstrated good results without overfitting, and for fair comparison, we choose the exact same number of filters in CNNs as our dictionary size. Since the matrix in Q-Metric part is actually a surrogate to the pseudo-inverse of the dictionary, we set this weighted matrix in RNN of the same size as the dictionary pseudo-inverse. So, we use convolutional RNNs having the same filter size (resp. number of channels) as those in the convolutional layer before. The number of parameters of each model as well as the number of iterations performed in RNNs, are indicated in Table 6. Each iteration number shown in Table 6 is also chosen by cross validation.

6.2. Datasets and Training Settings

6.2.1. Small Datasets

Extended YaleB Georghiades et al. (2001): The Extended YaleB face dataset contains 2414 frontal face images of 38 persons under various illumination and expression conditions. Each person has about 64 images, each cropped to 168×192 pixels. In our experiment, each Extended YaleB face image is reduced to a 504-dimensional feature vector. Half of the images are randomly chosen for training, and the rest for testing. The dictionary size is set to 1216 atoms and the total number of epochs is 226. **AR** Martinez and Benavente (June 1998): The AR Face dateset has 2600 color

DeTraMe-PlainNet 3-layer	PlainNet 3-layer	PlainNet 6-layer	PlainNet 9-layer	PlainNet 12-layer
Input 32 x 32 RGB Image with dropout(0.2)				
3×3 conv 96 + Q-Metric: 3×3 conv 96	3×3 conv 96 RELU	3×3 conv 96 RELU 3×3 conv 96 RELU	3×3 conv 96 RELU 3×3 conv 96 RELU 3×3 conv 96 RELU with stride=2, dropout(0.5)	3×3 conv 96 RELU 3×3 conv 96 RELU 3×3 conv 96 RELU with stride=2, dropout(0.5) 3×3 conv 192 RELU
3×3 conv 96 with stride=2 + Q-Metric: 3×3 conv 96	3×3 conv 96 RELU with stride=2	3×3 conv 96 RELU with stride=2, dropout(0.5) 3×3 conv 192 RELU	3×3 conv 192 RELU 3×3 conv 192 RELU 3×3 conv 192 RELU with stride=2, dropout(0.5)	3×3 conv 192 RELU 3×3 conv 192 RELU with stride=2, dropout(0.5) 3×3 conv 192 RELU
3×3 conv 10 with stride=2 +Q-Metric: 3×3 conv 10	3×3 conv 10 RELU with stride=2	3×3 conv 192 RELU 3×3 conv 10 RELU with stride=2	3×3 conv 192 RELU 1×1 conv 192 RELU 1×1 conv 10 RELU	3×3 conv 192 RELU with stride=2 3×3 conv 192 RELU 1×1 conv 192 RELU 1×1 conv 10 RELU
Global Average Pooling				
Softmax				

Table 1
Model Description of PlainNet

output map size	32×32	16×16	8×8
# layers	$1+2n$	$2n$	$2n$
#filters	16	32	64
WideResNet #filters	$16\times q$	$32\times q$	$64\times q$

Table 2
ResNet Model He et al. (2016)

images of 50 females and 50 males. Each person has about 26 images of size 165×120. Each face image is projected to a 540 dimensional feature vector for its random face feature. 20 images of each person are randomly selected as a training set and the other 6 images for testing. The dictionary size is 2000 atoms and the total number of epochs is 133. **Caltech101** Fei-Fei et al. (2007): The Caltech101 dataset has 101 different categories of different objects and one non-object category. Most categories have around 50 images. A three-level segmentation Scale-Invariant Feature Transform (SIFT) is used for feature extraction. Its codebook size is 1024 and its final PCA reduction feature dimension is 3000. 30 images per class are randomly chosen as training data, and other images are used as testing data. The dictionary size is set to 3060 and the total number of epochs is 60. **Scene15** Lazebnik et al. (2006): Scene15 dataset contains a total of 15 categories of different scenes, and each category has around 200 images. A four-level spatial pyramid dense SIFT and a 200 size codebook are used here, and 3000 features are obtained. 100 images per class are randomly picked as training data, and the rest of images are used as testing data. The dictionary size is set to 1500 and the total number of epochs is 107.

Preprocessing: Each face dataset image is projected onto an n-dimensional random face feature vector. The projection is performed by a randomly generated matrix with a zero mean normal distribution whose rows are L_2 normalized. For the object or scene images, a spatial pyramid method Lazebnik et al. (2006) based on the dense SIFT features with three or four segmentation sizes, such as 1×1, 2×2, 4×4 and 8×8, is applied to capture salient information at different scales. The dense SIFT feature is computed using 16×16 patches and with a 6-pixel stride. At the same time, an associated codebook is trained by k-means clustering for spatial pyramid features. Spatial pyramid features of each subregion are then concatenated together as a vector to represent one image. Due to the sparse nature of the spatial pyramid features, PCA is finally used to reduce the feature dimensions. All the preprocessing procedures follow the same settings used in Jiang, Lin and Davis (2013).

Training Settings: The models for all small datasets are trained by SGD optimizer to minimize the MSE loss with a weight decay of 1×10^{-5} and 0 momentum on an Nvidia V100 32Gb GPU. The learning rate starts with 0.001. For AR, Caltech101 and Scene15 datasets, the learning rate is also reduced by 0.1 at the 100-th, 120-th epochs. Rather than using a whole batch for training, all datasets are trained by the mini-batches, whose size is respectively 2, 20, 12 and 15 for the Extended YaleB, AR, Caltec101 and Scene 15 dataset. No data augmentation is used for this case.

6.2.2. Big Datasets

CIFAR10 Krizhevsky et al. (2009) contains 60,000 32×32 color images divided into 10 classes. 50,000 images are used for training and 10,000 images for testing. **CIFAR100** Krizhevsky et al. (2009) is also constituted of 32×32 color images. However, it includes 100 classes with 50,000 images for training and 10,000 images for testing. **SVHN** Netzer et al. (2011) contains 630,420 color images with size 32×32. 604,388 images are used for training and 26,032 images are used for testing.

Preprocessing: For CIFAR datasets, the normalized input image is 32×32 randomly cropped after 4×4 padding on each sides of the image and random flipping, similarly to He et al. (2016); Zagoruyko and Komodakis (2016). No other data augmentation is used. For SVHN, we normalize the range of the images between 0 and 1.

Methods	Extended YaleB	AR	Caltech101	Scene15
SRC Wright, Yang, Ganesh, Sastry and Ma (2009)	96.5%*	97.5%*	70.7%*	91.8% *
CRC Zhang, Yang and Feng (2011)	**97.0%***	98.0%*	68.2%*	92.0%*
DLSI Ramirez, Sprechmann and Sapiro (2010b)	**97.0%***	97.5%*	73.1%*	91.7%*
FDDL Yang et al. (2014)	96.7%	97.5%	73.2%	92.3%
LC-KSVDJiang et al. (2013)	96.7%*	97.8%*	**73.6%***	92.9%*
DFEDLLi, Zhang, Qin, Zhang and Shao (2019)	93.8%	92.0%	73.1%	98.2%
DeTraMe-Net	96.5%	**99.2%**	73.2%	**98.5%**

Table 3
Accuracy: DeTraMe-Net vs. single-layer DL. All the results with '*' of the competing methods were reported from the original papers.

Model	# Parameters	CIFAR10	CIFAR100
PlainNet 9-layer Springenberg et al. (2014)	1.4M	90.31% ± 0.31%	66.15% ± 0.61%
DDL 9 Mahdizadehaghdam et al. (2019)	1.4M	93.04%*	68.76%*
DeTraMe-Net 9	3.0M	**93.05%** ± 0.46%	**69.68%** ± 0.50%
DeTraMe-Net 9 (Best)	3.0M	**93.40%**	**70.34%**

Table 4
Accuracy: DeTraMe-Net vs. DDL: the architectures are listed in the fourth column in Table 1. The number with '*' was reported in the original paper.

Model	#Parameters	Training (s)	Testing (s)
DDL Mahdizadehaghdam et al. (2019)	0.35 M	0.2784*	$9.40\times10^{-2*}$
DeTraMe-Net 12	**2.4 M**	**0.1605**	$\mathbf{3.52}\times10^{-4}$

Table 5
Time Complexity: DeTraMe-Net vs. DDL: The number with '*' was averaged based on the reported one in the original paper.

Training Settings: All the models are trained on an Nvidia V100 32Gb GPU with 128 mini-batch size. The models of both PlainNet and ResNet architectures are trained by SGD optimizer with momentum equal to 0.9 and a weight decay of 5×10^{-4}. On CIFAR datasets, the algorithm starts with a learning rate of 0.1. 200 epochs are used to train the models, and the learning rate is reduced by 0.2 at the 60-th, 120-th, 160-th and 200-th epochs. On SVHN dataset, a learning rate of 0.01 is used at the beginning and is then divided by 10 at the 80-th and 120-th epochs within a total of 160 epochs. The same settings are used as in Zagoruyko and Komodakis (2016).

6.3. Results

To comprehensively compare our DeTraMe-Net with the DL technologies, the single-layer DL and deep DL are respectively evaluated. The single-layer DL comparison aims to show the improved discrimination power of our DeTrame-Net compared to that of state-of-the-art DL methods. The deep DL evaluation is used for an in-depth evaluation of the discriminative ability, efficiency, and robustness of our DeTrame-Net and of other learning methods.

6.3.1. DeTraMe-Net vs. Single-layer DL

First, we compare our results with six single-layer discriminative DL methods, which include two classical DL algorithms: Sparse Representation based Classification (SRC) Wright et al. (2009) and Collaborative Representation based Classification (CRC) Wright et al. (2009), and four state-of-the-art DL methods: Dictionary Learning with Structured Incoherence (DLSI) Ramirez et al. (2010b), Fisher Discrimination Dictionary Learning (FDDL) Yang et al. (2014), Label Consitent KSVD (LC-KSVD) Jiang et al. (2013) and Discriminative Fisher Embedding Dictionary Learning (DFEDL) Li et al. (2019).

For the small datasets, the proposed DeTraMe-Net achieves comparable or better performance. For the Extended YaleB dataset, although accuracy of the DeTraMe-Net is not the best one, it is still comparable to other competitive results. It is barely 0.2% lower than the FDDL Yang et al. (2014) and LC-KSVD Jiang et al. (2013), and still achieves the same performance as SRC Wright et al. (2009). For the AR dataset, DeTraMe-Net achieves the best performance with a 1.18% significant improvement over the 98% basis of the best state-of-the-art methods. For the objective dataset, Caltech101, DeTraMe-Net is just slightly lower than LC-KSVD Jiang et al. (2013) but is still equal or greater than other state-of-the-art approaches. For the Scene 15 dataset, DeTraMe-Net also achieves a 0.3% higher accuracy than state-of-the-art methods. The comparable performance, and sometimes significant improvements demonstrate that the discriminative ability of our DeTraMe-Net is truly increased by the independency between the dictionary variable and its pseudo-inverse. The new added variable of the dictionary pseudo-inverse plays an important role in re-weighting each input representation feature in the Q-Metric ReLU to obtain the best output feature representation.

In our comparison, we should point that breaking the dictionary and its pseudo-inverse into two independent components, helps enhance the discriminative capacity of our DETraME-Net. In contrast, the Fisher discrimination criterion was used in both FDDL Yang et al. (2014) and DFEDL Li et al. (2019), and label consistency was used in LC-KSVD Jiang et al. (2013). Comparing to other methods, this change in our method is lower cost than other discriminative techniques.

6.3.2. DeTraMe-Net vs. DDL

We subsequently compare our results with those achieved by the DDL approach in Mahdizadehaghdam et al. (2019), as both DeTraMe-Net and DDL with 9-layer follow the ALL-CNN architecture in Springenberg et al. (2014). DDL in Mahdizadehaghdam et al. (2019) is a state-to-the-art DDL based classifier, whose solution is based on alternatively updating the dictionaries and its sparse coefficients at each layer by K-SVD Aharon et al. (2006) and FISTA Beck and Teboulle (2009), while each layer dictionary and sparse coefficients of our DeTraMe-Net are obtained by a linear forward layer followed by a Q-Metric learning. In addition, the dictionary and its pseudo-inverse are coupled in Mahdizadehaghdam et al. (2019) while they are decoupled into two independent variables in DeTraMe-Net.

As shown in Table 4, DDL Mahdizadehaghdam et al. (2019) leads to improvements compared to PlainNet 9-layer Springenberg et al. (2014), which is similar to the results shown in Mahdizadehaghdam et al. (2019). However, to compare with DDL Mahdizadehaghdam et al. (2019) , our DeTraMe-Net obtain another improvements of accuracy and training and testing time over all datasets. In comparison to the best performance in Mahdizadehaghdam et al. (2019),

Accuracy (%)	CIFAR10 +		CIFAR100 +		SVHN	
Network Architectures	Original	DeTraMe-Net (#iteration)	Original	DeTraMe-Net (#iteration)	Original	DeTraMe-Net (#iteration)
PlainNet 3-layer	35.14 ± 4.94	**88.51** ± 0.17 (5)	22.01 ± 1.24	**64.99** ± 0.34 (3)	45.64	**97.21** (8)
PlainNet 6-layer	86.71 ± 0.36	**92.24** ± 0.32 (2)	62.81 ± 0.75	**69.49** ± 0.61 (2)	97.55	**98.17** (5)
PlainNet 9-layer	90.31 ± 0.31	**93.05** ± 0.46 (2)	66.15 ± 0.61	**69.68** ± 0.50 (2)	97.98	**98.26** (5)
PlainNet 12-layer	91.28 ± 0.27	**92.03** ± 0.54 (2)	68.70 ± 0.65	**70.92** ± 0.78 (2)	98.14	**98.27** (3)
ResNet 8	87.36 ± 0.34	**89.13** ± 0.23 (3)	60.38 ± 0.49	**64.50** ± 0.54 (2)	96.70	**97.50** (3)
ResNet 20	92.17 ± 0.15	**92.19** ± 0.30 (3)	68.42 ± 0.29	**68.62** ± 0.27 (2)	97.70	**97.82** (2)
ResNet 56	93.48 ± 0.16	**93.54** ± 0.30 (3)	**71.52** ± 0.34	**71.52** ± 0.44 (2)	97.96	**98.04** (2)
ResNet 110	93.57 ± 0.14	**93.68** ± 0.32 (2)	72.99 ± 0.43	**73.05** ± 0.40 (2)	-	-
WideResNet 16-4	**95.18** ± 0.10	**95.18** ± 0.13 (2)	76.72 ± 0.13	**76.85** ± 0.48 (3)	98.06	**98.16** (3)
WideResNet 16-8	95.62 ± 0.12	**95.66** ± 0.22 (2)	79.55 ± 0.12	**79.69** ± 0.55 (3)	98.17	**98.23** (3)

Table 6
CIFAR10 and CIFAR100 with + is trained with simple translation and flipping data augmentation. All the presented results are re-implemented and run by using the same settings. SVHN is too large to train, so it is only run once for reference.

the best DeTraMe-Net accuracy also respectively achieves 0.36% and 1.58% improvements on CIFAR10 and CIFAR100 datasets and, in terms of averaged performance, 0.01% and 0.92% accuracy improvements are still respectively obtained on these two datasets. Furthermore, when DDL Mahdizadehaghdam et al. (2019) processes a 28×28 image with 0.35M parameters in 0.2784 second for training and 9.4×10^{-2} s for testing, the proposed DeTraMe-Net processes a 32×32 image with 2.4M parameters in 0.1605 second for training and 3.52×10^{-4} s for testing. This shows that our method with 6 times more parameters than DDL only requires half training time and a faster testing time by a factor 100.

Although it is also shown in Table 4 that a higher number of parameters is involved in DeTraMe-Net than in Mahdizadehaghdam et al. (2019), DeTraMe-Net presents three main advantages: **(1)** it has a better discriminative capability, as we decouple the dictionary and its pseudo-inverse into two independent variables. **(2)** DeTraMe-Net is much faster for both training and testing phases. Since it is implemented in a neural network structure, with no need for extra functions to compute gradients at each layer, which greatly reduces the time complexity. **(3)** Also, owing to the network implementation, DeTraMe-Net is easy to scale up to 110 layers, while the maximum number of layers in Mahdizadehaghdam et al. (2019) is 23.

6.3.3. DeTraMe-Net vs. CNNs

We next compare DeTraMe-Net with CNNs with respect to five different aspects: Accuracy, Parameter number, Capacity, Adversarial robustness and robustness to random noise and Time complexity. In comparison to deep CNNs, first, the proposed DeTraMe-Net improves the classification accuracy with respect to the same architectures of CNNs over all three datasets. Then, in terms of trade-off between the accuracy and the number of parameters, for the same architecture, DeTraMe-Net has twice as many parameters as CNNs. However, by allocating the same number of parameters, DeTraMe-Net still achieves a better performance than CNNs. Note, that a significant improvement in performance for DeTraMe-Net is achieved with a lower network depth and a slightly larger width. In addition, DeTraMe-Net is demonstrated to be much more robust against the noise and adversarial perturbations than CNNs. Finally, while the training time for DeTraME with a larger number of parameters is about twice that of CNN, the testing time is generally only marginally higher. Moreover, our testing time is only slightly increased. In general, although DeTraMe-Net has more number of parameters, it also shows a better dicriminative power and robustness than CNNs.

Accuracy. As shown in Table 6, with the same architecture, using DeTraMe-Net structures achieves an overall better performance than all various CNN models do. For PlainNet architecture, DeTraMe-Net increases the accuracy with a median of 3.99% on CIFAR10, 5.11% on CIFAR100 and 0.45% on SVHN, and respectively increases the accuracy of at least 0.75%, 2.22%, 0.13% on theses three datasets. For ResNet architecture, DeTraMe-Net also consistently increases the accuracy with a median of 0.05% on CIFAR10, 0.13% on CIFAR100 and 0.10% on SVHN.

Parameter number. Although, in comparison to original standard CNNs, DeTraMe-Net involves more parameters for a given architecture, it improves the accuracy. Meanwhile, as demonstrated in Figure 2, for a given number of parameters, DeTraMe-Net still outperforms the original CNNs over all three datasets. Plots corresponding to DeTraMe-Net for both PlainNet and ResNet architectures are indeed above those associated with standard CNNs.

Capacity. In terms of *depth*, comparing improvements with PlainNet and ResNet, shows that the shallower the network, the more accurate. It is remarkable that DeTraMe-Net leads to more than 42% accuracy increase for PlainNet 3-layer on CIFAR10, CIFAR100 and SVHN datasets. When the networks become deeper, they better capture discriminative features of the classes, and albeit with smaller gains, DeTraMe-Net still achieves a better accuracy than a deep CNN, e.g. around 0.11% and 0.05% higher than ResNet 110 on CIFAR10 and CIFAR100. In terms of *width*, we use WideResNet-16-4 and WideResNet-16-8 as two reference

Model	CIFAR10+ ($\xi = 5e-2$)	CIFAR100+ ($\xi = 5e-2$)	SVHN($\xi = 2e-4$)
Original WideResNet-16-8	18.03% $\pm$ 1.75%	59.61% $\pm$ 2.72%	46.84% $\pm$ 3.05%
DeTraMe WideResNet-16-8	**6.78%** $\pm$ 0.73%	**23.65%** $\pm$ 1.84%	**23.61%** $\pm$ 5.61%

Table 7
Fooling Rate versus Adversarial Attack. ξ in Moosavi-Dezfooli, Fawzi, Fawzi and Frossard (2017) controls the attack magnitude.

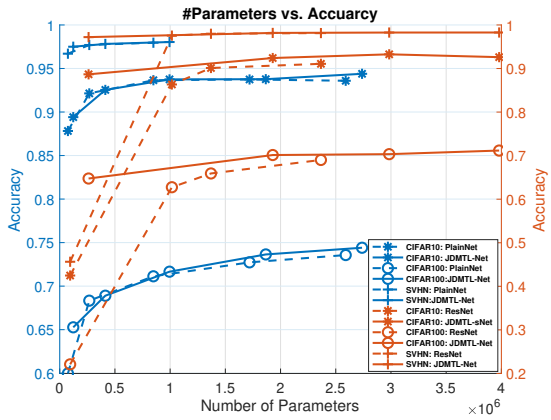


Figure 2: Classification accuracy versus number of parameters. The blue color curves are based on ResNet architecture (*left axis*), while the orange curves are based on PlainNet architecture (*right axis*). The solid line denotes DeTraMe-Net, while the dash-line denotes the original CNNs. '*' denotes for CIFAR10, 'o' denotes for CIFAR100 and '+' denotes for SVHN.

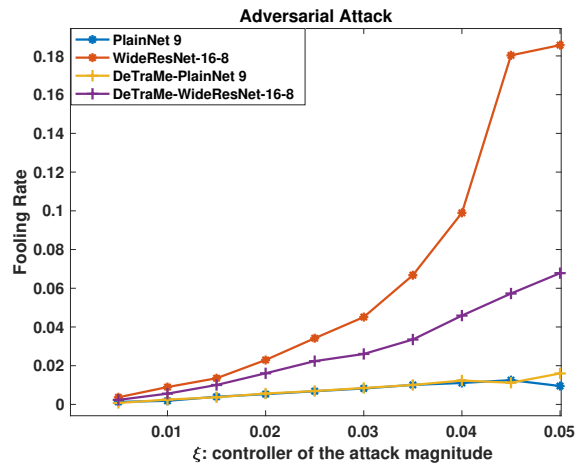


Figure 3: The fooling rate is averaged over 5 runs of CIFAR10+ dataset.

models, since both of them include 16 layers but have different widths. Table 6 shows that increasing width is beneficial to DeTraMe-Net. Since the original models have already achieved excellent performance for CIFAR10, CIFAR100 and SVHN, DeTraMe-Nets with various widths show similarly slightly improved accuracies. However, the experiments still demonstrate that enlarging the width for DeTraMe-Net leads to an increase in the accuracy gain. This may be explained by the fact that increasing width provides more redundant features, so yielding more flexibility for improvements.

Adversarial robustness and robustness to random noise. The UAP tool Moosavi-Dezfooli et al. (2017) is used to adversarially attack the best performance models of DeTraMe-Net and original CNN over 3 datasets. As shown in Table 7, the fooling rate of DeTraMe-Net is greatly reduced by more than half compared to the original CNN one. Moreover, by attacking PlainNet, Fig. 3 shows that while increasing the adversarial attack magnitudes, our DeTraMe-PlainNet has a performance similar to PlainNet architecture in terms of fooling rate. While in comparing with the ResNet architecture, DeTraMe-ResNet greatly reduces the fooling rate, probably by taking advantage of the firmly nonexpansiveness properties of the proximal operator in the Q-metric, which is also observed and mentioned in Hasannasab et al. (2020), a stacked proximal operator network. However, the robustness of residual networks in the presence of adversarial noise, remains theoretically an open issue, and hence deserves additional future investigation.

Concerning the robustness to random noise, we randomly generate a zero-mean Gaussian noise $\mathbf{v}$ and add it to the input data, where $\mathrm{E}(\|\mathbf{v}\|_2^2) = \rho\|\mathbf{x}\|_2^2$, ρ controls the magnitude of random noise level with respect to the average image energy.

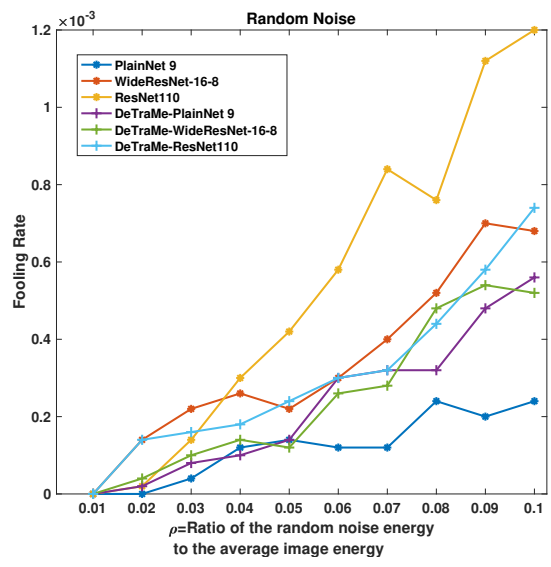


Figure 4: The fooling rate is averaged over 5 runs of CIFAR10 dataset.

As shown in Fig. 4, ResNet110 incurs a highest fooling rate as the noise is amplified by propagating deeply.

	Training Time (s)		Testing Time ($\times 10^{-4}$ s)	
	CIFAR10 +		CIFAR10 +	
Network Architectures	Original	DeTraMe	Original	DeTraMe
PlainNet 3-layer	2964.45	5837.72	1.93	2.50
PlainNet 6-layer	3762.75	6706.76	1.94	2.94
PlainNet 9-layer	3948.24	7451.42	2.01	3.24
PlainNet 12-layer	4087.98	8023.02	2.12	3.52
ResNet 8	3152.10	3962.09	1.76	2.10
ResNet 20	3840.02	5316.47	1.90	2.21
ResNet 56	6411.03	8752.61	2.27	3.37
ResNet 110	7709.55	12997.66	3.10	4.53
WideResNet 16-4	4562.02	6425.65	2.41	2.84
WideResNet 16-8	7104.62	11897.93	3.26	5.15

Table 8
CIFAR10 with + is trained with simple translation and flipping data augmentation. All the presented results are re-implemented and run by using the same settings.

However, WideResNet-16-8 incurs the second highest fooling rate, while our DeTraMe-ResNet110 and DeTraMe-WideResNet-16-8 achieve better performances. DeTraMe-PlainNet9 reaches a higher fooling rate than the original PlainNet, but it should be noticed that the magnitude of the fooling rate is very small, and our accuracy is about 3% higher than the one of the original PlainNet CNN.

Time complexity. Based on the running times in Table 8, training takes almost twice as much time than for CNNs. This appears consistent with the fact that the number of parameters of DeTraMe-Net is twice as many than for CNNs. However, it is worth noting that the training can be performed off-line and that testing can still be completed in real time, with only a slight increase of the testing time with respect to an original standard CNN, that is 100 times faster than a conventional DDL method (as shown in Table 5).

7. Conclusion

Starting from a DDL formulation, we have shown that it is possible to reformulate the problem in a standard optimization problem with the introduction of metrics within standard activation operators. This yields a novel Deep Transform and Metric Learning method. This has allowed us to show that the original DDL can be performed as a network mixing linear layer and RNN algorithmic structures, thus leading to a fast and flexible network framework for building more efficient and deeper DDL-based classifiers with a higher discriminiative ability. Our experiments show that the resulting DeTraMe-Net performs better than the original DDL approach and original CNNs with much more robustness against adversarial and random perturbations. We think that the bridge we established between DDL and DNN will help in further understanding and controlling these powerful tools so as to attain better performance and properties. It would also be interesting to explore other image processing applications and understand the scope of the proposed approach.

A. Alternative Derivation of Algorithm 1

We have presented in our paper a simple approach for deriving the recursive model:

$$\mathbf{U}_{t+1} = \mathrm{ReLU}\left((\mathbf{h}\mathbf{1}^\top) \odot \mathbf{Z} + \widetilde{\mathbf{W}}(\mathbf{U}_t - \mathbf{Z}) - \mathbf{b}\mathbf{1}^\top\right), \quad (24)$$

in order to compute

$$\mathrm{prox}^{\mathbf{Q}}_{\lambda\psi}(\mathbf{Z}) = \underset{\mathbf{U}\in\mathbb{R}^{k\times N}}{\mathrm{argmin}} \ \frac{1}{2}\|\mathbf{U}-\mathbf{Z}\|^2_{F,\mathbf{Q}} + \lambda\psi(\mathbf{U}). \quad (25)$$

We propose an alternative approach which is based on the classical forward-backward algorithm for solving the nonsmooth convex optimization problem in (25). The t-th iteration of the preconditioned form of this algorithm reads

$$\mathbf{U}_{t+1} = \mathrm{prox}^{\mathbf{\Theta}}_{\gamma\lambda\psi}(\mathbf{U}_t - \gamma\mathbf{\Theta}^{-1}\mathbf{Q}(\mathbf{U}_t - \mathbf{Z})) \quad (26)$$

where γ is a positive stepsize and $\mathbf{\Theta}$ is a preconditioning symmetric definite positive matrix, and $\mathbf{U}_0 \in \mathbb{R}^{k\times N}$. The algorithm is guaranteed to converge to the solution to (25) provided that

$$\gamma < \frac{2}{\|\mathbf{\Theta}^{-1/2}\mathbf{Q}\mathbf{\Theta}^{1/2}\|_{\mathrm{S}}}, \quad (27)$$

where $\|\cdot\|_{\mathrm{S}}$ denotes the spectral norm. Eq. (26) can be reexpressed as

$$\mathbf{U}_{t+1} = \mathrm{prox}^{\mathbf{\Theta}}_{\gamma\lambda\psi}\left((\mathbf{I} - \gamma\mathbf{\Theta}^{-1}\mathbf{Q})(\mathbf{U}_t - \mathbf{Z}) + \mathbf{Z}\right). \quad (28)$$

Assume now that $\mathbf{\Theta}$ is a diagonal matrix $\mathrm{Diag}(\theta_1, \dots, \theta_k)$ where, for every $i \in \{1, \dots, k\}$, $\theta_i > 0$. When the sparsity promoting penalization is chosen equal to

$$\psi = \|\cdot\|_1 + \imath_{[0,+\infty)^{k\times N}} + \frac{\beta}{2\lambda}\|\cdot\|^2_F, \quad (29)$$

the proximity operator involved in (28) simplifies as

$$\begin{aligned} &\forall \mathbf{U} = (u_{i,j})_{1\le i\le k, 1\le j\le N} \in \mathbb{R}^{k\times N}, \\ &\qquad \mathrm{prox}^{\mathbf{\Theta}}_{\gamma\lambda\psi} = \left(\mathrm{prox}_{\gamma\lambda\theta_i^{-1}\rho}(u_{i,j})\right)_{1\le i\le k, 1\le j\le N} \end{aligned} \quad (30)$$

where $\rho = \lambda|\cdot| + \imath_{[0,+\infty)} + \frac{\beta}{2}(\cdot)^2$. In addition, for every $u \in \mathbb{R}$ and $i \in \{1, \dots, k\}$,

$$\mathrm{prox}_{\gamma\lambda\theta_i^{-1}\rho}(u) = \underset{v\in[0,+\infty)}{\mathrm{argmin}} \ \frac{\theta_i}{2}(v-u)^2 + \gamma\left(\lambda|v| + \frac{\beta}{2}v^2\right). \quad (31)$$

After some simple algebra, this leads to

$$\mathrm{prox}_{\gamma\lambda\theta_i^{-1}\rho}(u) = \mathrm{ReLU}\left(\frac{\theta_i}{\theta_i + \gamma\beta}u - \frac{\gamma\lambda}{\theta_i + \gamma\beta}\right). \quad (32)$$

Altogether (28), (30), and (32) allow us to recover an update equation of the form (19), where

$$\begin{aligned} \widetilde{\mathbf{W}} &= (\mathbf{\Theta} + \gamma\beta\mathbf{I})^{-1}(\mathbf{\Theta} - \gamma\mathbf{Q}), \\ \mathbf{h} &= \left(\frac{\theta_i}{\theta_i + \gamma\beta}\right)_{1\le i\le k}, \\ \mathbf{b} &= \left(\frac{\gamma\lambda}{\theta_i + \gamma\beta}\right)_{1\le i\le k}. \end{aligned} \quad (33)$$

Note that, if $\gamma = 1$ and, for every $i \in \{1, \dots, k\}$, $\theta_i = q_{i,i}$, $\widetilde{\mathbf{W}}$ is a matrix with zeros on its main diagonal.

B. Illustration of DeTraMe-Net Architectures

B.1. DeTraMe-PlainNet

To replace all the RELU activation layers in PlainNet with Q-Metric ReLU leads to DeTraMe-PlainNet. Since all the RELU layers are replaced by Q-Metric ReLu, DeTraMe-PlainNet becomes equivalent to DDL.

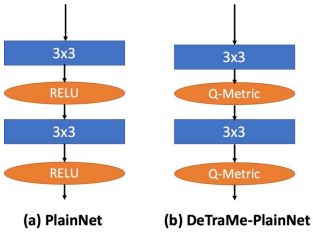


Figure 5: Architectures of PlainNet vs. DeTraMe-PlainNet

B.2. DeTraMe-ResNet

Replacing the RELU layer inside the block in ResNet by Q-Metric ReLU, allows us to build a new structure called DeTraMe-ResNet.

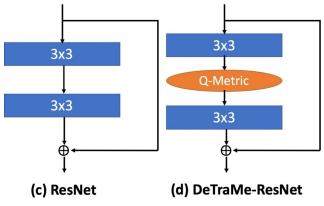


Figure 6: Architectures of ResNet vs. DeTraMe-ResNet

In our experiments, for ResNet architecture, the RNN part accounting for Q-Metric learning, makes use of 3×3 filters.

Acknowledgement

We gratefully acknowledge the partly supports of the U.S. Army Research Office under agreement W911NF1910202.

References

Aharon, M., Elad, M., Bruckstein, A., 2006. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on Signal Processing 54, 4311–4322.

Bach, F., Jenatton, R., Mairal, J., Obozinski, G., 2011. Optimization with sparsity-inducing penalties. arXiv preprint arXiv:1108.0775 .

Beck, A., Teboulle, M., 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences 2, 183–202.

Bian, X., Krim, H., Bronstein, A., Dai, L., 2016. Sparsity and nullity: Paradigms for analysis dictionary learning. SIAM Journal on Imaging Sciences 9, 1107–1126.

Boyd, S., Vandenberghe, L., 2004. Convex optimization. Cambridge university press.

Chouzenoux, E., Pesquet, J.C., Repetti, A., 2014. Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function. Journal of Optimization Theory and Applications 162, 107–132.

Combettes, P.L., Pesquet, J.C., 2010. Proximal splitting methods in signal processing, in: Bauschke, H.H., Burachik, R., Combettes, P.L., Elser, V., Luke, D.R., Wolkowicz, H. (Eds.), Fixed-Point Algorithms for Inverse Problems in Science and Engineering. Springer-Verlag, New York, pp. 185–212.

Combettes, P.L., Pesquet, J.C., 2020. Deep neural network structures solving variational inequalities. Set-Valued and Variational Analysis , 1–28.

Combettes, P.L., Pesquet, J.C., 2021. Fixed point strategies in data science. IEEE Transactions on Signal Processing .

Elad, M., Aharon, M., 2006. Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image processing 15, 3736–3745. URL: https://doi.org/10.1109/TIP.2006.881969, doi:10.1109/TIP.2006.881969.

Fei-Fei, L., Fergus, R., Perona, P., 2007. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding 106, 59–70.

Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J., 2001. From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Transactions on Pattern Analysis and Machine Intelligence 23, 643–660.

Gregor, K., LeCun, Y., 2010. Learning fast approximations of sparse coding, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, Omnipress. pp. 399–406.

Grosse, R., Raina, R., Kwong, H., Ng, A.Y., 2007. Shift-invariant sparse coding for audio classification, in: Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, AUAI Press, Arlington, Virginia, USA. p. 149–158.

Guo, J., Guo, Y., Kong, X., Zhang, M., He, R., 2016. Discriminative analysis dictionary learning, in: Thirtieth AAAI Conference on Artificial Intelligence.

Gupta, P., Maggu, J., Majumdar, A., Chouzenoux, E., Chierchia, G., 2020. DeConFuse: A Deep Convolutional Transform based Unsupervised Fusion Framework. Technical Report. Https://hal.archives-ouvertes.fr/hal-02461768.

Hasannasab, M., Hertrich, J., Neumayer, S., Plonka, G., Setzer, S., Steidl, G., 2020. Parseval proximal neural networks. Journal of Fourier Analysis and Applications 26, 1–31.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Huang, J.J., Dragotti, P.L., 2018. A deep dictionary model for image super-resolution, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'18), IEEE, Calgary, Canada.

Jiang, Z., Lin, Z., Davis, L.S., 2013. Label consistent k-svd: Learning a discriminative dictionary for recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 2651–2664.

Komodakis, N., Pesquet, J.C., 2014. Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems. IEEE Signal Processing Magazine 32, 31–5.

Krizhevsky, A., Hinton, G., et al., 2009. Learning multiple layers of features from tiny images. Technical Report. Citeseer.

Lazebnik, S., Schmid, C., Ponce, J., 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), IEEE. pp. 2169–2178.

Li, Z., Zhang, Z., Qin, J., Zhang, Z., Shao, L., 2019. Discriminative fisher embedding dictionary learning algorithm for object recognition. IEEE transactions on neural networks and learning systems 31, 786–800.

Liu, Y., Chen, Q., Chen, W., Wassell, I., 2018. Dictionary learning inspired deep network for scene recognition, in: Thirty-Second AAAI Conference on Artificial Intelligence.

Maggu, J., Majumdar, A., 2018. Unsupervised deep transform learning, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, Canada. pp. 6782–6786.

Mahdizadehaghdam, S., Dai, L., Krim, H., Skau, E., Wang, H., 2017. Image classification: A hierarchical dictionary learning approach, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing

(ICASSP), IEEE. pp. 2597–2601.

Mahdizadehaghdam, S., Panahi, A., Krim, H., Dai, L., 2019. Deep dictionary learning: A parametric network approach. IEEE Transactions on Image Processing .

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., Bach, F.R., 2009. Supervised dictionary learning, in: Advances in Neural Information Processing Systems, pp. 1033–1040.

Martinez, A., Benavente, R., June 1998. The ar face database. CVC Technical Report .

Moosavi-Dezfooli, S.M., Fawzi, A., Fawzi, O., Frossard, P., 2017. Universal adversarial perturbations, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1765–1773.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y., 2011. Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning.

Parikh, N., Boyd, S., 2014. Proximal algorithms. Foundations and Trends in optimization 1, 127–239.

Ramirez, I., Sprechmann, P., Sapiro, G., 2010a. Classification and clustering via dictionary learning with structured incoherence and shared features, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE. pp. 3501–3508.

Ramirez, I., Sprechmann, P., Sapiro, G., 2010b. Classification and clustering via dictionary learning with structured incoherence and shared features, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE. pp. 3501–3508.

Rubinstein, R., Peleg, T., Elad, M., 2013. Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model. IEEE Transactions on Signal Processing 61, 661–677.

Skau, E., Wohlberg, B., Krim, H., Dai, L., 2016. Pansharpening via coupled triple factorization dictionary learning, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 1234–1237.

Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M., 2014. Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806 .

Tang, W., Otero, I.R., Krim, H., Dai, L., 2016. Analysis dictionary learning for scene classification, in: Statistical Signal Processing Workshop (SSP), 2016 IEEE, IEEE. pp. 1–5.

Tang, W., Panahi, A., Krim, H., Dai, L., 2018. Structured analysis dictionary learning for image classification, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 2181–2185.

Tang, W., Panahi, A., Krim, H., Dai, L., 2019a. Analysis dictionary learning: an efficient and discriminative solution, in: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 3682–3686.

Tang, W., Panahi, A., Krim, H., Dai, L., 2019b. Analysis dictionary learning based classification: Structure for robustness. IEEE Transactions on Image Processing 28, 6035–6046.

Tariyal, S., Majumdar, A., Singh, R., Vatsa, M., 2016. Deep dictionary learning. IEEE Access 4, 10096–10109.

Wang, J., Guo, Y., Guo, J., Luo, X., Kong, X., 2017. Class-aware analysis dictionary learning for pattern classification. IEEE Signal Processing Letters 24, 1822–1826.

Wang, Q., Guo, Y., Guo, J., Kong, X., 2018. Synthesis k-svd based analysis dictionary learning for pattern classification. Multimedia Tools and Applications 77, 17023–17041.

Wang, Z., Liu, D., Yang, J., Han, W., Huang, T., 2015. Deep networks for image super-resolution with sparse prior, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 370–378.

Wang, Z., Yang, J., Nasrabadi, N., Huang, T., 2013. A max-margin perspective on sparse representation-based classification, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 1217–1224.

Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y., 2009. Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 210–227.

Xu, M., Jia, X., Pickering, M., Plaza, A.J., 2016. Cloud removal based on sparse representation via multitemporal dictionary learning. IEEE Transactions on Geoscience and Remote Sensing 54, 2998–3006. URL: https://app.dimensions.ai/details/publication/pub.1061614193, doi:10.1109/tgrs.2015.2509860.

Yang, M., Zhang, L., Feng, X., Zhang, D., 2014. Sparse representation based fisher discrimination dictionary learning for image classification. International Journal of Computer Vision 109, 209–232.

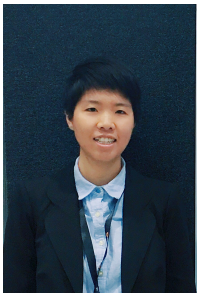
Zagoruyko, S., Komodakis, N., 2016. Wide residual networks. arXiv preprint arXiv:1605.07146 .

Zhang, D., Liu, P., Zhang, K., Zhang, H., Wang, Q., Jing, X., 2016. Class relatedness oriented-discriminative dictionary learning for multiclass image classification. Pattern Recognition 59, 168–175. URL: https://doi.org/10.1016/j.patcog.2015.12.005, doi:10.1016/j.patcog.2015.12.005.

Zhang, L., Yang, M., Feng, X., 2011. Sparse representation or collaborative representation: Which helps face recognition?, in: 2011 International Conference on Computer Vision, pp. 471–478. doi:10.1109/ICCV.2011.6126277.

Zhong, W., 2012. Robust object tracking via sparsity-based collaborative model, in: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, USA. p. 1838–1845.

Zhou, J.T., Di, K., Du, J., Peng, X., Yang, H., Pan, S.J., Tsang, I.W., Liu, Y., Qin, Z., Goh, R.S.M., 2018. Sc2net: Sparse lstms for sparse coding, in: Thirty-Second AAAI Conference on Artificial Intelligence.



Wen Tang received one B.S. degree in Information and Computing Science from East China University of Science and Technology, Shanghai, China, in 2012. In the same year, she also received another B.S. degree in Computer Science from Fayetteville State University, Fayetteville, NC, USA. In 2019, she was a visiting student at the Center for Visual Computing at CentraleSupelec, France. Then, She received her Ph.D. degree in the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, in 2020. Her current research interests include machine learning / deep learning and its applications for computer vision tasks.

Jean-Christophe Pesquet (IEEE Fellow 2012, EURASIP Fellow 2021) received the engineering degree from Supélec, Gif-sur-Yvette, France, in 1987, the Ph.D. and HDR degrees from the University Paris-Sud in 1990 and 1999, respectively. From 1991 to 1999, he was an Assistant Professor at the University Paris-Sud, and a research scientist at the Laboratoire des Signaux et Systèmes (CNRS). From 1999 to 2016, he was a Professor with University Paris-Est Marne-la-Vallée and from 2012 to 2016, he was the Deputy Director of the Laboratoire d'Informatique of the university (UMR-CNRS 8049). He is currently a Distinguished Professor with CentraleSupélec, University Paris-Saclay, and the director of the Center for Visual Computing (OPIS Inria group). He was also a senior member of the Institut Universitaire de France from 2016 to 2021. His research interests include multiscale analysis, statistical signal processing, inverse problems, imaging, and optimization methods with applications to data sciences and artificial intelligence.

Emilie Chouzenoux received the engineering degree from Ecole Centrale, Nantes, France, in 2007, and the Ph.D. degree in signal processing from the Institut de Recherche en Communications et Cybernétique (IRCCyN, UMR CNRS 6597), Nantes, in 2010. Between 2011 and 2019, she was a Maître de conferences at the University of Paris-Est Marne-la-Vallée, Champs-sur-Marne, France (LIGM, UMRCNRS 8049). Since September 2019, she has been a Researcher at Inria Saclay, within the project team OPIS, in Centre pour la Vision Numérique, CentraleSupélec. She is an Associated Editor of IEEE Transactions in Signal Processing. Since January 2020, she has been the PI of the ERC Starting Grant MAJORIS. Her research interests are in large scale optimization algorithms for inverse problems and machine learning problems of image processing.

Hamid Krim received the B.Sc. and M.Sc. degrees in electrical engineering. He was a Member of Technical Staff at AT&T Bell Labs, where he has conducted research and development in the areas of telephony and digital communication systems/subsystems. Following an NSF Postdoctoral Fellowship at Foreign Centers of Excellence, LSS/University of Orsay, Paris, France, he joined the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, as a Research Scientist and where he was performing and supervising research. He is currently a Professor of electrical engineering in the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, leading the Vision, Information, and Statistical Signal Theories and Applications Group. His research interests include statistical signal and image analysis and mathematical modeling with a keen emphasis on applied problems in classification and recognition using geometric and topological tools. He has served on the SP society Editorial Board and on TCs, and is the SP Distinguished Lecturer for 2015-2016, as has served in the US Army Research Office as a Research Program Manager.