An α -Divergence Based Approach for Robust Dictionary Learning

Asif Iqbal, *Student Member, IEEE* and Abd-Krim Seghouane, *Senior Member, IEEE*

Abstract—In this paper, a robust sequential dictionary learning (DL) algorithm is presented. The proposed algorithm is motivated from the maximum likelihood perspective on dictionary learning and its link to the minimization of the Kullback-Leibler divergence. It is obtained by using a robust loss function in the data fidelity term of the DL objective instead of the usual quadratic loss. The proposed robust loss function is derived from the α -divergence as an alternative to the Kullback-Leibler divergence, which leads to a quadratic loss. Compared to other robust approaches, the proposed loss has the advantage of belonging to class of redescending M-estimators, guaranteeing inference stability from large deviations from the Gaussian nominal noise model. The algorithm is obtained by solving a sequence of penalized rank-1 matrix approximation problems where the ℓ_1 -norm is introduced as a penalty promoting sparsity and then using a block coordinate descent approach to estimate the unknowns. Performance comparison with similar robust DL algorithms on digit recognition, background removal, and gray-scale image denoising are performed highlighting efficacy of the proposed algorithm.

Index Terms— α -divergence, dictionary learning, robust estimation

I. INTRODUCTION

In the last two decades, sparse signal representation [1] has emerged as a powerful solution to a wide range of signal processing problems. It has been successfully applied to solve problems like compressed sensing [2], image restoration [3], fMRI signal analysis [4]–[7], and face recognition [8] etc. Sparse representation can be directly linked to the problem of compressed sensing (CS) [2], i.e., under the sparse representation framework, an original signal can be represented by using a small number of coefficients corresponding to a given basis set (Fourier, DCT, wavelets). This number is usually much smaller than the respective Nyquist criterion. Such succinct representations not only lead to reduction in storage and communication requirements, but also help in faster processing and extraction of important information from the data.

Generally, in the presence of noise, a sparse approximation instead of a sparse representation can lead to even sparser representations. For an efficient sparse approximation, choice of the basis set is key. In [3], authors have shown that learning a basis (dictionary) from the data itself leads to a better sparse representation as compared to the use of pre-specified basis sets such as wavelets, curvelets, contourlets, Fourier transforms, and DCT etc. Thus, a lot of effort has been dedicated

to learn a dictionary [9] whose columns (atoms) can lead to a good sparse representation of the data. Most popular dictionary learning (DL) methods [3], [10] pose the DL problem with an ℓ_2 -norm fidelity term and a sparse regularization term. These data-driven methods can be seen as assuming the noise prior to be Gaussian, which leads to a squared ℓ_2 -norm data fidelity term as the maximum likelihood estimate. Consider a signal matrix $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\} \in \mathbb{R}^{n \times N}$ and a sparsity constraint, DL algorithms seek to find an overcomplete dictionary $\mathbf{D} \in \mathbb{R}^{n \times K}$ ($N > K > n$) and a sparse representation matrix $\mathbf{X} \in \mathbb{R}^{K \times N}$ such that $\|\mathbf{Y} - \mathbf{DX}\|_F^2 \leq \epsilon$ for some $\epsilon > 0$. The Frobenius norm used here leads to a quadratic loss function given as

$$\|\mathbf{Y} - \mathbf{DX}\|_F^2 = \sum_{i=1}^N \sum_{m=1}^n (y_i^m - \mathbf{d}^m \mathbf{x}_i)^2,$$

where $\mathbf{d}^m$ is the m^{th} row of $\mathbf{D}$, and y_i^m is the entry from i^{th} column and m^{th} row of $\mathbf{Y}$. If the Gaussian noise prior holds for the training data $\mathbf{Y}$, the dictionaries learned via such loss functions lead to state-of-the-art performance. However, this loss is adapted for the Gaussian noise assumption, it is very sensitive to the so-called *outliers* and offers no protection against the anomalous observations frequently encountered in various image processing applications [11].

The methods developed in robust statistics [12] can provide ways to address and counteract the effects of such *outliers* in different regression and classification tasks. By making assumptions on the outlier statistics, they develop learning methods that are insensitive to their presence. Several robust DL methods have been developed recently using tools from the area of robust statistics [13]–[19]. These methods replace the ℓ_2 -norm data fidelity term with the ℓ_1 -norm error, also called the *least absolute deviation* [12, Ch. 7.11], Huber loss, or truncated ℓ_1 -norm error term to counter the effects of outliers, leading to an outlier resistant robust dictionary estimate. Authors in [18] used ℓ_1 -norm data fidelity to separate the background from foreground in a given video sequence by modeling the background as a sparse linear representation with respect to a learned dictionary and the foreground as the sparse representation error. The authors in [19] proposed an online DL for novel document detection problem. Using the prior that representation error for text documents often contain large impulsive coefficients, they learned the dictionary by imposing ℓ_1 -penalty on the residuals. The DL method proposed in [17] used Huber-loss data fidelity which enjoys faster computations and robust tracker estimations without utilizing trivial object templates. Authors in [13] use the ℓ_2 -norm of the representa-

Asif Iqbal and Abd-Krim Seghouane are with the Department of Electrical and Electronic Engineering, University of Melbourne, Australia.

This work was supported in part by the Australian Research Council through Grant FT. 130101394.

tion error as a metric to classify an entire signal as an outlier, excluding it from the training dataset.

Instead of identifying outliers, the methods outlined above, use statistics of the outliers to diminish their effects on the data representations and the learned dictionaries. On the other hand, there is another class of robust methods like [20]–[22] which try to capture the influence of outliers using data models and develop tools for their identification. These methods do not use the statistical information of outliers and develop robust estimators using the notion that only a few outliers are present in the data. They model each training signal with a sparse linear approximation w.r.t. a learned dictionary and an outlier signal. They use ℓ_2 -norm data fidelity term while forcing the outlier signal to be sparse using an ℓ_1 -norm penalty term in the formulation. As a consequence, these methods can identify outliers quite readily.

In this paper, we propose a new robust DL algorithm which has the advantage of minimizing the assumption on the noise. It is obtained by using a function derived from the α -divergence [23] to be used in the data fidelity term of the DL objective function instead of the quadratic loss or the Frobenius norm. The proposed algorithm is robust in the sense that the learning is carried out using an objective function guaranteeing a stable estimate in the presence of outliers. In particular, the function used in the fidelity term belongs to the class of redescending M-estimators which guarantee stability of inference for relatively large deviations from the nominal noise model [24]. The approach adopted for deriving the proposed algorithm exploits the observation that the observed data matrix can be approximated as a sum of rank-1 matrix approximation [25]–[27]. The proposed algorithm is obtained via adaptive sequential penalized rank-1 matrix approximation where a block coordinate descent approach is used to determine the unknowns of the different rank-1 approximation matrices. The adaptive aspect of the algorithm allows different amount of shrinkage to be used for different entries of the sparse code matrix $\mathbf{X}$ [28]. Furthermore, the DL algorithm has the advantage of having simple sparse coding and dictionary update stages as the former corresponds to a vector thresholding step whereas the later corresponds to matrix vector multiplication. An early version of part of this work appears in conference form [29]. In this journal version additionally we provide the derivation of the proposed loss function from the α -divergence, a motivation for using the α -divergence for the development of the proposed dictionary learning algorithm as an alternative to maximum likelihood, discuss the robust aspect of the proposed procedure and provide performance comparison on two more image processing applications; namely ‘Background subtraction’ and ‘Image denoising’. We begin by reviewing some background work on DL in Section II. The proposed DL algorithm is described in Section III, experimental evaluation is provided in Section IV followed by concluding remarks in Section V.

II. BACKGROUND

Here, we briefly review some background work on DL that sets the framework for the method developed in this study.

Given a collection of signals $\mathbf{Y} \in \mathbb{R}^{n \times N}$ and a sparsity constraint, DL algorithms seek to find an overcomplete dictionary $\mathbf{D} \in \mathbb{R}^{n \times K}$ and a sparse representation matrix $\mathbf{X} \in \mathbb{R}^{K \times N}$ by attempting to optimize the following objective function

$$\min_{\mathbf{D}, \mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{x}_i\|_0 \leq s, \forall 1 \leq i \leq N, \\ \text{and} \quad \|\mathbf{d}_j\|_2 = 1, \forall 1 \leq j \leq K \quad (1)$$

where $\mathbf{d}_j$ and $\mathbf{x}_i$ denote the j^{th} column of the dictionary $\mathbf{D}$ and i^{th} column of the sparse code matrix $\mathbf{X}$. Variants of (1) have also been used to design DL algorithms, among these, one can cite methods that replace the ℓ_0 -norm sparsity constraint by an ℓ_1 -norm penalty for sparsity [25], [26], [28], [30]–[33] or includes additional properties on the dictionary such as incoherence [34], [35].

With the ℓ_1 -norm as a penalty promoting sparsity, the DL objective (1) becomes

$$\min_{\mathbf{D} \in \mathcal{D}, \mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 + \lambda \|\mathbf{X}\|_1 \quad (2)$$

where the $\|\mathbf{X}\|_1 = \sum_{i=1}^N \sum_{j=1}^K |x_i^j|$, λ is the regularization or tuning parameter controlling the amount of sparsity, and $\mathcal{D} = \{\mathbf{d}_j \in \mathbb{R}^n \mid \|\mathbf{d}_j\|_2 = 1 \forall j\}$. While the objective is not convex, it admits a multiconvex structure [36] and can be solved using an alternating optimization scheme that guarantees the convergence to a local minimum. This is the approach used in most algorithms for DL that attempt to approximately solve (1) which leads to a two stage alternating optimization scheme composed of a sparse coding stage and a dictionary update stage. In sparse coding stage, given a fixed dictionary $\mathbf{D}$, they optimize $\mathbf{X}$ by solving

$$\min_{\mathbf{x}_i} \sum_{i=1}^N \{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 + \lambda \|\mathbf{x}_i\|_1\} \quad (3)$$

where $\mathbf{y}_i$ and $\mathbf{x}_i$ are the column vectors of $\mathbf{Y}$ and $\mathbf{X}$ respectively. The sparse coding problem (3) consists of a set of ℓ_1 -regularized least squares problems. A number of efficient methods are available for solving this type of problem, among them one can cite the homotopy method [37] or coordinate descent with soft-thresholding [38], [39]. Next, the dictionary update stage optimizes $\mathbf{D}$ given a fixed $\mathbf{X}$ by solving

$$\mathbf{D} = \arg \min_{\mathbf{D}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \quad \text{s.t.} \quad \mathbf{D} \in \mathcal{D}. \quad (4)$$

Methods used for the dictionary update stage either update the atoms sequentially [3] by breaking the global minimization (4) into K sequential minimization problems or simultaneously all at once [10]. DL algorithms of this form solve (1) or its variants typically by alternating between the sparse coding stage and the dictionary update stage until convergence.

Another approach recently adopted in [25]–[28] for DL methods consists in approximating the dataset $\mathbf{Y}$ by a penalized sum of rank-1 matrices and uses a simple and exact block coordinate descent approach to estimate the factors of various rank-1 matrices. This approach can be seen as an extension of the power method or alternating least square method [40] for computing the rank-1 matrix approximation generated by the SVD. This is the approach adopted in [25]–[28] and it works

directly on $\mathbf{E}_j = \mathbf{Y} - \sum_{i=1, i \neq j}^K \mathbf{d}_i \mathbf{x}^i$ to generate new pairs $(\mathbf{d}_j, \mathbf{x}^j)$ using the objective

$$\{\mathbf{d}_j, \mathbf{x}^j\} = \arg \min_{\mathbf{d}_j, \mathbf{x}^j} \|\mathbf{E}_j - \mathbf{d}_j \mathbf{x}^j\|_F^2 + \lambda \|\mathbf{x}^j\|_1 \text{ s.t. } \mathbf{D} \in \mathcal{D}. \quad (5)$$

The resulting algorithm is a variant of the power method or alternating least square method for regularized rank-1 approximation where a sparsity penalty is introduced in the minimization problem to promote sparsity of $\mathbf{x}^j$ row. The estimates of $\mathbf{d}_j$ and $\mathbf{x}^j$ are then given by

$$\mathbf{d}_j = \frac{\mathbf{E}_j \mathbf{x}^j{}^\top}{\|\mathbf{E}_j \mathbf{x}^j{}^\top\|_2}, \quad (6)$$

$$\mathbf{x}^j = \text{sgn}(\mathbf{d}_j^\top \mathbf{E}_j) \circ \left(|\mathbf{d}_j^\top \mathbf{E}_j| - \frac{\lambda}{2} \mathbf{1}_{(N)} \right)_+ \quad (7)$$

where $\circ$, $|\cdot|$, $\text{sign}(\cdot)$, $(x)_+$ define the entrywise variants of Hadamard product, absolute value, sign, and $\max(0, x)$ functions respectively. The $\mathbf{1}_N$ is a vector of ones of size N . As illustrated by different experimental results [25], [28], this alternative DL scheme leads to improved performance compared to state-of-the-art methods. In the next section, this approach is adopted for the development of an extension of [28] for robust dictionary learning.

III. PROPOSED METHOD

From a probabilistic point of view, the DL objective (2) can be seen as a penalized maximum likelihood (ML) estimation approach with the assumptions that the signals, $\mathbf{y}_i \in \mathbb{R}^n$, $i = 1, \dots, N$ are conditionally i.i.d. $\mathcal{N}(\mathbf{D}\mathbf{x}_i, \sigma^2 \mathbf{I})$ and the entries of $\mathbf{x}_i$ are zero-mean i.i.d. with Cauchy or Laplace distributions [31]. The quadratic loss derived from the Gaussian noise assumption and used in (2) is however very sensitive to outliers and offers no protection against anomalous observations or signals encountered in various signal and image processing applications.

Below we introduce a robust loss function and use it, instead of the quadratic loss, for stable DL from signals corrupted by contaminated Gaussian noise. Before its derivation, the link between ML and Kullback-Leibler (KL) divergence minimization is reviewed as a motivation for the proposed approach.

A. Maximum likelihood estimation

Consider an i.i.d. data sample $\mathbf{z}_1, \dots, \mathbf{z}_N$, where $\mathbf{z}_i \in \mathbb{R}^p$ sampled from a probability distribution $F(\mathbf{z}, \boldsymbol{\theta}^*)$ with $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^m$ and let $\{F(\mathbf{z}, \boldsymbol{\theta}), \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ be a family of approximating probability distributions indexed by $\boldsymbol{\theta}$. The estimation of $\boldsymbol{\theta}$ from $\mathbf{z}_1, \dots, \mathbf{z}_N$ is then obtained by using an estimating function $\psi(\mathbf{z}, \boldsymbol{\theta})$ satisfying

$$\begin{aligned} \int \psi(\mathbf{z}, \boldsymbol{\theta}) dF(\mathbf{z}, \boldsymbol{\theta}^*) &= 0 \quad \text{if } \boldsymbol{\theta} = \boldsymbol{\theta}^* \\ \int \psi(\mathbf{z}, \boldsymbol{\theta})^2 dF(\mathbf{z}, \boldsymbol{\theta}^*) &< \infty \end{aligned} \quad (8)$$

The ML estimate which corresponds to the least squares estimates under Gaussian assumption is the one obtained with an estimating equation function corresponding to the log-likelihood. From this point of view, other functions for ψ can be

used for estimation and makes the ML estimation a particular case in this more general framework [41].

B. Relation to Kullback-Leibler divergence

The ML is obtained from data sample $\mathbf{z}_1, \dots, \mathbf{z}_N$ using the log-likelihood function $\log(f(\mathbf{z}, \boldsymbol{\theta}))$ and is given by

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{ML} &= \arg \max_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^N \log(f(\mathbf{z}_i, \boldsymbol{\theta})) \\ &= \arg \max_{\boldsymbol{\theta}} \int \log(f(\mathbf{z}, \boldsymbol{\theta})) f_N(\mathbf{z}) d\mathbf{z} \\ &= \arg \max_{\boldsymbol{\theta}} \int \log(f(\mathbf{z}, \boldsymbol{\theta})) dF_N(\mathbf{z}) \\ &= \arg \min_{\boldsymbol{\theta}} \int \log(f(\mathbf{z}, \boldsymbol{\theta}^*)) dF_N(\mathbf{z}) \\ &\quad - \int \log(f(\mathbf{z}, \boldsymbol{\theta})) dF_N(\mathbf{z}) \\ &= \arg \min_{\boldsymbol{\theta}} \int \log(f(\mathbf{z}, \boldsymbol{\theta}^*)) dF(\mathbf{z}, \boldsymbol{\theta}^*) \\ &\quad - \int \log(f(\mathbf{z}, \boldsymbol{\theta})) dF(\mathbf{z}, \boldsymbol{\theta}^*) \\ &= \arg \min_{\boldsymbol{\theta}} KL(f(\mathbf{z}, \boldsymbol{\theta}^*), f(\mathbf{z}, \boldsymbol{\theta})) \end{aligned} \quad (9)$$

where $f_N(\mathbf{z}) = \frac{1}{N} \sum_{i=1}^N \delta(\mathbf{z} - \mathbf{z}_i)$ is the empirical distribution function which puts a mass of $\frac{1}{N}$ at $\mathbf{z}_i$ s and the last line of (9) is obtained from the law of large numbers [42]. Therefore, the ML estimate is also the one that minimizes a specific discrepancy between a family of parameterized distributions and the true distribution. In this case of ML, the discrepancy is the KL divergence.

C. From Kullback-Leibler divergence to α -divergence

The KL is, however, sensitive to outliers and offers no protection against anomalous observations or signals. From this point of view, a more general discrepancy or measure of similarity that encompasses the KL divergence and offers protection against contaminated Gaussian noise can be employed. Such measures of similarity exist in the class of divergences referred to as α -divergence [43]. For two probability density functions $f(\mathbf{z}, \boldsymbol{\theta}^*)$ and $f(\mathbf{z}, \boldsymbol{\theta})$, the α -divergence is defined as [43]

$$\begin{aligned} D_\alpha(f(\mathbf{z}, \boldsymbol{\theta}^*) \parallel f(\mathbf{z}, \boldsymbol{\theta})) &= \\ \frac{1}{\alpha(\alpha-1)} \left[\int f(\mathbf{z}, \boldsymbol{\theta}^*)^\alpha f(\mathbf{z}, \boldsymbol{\theta})^{1-\alpha} d\mathbf{z} - 1 \right]. \end{aligned} \quad (10)$$

This generalization of the KL divergence defines a class of divergences indexed by a single parameter α . When $\alpha = 1$, it reduces to the KL divergence and the method of ML. This can be seen by deriving D_α in terms of the α -logarithm function

$$\log_\alpha(x) = \frac{x^{(1-\alpha)} - 1}{1 - \alpha} \quad (11)$$

where $\log_1(x) = \log(x)$. This is detailed in the Appendix. Compared to the KL divergence, this divergence provides us with a loss function adapted to address robustness against Gaussian noise contamination. Its derivation is obtained by

drawing a parallel representation of the ML estimation in (9) by quantifying the discrepancy between $f(\mathbf{z}, \boldsymbol{\theta})$ and $f(\mathbf{z}, \boldsymbol{\theta}^*)$ in terms of the α -divergence instead of the KL divergence. Similar to the KL divergence, the α -divergence can be split into the difference of two terms using $\log_\alpha(\cdot)$ as follows

$$D_\alpha \left(\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) \parallel f(\mathbf{z}, \boldsymbol{\theta}) \right) = \frac{-1}{\eta} \int f(\mathbf{z}, \boldsymbol{\theta}^*) \left[\log_\alpha \{f(\mathbf{z}, \boldsymbol{\theta})\} + \log_\alpha \left\{ \tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) \right\} \right] d\mathbf{z} \quad (12)$$

where

$$\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) = \frac{f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha}}{\int f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha} d\mathbf{z}}, \eta = \alpha \left[\int f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha} d\mathbf{z} \right]^\alpha$$

The derivation of (12) is detailed in the Appendix. An alternative to the ML estimate can then be obtained by finding $\boldsymbol{\theta}$ that minimizes (12) or

$$D_\alpha \left(\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) \parallel f(\mathbf{z}, \boldsymbol{\theta}) \right) = \frac{1}{\eta} \int f(\mathbf{z}, \boldsymbol{\theta}^*) \left[\log_\alpha \left\{ \tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) \right\} - \log_\alpha \{f(\mathbf{z}, \boldsymbol{\theta})\} \right] d\mathbf{z} \quad (13)$$

instead of minimizing the KL divergence (9). Since the first term of (13) is independent of $\boldsymbol{\theta}$, it doesn't affect the minimization of D_α with respect to $\boldsymbol{\theta}$. Therefore, minimizing the α -divergence D_α with respect to $\boldsymbol{\theta}$ is equivalent to the following maximization

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}} \int \log_\alpha \{f(\mathbf{z}, \boldsymbol{\theta})\} dF(\mathbf{z}, \boldsymbol{\theta}^*) \quad (14)$$

or

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^N \log_\alpha \{f(\mathbf{z}_i, \boldsymbol{\theta})\} \quad (15)$$

when considering the empirical distribution. Assuming the probabilistic DL data model $\mathbf{y} = \mathbf{D}\mathbf{x} + \boldsymbol{\epsilon}$ introduced at the beginning of section III where the index i has been suppressed for simplicity of presentation and where $\boldsymbol{\epsilon} = \mathbf{y} - \mathbf{D}\mathbf{x}$ is a zero mean Gaussian white residual vector of dimension n with variance σ^2 . Considering the entry ϵ_i of $\boldsymbol{\epsilon}$ as z_i then

$$\begin{aligned} f(\epsilon_i, \boldsymbol{\theta}) = f(y_i - \mathbf{d}^i \mathbf{x}) &\propto \exp \left(-\frac{(y_i - \mathbf{d}^i \mathbf{x})^2}{2\sigma^2} \right) \\ &= \exp \left(-\frac{\epsilon_i^2}{2\sigma^2} \right) \end{aligned} \quad (16)$$

where $\mathbf{d}^i$ is the i^{th} row of $\mathbf{D}$ and $\boldsymbol{\theta}$ represents either $\mathbf{D}$ or $\mathbf{x}$ depending on the dictionary learning optimization stage. Using the definition of $\log_\alpha$ (11) and the probability density function (16) in (15) gives

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \arg \max_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \log_\alpha \{f(\epsilon_i, \boldsymbol{\theta})\} \\ &= \arg \max_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \frac{f(\epsilon_i, \boldsymbol{\theta})^\tau - 1}{\tau} \\ &= \arg \max_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \frac{\exp \left(-\frac{\tau}{2\sigma^2} \epsilon_i^2 \right) - 1}{\tau} \end{aligned}$$

$$\begin{aligned} &= \arg \min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \frac{1}{\tau} \left(1 - \exp \left(-\frac{\tau}{2} \left(\frac{\epsilon_i}{\sigma} \right)^2 \right) \right) \\ &= \arg \min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^n \ell_\tau(v_i) \end{aligned} \quad (17)$$

where $v_i = \epsilon_i/\sigma$ and $\ell_\tau(v_i)$ is defined as

$$\ell_\tau(v_i) = \frac{1}{\tau} \left(1 - \exp \left(-\frac{\tau v_i^2}{2} \right) \right) \quad (18)$$

for $v_i \in \mathbb{R}$. If $\tau \rightarrow 0$, we have $\ell_\tau(v_i) \rightarrow \ell_0(v_i) = v_i^2/2$, and the familiar quadratic loss associated with the Frobenius norm is recovered, which is highly sensitive to the outliers present in the data. More insight can be gained by differentiating (18) to obtain the score function

$$\ell'_\tau(v_i) = v_i \exp \left(-\frac{\tau v_i^2}{2} \right) \quad (19)$$

represented in Fig. 1. (d) and observing that this leads to a weighted least squares estimation (as detailed in subsections III.D and III.E) with weights proportional to $\exp \left(-\frac{\tau v_i^2}{2} \right)$ that are nearly zero for outliers. The special case $\tau = 0$ or $\alpha = 1$ corresponds to uniform weights equal to 1 and the resulting estimator is just the least squares which offers no protection against outliers present in the data. The case $\tau > 0$, corresponds to a weighted estimation that tends to downweight the errors that are far from the nominal density, effectively removing outliers' and offering stability of inference in the learning process [24]. Due to the form of the weights, anomalous data or outliers far from the bulk of the data are automatically downweighted and have little impact on the final estimate which makes the DL robust to outliers. It can be seen that the score function satisfies

$$\lim_{v_i \rightarrow \infty} \ell'_\tau(v_i) = 0$$

which makes the proposed loss redescending.

D. Proposed Dictionary Learning Approach

The proposed robust DL algorithm is derived by using a variant of (2) where the function (18), derived above, is used instead of the Frobenius norm in the fidelity term to give

$$\min_{\mathbf{D}, \mathbf{X}} \ell_\tau(\mathbf{Y} - \mathbf{D}\mathbf{X}) + \lambda \|\mathbf{X}\|_1 \text{ s.t. } \mathbf{D} \in \mathcal{D} \quad (20)$$

where $\|\mathbf{X}\|_1 = \sum_{i=1}^N \sum_{j=1}^K |x_i^j|$ is the sparse regularization term, λ is the sparsity controlling parameter, and

$$\ell_\tau(\mathbf{Y} - \mathbf{D}\mathbf{X}) = \sum_{i=1}^N \sum_{m=1}^n \ell_\tau \left(\frac{y_i^m - \mathbf{d}^m \mathbf{x}_i}{\sigma} \right) \quad (21)$$

where ℓ_τ (termed Gaussian fidelity in Fig. 1) is defined in (18) and $\mathbf{d}^m$ is the m^{th} row of $\mathbf{D}$. The proposed procedure is robust in the sense that the DL is carried out using an objective function guaranteeing a stable solution in the presence of outliers. In particular, the function used in the fidelity term of (20) leads to an estimator in the class of redescending M-estimators which guarantees stability of inference for relatively large deviations from the nominal noise model [24].

To solve the problem in (20), we start by expressing the matrix $\mathbf{D}\mathbf{X}$ as a sum of K rank-1 matrices i.e. $\sum_{j=1}^K \mathbf{d}_j \mathbf{x}^j$, and propose an exact block coordinate descent approach to estimate the various rank-1 matrices as done in [25]–[28]. Thus, we start by writing (20) as

$$\{\mathbf{d}_j, \mathbf{x}^j\} = \min_{\mathbf{d}_j \in \mathcal{D}, \mathbf{x}^j} \sum_{i=1}^N \ell_\tau \left(\frac{\mathbf{e}_{ij} - \mathbf{d}_j x_i^j}{\sigma} \right) + \lambda |\mathbf{x}_i^j| \quad (22)$$

where $\mathbf{e}_{ij}$ is the i^{th} column of $\mathbf{E}_j$ and $\mathbf{E}_j = \mathbf{Y} - \sum_{i=1, i \neq j}^K \mathbf{d}_i \mathbf{x}^i$. The objective in (22) aims at estimating components of one rank-1 matrix at a time and to solve it, we propose an iterative alternating optimization strategy. During each iteration of the algorithm, we perform K penalized rank-1 matrix approximations using (22). The updates for $\mathbf{d}_j$ and $\mathbf{x}^j$ are found by minimizing (22) over one variable at a time, while fixing the other. Solutions for each of the variables are derived in their respective sections next.

1) *Atom update*: With a fixed $\mathbf{x}^j$, the update for $\mathbf{d}_j$ is found by solving

$$\mathbf{d}_j = \min_{\mathbf{d}_j \in \mathcal{D}} \sum_{i=1}^N \left(\frac{1}{\tau} - \frac{1}{\tau} \exp \left(-\frac{\tau (\mathbf{e}_{ij} - \mathbf{d}_j x_i^j)^2}{2\sigma^2} \right) \right). \quad (23)$$

Differentiating (23) w.r.t. $\mathbf{d}_j$ and setting it to zero, we get

$$\sum_{i=1}^N \exp \left(-\frac{\tau (\mathbf{e}_{ij} - \mathbf{d}_j x_i^j)^2}{2\sigma^2} \right) \frac{1}{\sigma} (\mathbf{d}_j x_i^j - \mathbf{e}_{ij}) = 0 \quad (24)$$

with the solution for $\mathbf{d}_j$ given by

$$\mathbf{d}_j = \left(\sum_{i=1}^N \mathbf{W}_i x_i^j \right)^{-1} \left(\sum_{i=1}^N \mathbf{W}_i \mathbf{e}_{ij} x_i^j \right) \quad (25)$$

followed by ℓ_2 -normalization of $\mathbf{d}_j$. Here $\mathbf{W}_i \in \mathbb{R}^{n \times n}$ is a diagonal weighting matrix with $\exp(-\tau(\mathbf{e}_{ij} - \mathbf{d}_j x_i^j)^2/2\sigma^2)$ as weights on the diagonal.

2) *Sparse code update*: Next, we fix the atom $\mathbf{d}_j$, and find an update for $\mathbf{x}^j$ by solving

$$x_i^j = \min_{x_i^j} \ell_\tau (\mathbf{e}_{ij} - \mathbf{d}_j x_i^j) + \lambda |x_i^j| \quad \forall i = 1, \dots, N. \quad (26)$$

Taking derivative of (26) w.r.t. x_i^j and setting it to zero, we get

$$\frac{1}{\sigma} \mathbf{d}_j^\top \mathbf{W}_i (\mathbf{e}_{ij} - \mathbf{d}_j x_i^j) + \lambda \text{sign}(x_i^j) = 0 \quad (27)$$

here $\mathbf{W}_i$ is the weighting matrix and the solution for x_i^j is given by

$$x_i^j = \text{sign}(\eta_i) \left(|\eta_i| - \tilde{\lambda}_i \right)_+ \quad \forall i = 1, \dots, N \quad (28)$$

with $\eta_i = \frac{\mathbf{d}_j^\top \mathbf{W}_i \mathbf{e}_{ij}}{\mathbf{d}_j^\top \mathbf{W}_i \mathbf{d}_j}$ and $\tilde{\lambda}_i = \frac{\lambda \sigma}{\mathbf{d}_j^\top \mathbf{W}_i \mathbf{d}_j}$.

here for simplicity, we assume $\sigma = 1$. The resulting $(\mathbf{d}_j, \mathbf{x}^j)$ updates can be seen as a variant of the iterative re-weighted least squares method for rank-1 matrix approximation and based on our experience, only requires 3–6 iterations of (25) and (28) to reach a local minimum.

Fig. 1. (a) ℓ_2 , ℓ_1 , and Gaussian fidelity (18) terms, (b) respective weight functions, (c) Gaussian weights for different ζ values, (d) the score function for multiple values of τ .

E. Weights

The weighting function resulting from differentiation of (22) with respect to either $\mathbf{d}_j$ or x_i^j , leading to a weighted least squares estimating equation is given by

$$\mathbf{W}_i (\mathbf{e}_{ij}, \mathbf{d}_j, x_i^j, \zeta) = \exp \left(-\zeta/2 (\mathbf{e}_{ij} - \mathbf{d}_j x_i^j)^2 \right) \quad (29)$$

with weights that are nearly zero for outliers (i.e., when the residual matrix entry $\mathbf{e}_{ij}$ is far from $\mathbf{d}_j x_i^j$). As the estimation of σ^2 in the presence of contaminated noise is not trivial, we combine it with the parameter τ as $\zeta = \tau/\sigma^2$ and estimate ζ instead as outlined in section IV-A. The special case $\zeta = 0$ (used in first iteration of the algorithm), corresponds to uniform weights $w_1 = \dots = w_n = 1$ and the implied learning algorithm is the one obtained by the Frobenious norm [28]. On the other hand, if $\zeta > 0$, entries of $\mathbf{e}_{ij}$ far from the $\mathbf{d}_j x_i^j$ receive relatively low weight compared to observations near $\mathbf{d}_j x_i^j$. Due to the form of the weights, anomalous observations far from the bulk of the data are automatically down-weighted and have little impact on the final estimate which makes the learning algorithm robust against outliers.

F. Discussion

For each of K atom-sparse code updates, we compute $\mathbf{W}_i$ using $(\mathbf{e}_{ij} - \mathbf{d}_j x_i^j)$, which can also be written as $\mathbf{g}_i = \mathbf{y}_i - \mathbf{D}\mathbf{x}_i$. Upon inspection of $\mathbf{g}_i$ during each K atom-sparse code updates, the relative changes were found to be very small, which was expected as the $\mathbf{g}_i$ entries with high amplitudes (*outliers*) are down-weighted and have no impact on the final estimates $(\mathbf{d}_j, x_i^j)$, thus remaining largely unchanged.

Subsequently we tested the algorithm with a single $\mathbf{W}_i$ computation for all K updates and found negligible performance differences in terms of algorithm convergence and the final results. However, with this strategy, the computational cost of the algorithm decreased by a factor of K . Thus we

opted to use this efficient implementation for all experimental evaluations and summarize it in algorithm 1.

Algorithm 1: Efficient implementation of the proposed robust DL algorithm.

Input: Data matrix $\mathbf{Y} \in \mathbb{R}^{n \times N}$, $\mathbf{D} \in \mathbb{R}^{n \times K}$, λ , δ , *noIt*

1 **Initialization:** Set $\mathbf{X} = 0$, $\epsilon = 10^{-4}$, and $\mathbf{W}_i = \mathbf{I}$, $\forall i = 1, \dots, N$.

2 **for** $t = 1 : \text{noIt}$ **do**

3 **if** $t \neq 1$ **then**

4 Using δ and $\mathbf{g}_i = \mathbf{y}_i - \mathbf{D}\mathbf{x}_i$, compute $\mathbf{W}_i$ $\forall i = 1, \dots, N$ as outlined in section IV-A.

5 **for** $j = 1 : K$ **do**

6 Compute $\mathbf{E}_j = \mathbf{Y} - \sum_{i=1, i \neq j}^K \mathbf{d}_i \mathbf{x}_i^i$

7 **while** $\|\mathbf{d}_j^t - \mathbf{d}_j^{t-1}\|_2 > \epsilon$ **do**

8 Compute η_i and $\tilde{\lambda}_i$ using (28)

9 $x_i^j = \text{sign}(\eta_i) \left(|\eta_i| - \tilde{\lambda}_i \right)_+ \quad \forall i = 1, \dots, N$

10 $\mathbf{d}_j = \left(\sum_{i=1}^N \mathbf{W}_i x_i^j x_i^j \right)^{-1} \left(\sum_{i=1}^N \mathbf{W}_i \mathbf{e}_{ij} x_i^j \right)$

11 $\mathbf{d}_j = \mathbf{d}_j / \|\mathbf{d}_j\|_2$

Output: $\mathbf{D}, \mathbf{X}$

G. Computation Complexity

Being an iterative DL algorithm, its computational complexity depends on the number of iterations. However, for the non-iterative part, the calculation for N $\mathbf{W}_i$'s requires $\mathcal{O}(nKN)$ operations. For each atom-sparse code update, we calculate $\mathbf{E}_j$ with $\mathcal{O}(nKN)$ complexity, then for the sparse codes, N η_i 's and $\tilde{\lambda}_i$'s require $\mathcal{O}(n^3N)$ computations each, making the sparse coding stage worth $\mathcal{O}(n^3N)$ operations. The atom update requires $\mathcal{O}(n^2N)$ for computation of numerator term and $\mathcal{O}(n^3)$ for the inverse operation. As $N > n$, the atom update stage complexity becomes $\mathcal{O}(n^2N)$. Thus, a single atom-sparse code update requires $\mathcal{O}(n^3N)$ operations, resulting in the overall complexity of $\mathcal{O}(n^3NK)$ for all K approximations.

IV. EXPERIMENTS

In the following experimental section, we illustrate robustness of the proposed algorithm with respect to other state-of-the-art methods against the presence of strong outliers in the data. The applications selected for such evaluation are digit recognition, background subtraction, and gray-scale image denoising. The implementation details of each application are given below in their respective sections.

A. Parameter setting

The parameter ζ in (29) tunes the extent to which we down-weight anomalies present in the data (see Fig. 1 c)), and its choice is of practical importance in applications. Let $\mathbf{e} \in \mathbb{R}^n$ be the residual vector and γ be the demarcation point such that if the square of an entry of $\mathbf{e}$ is above γ , its corresponding weight will be less than β . Thus to find the optimum parameter

ζ for $\mathbf{e}$, we need to find a relationship between ζ and γ . Such a relationship can be obtained from (29), and is given by

$$\zeta = -2 \ln(\beta) / \gamma \quad (30)$$

here β is the respective weight at demarcation point γ . The selection of γ is performed using the following procedure. Let $I = \lceil \delta n \rceil$, where $\delta \in (0, 1)$ is a scalar, and $\lceil \delta n \rceil$ outputs smallest integer larger than δn . Then we set γ to be the I^{th} largest element from the set $\{e_m^2, m = 1, \dots, n\}$. The parameter δ corresponds to the expected proportion of outliers present in $\mathbf{e}$ and can be used in conjunction with β to control the shape (sharpness of the slope) of the weighting function. However, instead of tuning both parameters, we fix $\beta = 0.5$, and tune δ to induce robustness into the dictionary learning process. To find the optimum δ and sparsity controlling parameter λ , in all experiments (unless specified otherwise), we perform a 2D grid search over $\delta \in [0, 0.5]$ and $\lambda \in [0, \lambda_{\max}]$, and select parameters leading to best experimental performance.

B. Evaluation on Digit Recognition

Digit recognition is an integral part of any document processing systems. These systems require large amount of training data to learn from, however, the data could contain different types of contaminations (noise, occlusion / outliers). One way of tackling such problems are to find and discard the highly contaminated signals before the learning process. This technique, however, becomes prohibitive as the training size increases. Under such conditions, a technique which is robust against such contaminations could prove beneficial.

In this section we present performance comparison of the proposed robust DL method with respect to K-SVD [3], ORDL¹ [16], and L1-KSVD² [14]. We use USPS [44] and MNIST [45] handwritten digits datasets for the evaluation. Each dataset contains two sub-datasets for training and testing purposes. USPS dataset contains 7,291 training and 2,007 testing digit images, whereas, the MNIST dataset has 60,000 training and 10,000 testing digit images. We resized all gray-scale images to the same size of 16×16 pixels and normalized them w.r.t. the maximum pixel intensity. We generated outliers of the same size using a higher order polynomial with random parameters and added them to the training as well as testing images while making sure that the overlapping pixel intensities did not exceed 1. Few examples of corrupted images from MNIST training dataset are shown in Fig. 2. These corrupted training images were vectorized and placed as columns of a training matrix $\mathbf{Y} \in \mathbb{R}^{n \times N}$, where $n = 256$ and N represents the number of training samples.

We learn 10 class-specific dictionaries for each digit where the individual dictionaries were initialized with K samples from the respective training data itself. The sparsity parameters were set to $\lambda = 0.1$ for ORDL, L1-KSVD, and the proposed algorithm. For K-SVD, the sparsity parameter s was set to $s = K$. For the proposed algorithm, we set $\delta = 0.1$ for USPS and $\delta = 0.12$ for the MNIST datasets. Tuning parameters

¹The source code available at http://www.cse.cuhk.edu.hk/~leo/jia/programs/ORDL_code.zip

²Code provided by the authors

Fig. 2. Few examples from MNIST training digits with outliers.

δ and λ were selected using 2D grid search as outlined in section IV-A. The learning process was iterated 40 times or was stopped early if the relative dictionary change³ dropped below 0.01. The batch-size for ORDL was set to 250. Once the learning was complete, given a test sample $\mathbf{y}_i$, we used orthogonal matching pursuit (OMP) [46] to solve the following problem:

$$\mathbf{x}_i = \min_{\mathbf{z}} \|\mathbf{y}_i - \mathbf{D}\mathbf{z}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{z}\|_0 \leq 2K \quad (31)$$

where $\mathbf{D} \in \mathbb{R}^{n \times 10K}$ is the full dictionary. Using the resulting sparse vector $\mathbf{x}_i$, we calculate the representation error w.r.t. all class-specific dictionaries and select the one with smallest representation error. We repeated this procedure over $K = \{2, 3, 4\}$ and 20 trials, and the mean recognition results are reported in Table I. We present the results on outlier free test datasets in Table II as well. Looking at both tables, we observe that dictionaries learned by the proposed algorithm outperformed the competition. For visualization, the recovered dictionaries (for $K = 3$) are shown in Fig. 3 highlighting that the proposed algorithm was able to fully reject the outliers while learning a set of clean representative atoms. The runtime of a single trial for all algorithms is given in Table III. By analyzing tables I, II, and III, we see that although L1-KSVD's recognition results are close to the proposed algorithms' results, it is highly computationally inefficient (around 10 times slower).

TABLE I
MEAN RECOGNITION RESULTS ON CONTAMINATED TEST DATASETS.

	K	K-SVD	ORDL	L1-KSVD	Proposed
USPS	2	0.764	0.804	0.826	0.844
	3	0.775	0.819	0.845	0.856
	4	0.788	0.824	0.851	0.870
MNIST	2	0.677	0.696	0.719	0.792
	3	0.649	0.690	0.722	0.797
	4	0.649	0.669	0.726	0.800

C. Evaluation on Background Subtraction

In this section, we evaluate the proposed algorithm on the problem of background subtraction. The input to this problem is a video sequence, and the aim is to use all frames of the sequence to learn a dictionary which can effectively model the background, considering all the pixels that deviate from this model as foreground or outliers. In [16], [18] this problem was cast as a sparse signal recovery problem, i.e.,

³ $\|\mathbf{D}_t - \mathbf{D}_{t-1}\|_F / \|\mathbf{D}_t\|_F < 0.01$

TABLE II
MEAN RECOGNITION RESULTS ON OUTLIER FREE TEST DATASETS.

	K	K-SVD	ORDL	L1-KSVD	Proposed
USPS	2	0.805	0.824	0.852	0.857
	3	0.826	0.841	0.870	0.883
	4	0.837	0.854	0.879	0.895
MNIST	2	0.761	0.796	0.820	0.845
	3	0.760	0.813	0.849	0.864
	4	0.756	0.823	0.865	0.878

Fig. 3. Recovered dictionaries for MNIST dataset by all methods.

once the background is learned by the dictionary, then for some frame $\mathbf{y}_i$, its background can be modeled using the linear combination of only a few atoms from the dictionary $\mathbf{D}$. Let $\mathbf{x}_i$ be the corresponding sparse coefficient vector, then the foreground (residual) of $\mathbf{y}_i$ is given by $|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i|$. Three gray-scale video sequences were used for the evaluation, i.e., *airport* sequence⁴ and two sequences from the UCSD dataset [47]. In all video sequences, the camera position was fixed. Out of the two UCSD datasets, sequence one (*UCSD 1*) had few pedestrians i.e., low outlier density, whereas, sequence two (*UCSD 2*) had a high number of pedestrians i.e., high outlier density in all of the frames. Image size of the airport sequence is 176×144 with a total of 400 frames. The image size of UCSD datasets is 180×120 with 200 frames. Each video frame was vectorized and placed as column in the training matrix $\mathbf{Y}$. We used ORDL [16], ODL⁵ (DL using ℓ_2 norm data fidelity term) [33], L1-KSVD [14], and the proposed algorithm to train dictionaries using all available frames. The dictionaries were initialized with Gaussian entries followed by ℓ_2 normalization. The dictionary size was kept fixed at $K = 5$ for all algorithms and all algorithms were iterated 40 times to

⁴The source code available at <http://vis-www.cs.umass.edu/~narayana/castanza/I2Rdataset/>

⁵The source code available at <http://spams-devel.gforge.inria.fr/>

TABLE III
THE RUNTIME OF ALL ALGORITHMS IN SECONDS

	K	K-SVD	ORDL	L1-KSVD	Proposed
USPS	2	4.0	14.4	56.8	3.4
	3	5.4	17.0	69.7	4.4
	4	7.1	17.0	75.0	5.8
MNIST	2	38.1	54.4	416.1	37.9
	3	53.8	63.7	542.4	47.2
	4	71.3	61.3	557.9	61.5

learn the dictionaries. The best performing sparsity controlling parameters (in terms of F-score in Table V) are given in Table IV. For the proposed algorithm, for *airport* and *UCSD 1* sequences, $\delta = 0.01$, and for *UCSD 2* sequence, $\delta = 0.1$. The optimum values for λ and δ were found by performing a 2D grid search over $\lambda \in (0, 10)$ and $\delta \in (0.01, 0.2)$ as discussed in section IV-A.

TABLE IV
SPARSITY CONTROLLING PARAMETER λ FOR ALL ALGORITHMS.

		ODL	ORDL	L1-KSVD	Proposed
λ	Airport	5	5	5	1
	UCSD 1	5	5	5	1
	UCSD 2	5	0.05	5	0.2

Once the dictionaries were learned, we picked two query images from the *airport* sequence, and generated the foreground and background using the learned dictionaries. The results are presented in Fig. 4 showing that the proposed algorithm was able to model the background effectively while discarding the outliers. ORDL and L1-KSVD algorithms' results are promising as well, but in both query images, they failed to fully remove the passengers standing in middle of the frame, whereas, the ODL failed completely. Similarly, the results for UCSD 1 and UCSD 2 datasets are shown in Fig. 5 as well. Here the best performance was achieved by the proposed and L1-KSVD algorithms. To provide quantitative comparison, we used three ground truth foreground masks from the *airport* sequence to calculate the *precision*, *recall*, and F-score metrics. The F-score tends to stay close to smaller one of *precision* and *recall*, thus we chose the threshold value that maximized the F-score for each method. These results are reported in Table V and the frames along with the recovered foregrounds and masks are shown in Fig. 6. From the figure and table, we can see that the results with highest accuracy are from the proposed method.

TABLE V
THE PRECISION, RECALL, AND F-SCORES CORRESPONDING TO THE RESULTS SHOWN IN FIG. 6.

Method	Frame 1			Frame 2			Frame 3		
	Prec	Reca	F-sc	Prec	Reca	F-sc	Prec	Reca	F-sc
ODL	0.771	0.634	0.696	0.863	0.582	0.695	0.752	0.590	0.661
ORDL	0.745	0.680	0.711	0.865	0.788	0.825	0.897	0.744	0.813
L1-KSVD	0.660	0.654	0.657	0.893	0.787	0.836	0.917	0.721	0.807
Proposed	0.830	0.657	0.734	0.892	0.796	0.841	0.912	0.736	0.814

D. Gray-scale Image Denoising

In this section, we compare the denoising performance of proposed algorithm with ORDL [16], L1-KSVD [14], and W-KSVD⁶ [11] under two noise corruption scenarios, i.e., random valued noise and mixture of Gaussian and random valued noise. All used images have been resized to 256×256 pixels, converted to gray-scale, and their pixel values lie in the range of $[0, 1]$. The noise model used here [11] can be written as

$$n = \begin{cases} n_1, & \text{with probability } 1 - r, \\ n_2, & \text{with probability } r, \end{cases} \quad (32)$$

⁶The source code available at <http://math0.bnu.edu.cn/~liujun/>

here n_1 represents Gaussian noise and n_2 is the random valued noise. The random valued noise is generated by using uniformly distributed values between $[0, 1]$. Starting with the noisy image, we extract every other 8×8 patch from the image, vectorize it, remove its mean, and place it as columns of a training matrix $\mathbf{Y} \in \mathbb{R}^{64 \times 15,625}$. Using an overcomplete DCT matrix as the initial dictionary $\mathbf{D} \in \mathbb{R}^{64 \times 100}$, all algorithms use the training matrix $\mathbf{Y}$ to learn a dictionary such that $\mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{E}$, where $\mathbf{X}$ is the sparse coefficient matrix and $\mathbf{E}$ contains the residuals. The batch size for ORDL [16] was set at 1,000 and DL iterations for W-KSVD [11], L1-KSVD [14], and the proposed algorithms were set to (20, 10, 30) respectively. The sparsity controlling parameter λ for ORDL, L1-KSVD, and the proposed algorithm were selected empirically. For the proposed algorithm, we assume the noise density level known, thus we set $\delta = r$. Once the learning process is complete, the denoised image is reconstructed by using a simple weighted average scheme [14]. The complete implementation detail for all algorithms under both contamination scenarios are given in their respective sections below.

1) *Random Valued Noise*: Random-valued (RV) noise belongs to the impulse noise category. This type of noise is not additive in nature but replaces intensity of the affected pixel with an entry from $\mathcal{U}[0, 1]$. The amount of pixel corruption of a clean gray-scale image I is controlled by a parameter r , referred to as the *noise density*, i.e., $r \times P$ pixels will be affected by RV noise. Here P represents the total number of pixels. The resulting noisy image is converted into a training matrix $\mathbf{Y}$ (as outlined in section IV-D) and used by all algorithms to produce a denoised image. We used multiple images and various noise density levels for denoising performance evaluation. The testing include the widely used *lena*, *barbara*, *boat*, and *baboon* images with $r = (0.1, 0.2, 0.3, 0.4)$. For each image and noise density level, we performed 6 denoising trials, each time using a new noise realization. The final peak signal to noise ratios (PSNR) of the noisy and denoised images recovered by different algorithms are given in Table VI. Here it can be seen that the proposed algorithm was able to outperform other competing robust DL methods with W-KSVD performing slightly better for the test image *lena*. By visually inspecting the denoised images, we found that W-KSVD method was over-smoothing the images which leads to better performance on images like *lena*, which does contain a lot of smooth intensity areas, but performs poorly for images with a lot of textures like *boat*, *barbara*, and *baboon*. This effect is highlighted in Fig. 7 (first row), where the W-KSVDs result looks very good, however upon close inspection of area above the ship, it can be seen that its beams and wires have been completely smoothed out. This effect, however, is not observed in other denoised images, where the wires and beams were still recovered.

2) *Gaussian mixture noise*: In this subsection, we consider the Gaussian plus random-valued noise mixture model as given in eq (32), i.e., $r \times P$ image pixels were affected by RV noise, while the remaining $(1 - r) \times P$ pixels were contaminated by additive white Gaussian noise entries from $\mathcal{N}(0, \sigma^2)$. We contaminated the image *barbara* using different

 Fig. 4. Background (top) and foregrounds (bottom) from the *Airport* dataset. From left to right: Query frame, ODL, ORDL, L1-KSVD, Proposed algorithm.

 Fig. 5. Background (top) and foregrounds (bottom) from the *UCSD* dataset. From left to right: Query frame, ODL, ORDL, L1-KSVD, Proposed algorithm.

Fig. 6. On left, query frames along with foregrounds generated by ODL, ORDL, L1-KSVD, and the proposed algorithm. On right, the ground truth foreground mask and the respective masks recovered for all algorithms. The quantitative results are given in Table V.

TABLE VI
MEAN PSNR VALUES (IN dB) OF NOISY AND DENOISED IMAGES
RECOVERED BY DIFFERENT METHODS OVER 6 TRIALS IN THE PRESENCE
OF RANDOM-VALUED NOISE.

Images	r	Noisy	ORDL	L1-KSVD	W-KSVD	Proposed
Lena	0.1	19.10	30.10	29.47	34.65	33.80
	0.2	16.31	26.74	26.07	33.16	32.28
	0.3	14.51	24.05	24.21	30.23	29.77
	0.4	13.31	21.53	21.78	26.85	27.46
Barbara	0.1	18.87	29.60	28.15	33.31	33.78
	0.2	16.01	25.60	25.93	31.28	31.18
	0.3	14.18	22.74	23.24	28.90	29.12
	0.4	12.96	20.59	21.04	20.61	26.59
Boat	0.1	19.35	28.82	27.77	29.10	32.27
	0.2	16.39	25.31	25.39	27.65	29.80
	0.3	14.57	22.79	23.34	26.11	27.37
	0.4	13.38	21.08	21.36	23.24	25.23
Baboon	0.1	19.89	28.39	26.35	27.16	29.11
	0.2	16.99	25.74	25.15	25.27	27.76
	0.3	15.22	23.76	23.82	23.89	26.61
	0.4	14.04	22.33	22.68	22.11	25.28
Mean		15.94	24.95	24.74	27.74	29.21

σ and r combinations, where $\sigma = (10, 15, 25)/255$ and $r = (0.1, 0.2, 0.3)$. Similar to subsection IV-D1, we repeated the denoising process 6 times and recorded the PSNR values of the denoised images generated by all algorithms. The mean PSNR values for all algorithms are presented in Table VII.

 Fig. 7. First row: *Boat* image corrupted by random-valued noise with $r = 0.2$ only. Second row: *Barbara* image corrupted by Gaussian mixture noise with $\sigma = 15/255$ and $r = 0.2$. The respective denoised images recovered by different algorithms follow the noisy image.

From the table, we can see that the highest PSNR values over all σ and RV noise density r combinations belong to our method. For visual comparison, the denoised images for the Gaussian mixture noise combination of $\sigma = 15/255$ and $r = 0.2$ are shown in Fig. 7 (second row). Here again, the over-smoothing effect of W-KSVD algorithm can be seen.

All experiments were carried out on a Windows 7 PC with a 3.5 GHz Core i7 processor and 16 GB memory. The overall average runtime for all experiments discussed in section IV-D is reported in Table VIII highlighting the computational efficiency of the proposed algorithm.

TABLE VII
MEAN PSNR VALUES (IN DB) OF NOISY *Barbara* IMAGE AND DENOISED IMAGES RECOVERED BY DIFFERENT METHODS OVER 6 TRIALS IN THE PRESENCE OF GAUSSIAN MIXTURE NOISE.

σ_r	10/255			15/255			25/255		
	0.1	0.2	0.3	0.1	0.2	0.3	0.1	0.2	0.3
Noisy	18.43	15.80	14.05	17.93	15.55	13.91	16.65	14.83	13.47
ORDL	28.03	24.87	22.09	26.69	24.27	22.13	25.30	23.36	21.46
L1-KSVD	28.08	25.32	22.96	27.60	24.68	22.58	26.02	23.81	21.66
W-KSVD	29.42	27.79	25.65	27.70	25.77	22.24	25.47	23.09	20.68
Proposed	30.50	29.26	27.80	29.17	27.90	26.42	26.28	25.23	24.31

TABLE VIII
AVERAGE RUNTIMES (SECONDS) OF DIFFERENT METHODS OVER ALL EXPERIMENTS GIVEN IN SECTION IV-D

Method	ORDL	L1-KSVD	W-KSVD	Proposed
Time	229	923	520	152

V. CONCLUSION

In this paper, a new DL algorithm is presented to learn robust dictionary atoms from training datasets contaminated by strong outliers. The algorithm is inspired from the maximum likelihood estimation perspective while minimizing the assumption on the noise model corrupting the data. To this end, we derived a loss function belonging to the class of redescending M-estimators from the so-called α -divergence. The resulting loss function guarantees stability of inference even for relatively large deviations from the nominal noise model. Thus, instead of using quadratic loss (which is only effective for Gaussian noise case), we use the derived loss function in the DL objective function. We exploit the observation that the data matrix can be approximated by a sum of rank-1 matrices and use an adaptive sequential penalized rank-1 matrix approximation to learn these rank-1 matrices. A block coordinate descend approach is adopted to update the atom and the respective sparse codes using simple closed form solutions. Experimental results on digit recognition, background subtraction, and mixed-noise gray-scale denoising applications showed that the proposed algorithm is not only computationally faster than its competitors, but provides superior performance as well.

APPENDIX

This section provides the derivation of the α -divergence split into the difference of two terms given in (12). The derivation of D_α in terms of $\log_\alpha$ is first provided and then the split into the difference of two terms is presented. Note, for notational convenience, single variables probability densities $p(x)$ and $q(x)$ are used in this section, the generalization to $f(\mathbf{z}, \boldsymbol{\theta})$ and $f(\mathbf{z}, \boldsymbol{\theta}^*)$ is straightforward. Below, p corresponds to the true underlying probability density.

Deriving D_α in terms of $\log_\alpha$

Using the definition $\int p(x)dx = 1$ in (10) gives

$$\begin{aligned}
 D_\alpha(p(x) \parallel q(x)) &= \frac{1}{\alpha(\alpha-1)} \left[\int p(x)^\alpha q(x)^{1-\alpha} dx - \int p(x) dx \right] \\
 &= \frac{1}{\alpha(\alpha-1)} \left[\int p(x) \left(\frac{q(x)}{p(x)} \right)^{1-\alpha} dx - \int p(x) dx \right] \\
 &= \frac{1}{\alpha} \frac{1}{\alpha-1} \left[\int p(x) \left[\left(\frac{q(x)}{p(x)} \right)^{1-\alpha} - 1 \right] dx \right] \\
 &= -\frac{1}{\alpha} \int p(x) \left[\frac{\left(\frac{q(x)}{p(x)} \right)^{1-\alpha} - 1}{1-\alpha} \right] dx \\
 &= -\frac{1}{\alpha} \int p(x) \log_\alpha \left(\frac{q(x)}{p(x)} \right) dx.
 \end{aligned} \tag{33}$$

In terms of expectation with respect to the density $p(x)$

$$D_\alpha(p(x) \parallel q(x)) = -\frac{1}{\alpha} E \left[\log_\alpha \left(\frac{q(x)}{p(x)} \right) \right].$$

Splitting D_α into the difference of two terms

From (33) we have

$$\begin{aligned}
 D_\alpha(p(x) \parallel q(x)) &= -\frac{1}{\alpha} \int p(x) \left[\frac{\left(\frac{q(x)}{p(x)} \right)^{1-\alpha} - 1}{1-\alpha} \right] dx \\
 &= -\frac{1}{\alpha} \int \frac{p(x)}{1-\alpha} \frac{q(x)^{1-\alpha} - 1 - p(x)^{1-\alpha} + 1}{p(x)^{1-\alpha}} dx \\
 &= -\frac{1}{\alpha} \int p(x)^\alpha \left[\frac{q(x)^{1-\alpha} - 1}{1-\alpha} - \frac{p(x)^{1-\alpha} - 1}{1-\alpha} \right] dx.
 \end{aligned} \tag{34}$$

Now setting in (34) $p(x) = \frac{t(x)^{1/\alpha}}{\int t(x)^{1/\alpha} dx}$ where denominator is used to ensure a normalized probability density, gives

$$\begin{aligned}
 D_\alpha(p(x) \parallel q(x)) &= -\frac{1}{\alpha \left[\int t(x)^{1/\alpha} dx \right]^\alpha} \\
 &= \int t(x) \left[\log_\alpha(q(x)) - \log_\alpha \left(\frac{t(x)^{1/\alpha}}{\int t(x)^{1/\alpha} dx} \right) \right] dx.
 \end{aligned} \tag{35}$$

Replacing $t(x)$ with $f(\mathbf{z}, \boldsymbol{\theta}^*)$ and $q(x)$ with $f(\mathbf{z}, \boldsymbol{\theta})$ gives (12)

$$\begin{aligned}
 D_\alpha(\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) \parallel f(\mathbf{z}, \boldsymbol{\theta})) &= -\frac{1}{\alpha \left(\int f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha} d\mathbf{z} \right)^\alpha} \\
 &\quad \int f(\mathbf{z}, \boldsymbol{\theta}^*) \left[\log_\alpha(f(\mathbf{z}, \boldsymbol{\theta})) - \log_\alpha(\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*)) \right] d\mathbf{z}
 \end{aligned}$$

where

$$\tilde{f}(\mathbf{z}, \boldsymbol{\theta}^*) = \frac{f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha}}{\int f(\mathbf{z}, \boldsymbol{\theta}^*)^{1/\alpha} d\mathbf{z}}.$$

REFERENCES

- [1] Z. Zhang *et al.*, "A survey of sparse representation: Algorithms and applications," *IEEE Access*, vol. 3, pp. 490–530, 2015.
- [2] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

- [3] M. Aharon *et al.*, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” *IEEE Transactions on Signal Processing*, vol. 54, pp. 4311–4322, 2006.
- [4] A. Iqbal *et al.*, “Shared and subject-specific dictionary learning algorithm for multi-subject fMRI data analysis (ShSSDL),” *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 11, pp. 2519–2528, Nov 2018.
- [5] A. K. Seghouane and A. Iqbal, “Sequential dictionary learning from correlated data: Application to fMRI data analysis,” *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 3002–3015, 2017.
- [6] A. Iqbal and A. K. Seghouane, “A dictionary learning algorithm for multi-subject fMRI analysis based on a hybrid concatenation scheme,” *Digital Signal Processing*, vol. 83, pp. 249–260, 2018.
- [7] A. K. Seghouane and A. Iqbal, “Basis expansions approaches for regularized sequential dictionary learning algorithms with enforced sparsity for fMRI data analysis,” *IEEE Transactions on Medical Imaging*, vol. 36, no. 9, pp. 1796–1807, 2017.
- [8] Y. Xu *et al.*, “A survey of dictionary learning algorithms for face recognition,” *IEEE Access*, vol. 5, pp. 8502–8514, 2017.
- [9] I. Todic and P. Frossard, “Dictionary learning,” *Signal Processing Magazine, IEEE*, vol. 28, no. 2, pp. 27–38, 2011.
- [10] K. Engan *et al.*, “Method of optimal directions for frame design,” *IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, pp. 2443–2446, 1999.
- [11] J. Liu *et al.*, “A weighted dictionary learning model for denoising images corrupted by mixed noise,” *IEEE Transactions on Image Processing*, vol. 22, no. 3, pp. 1108–1120, 2013.
- [12] P. J. Huber, *Robust Statistics*, ser. Wiley series in probability and statistics. Hoboken, N.J.: Wiley, New York, 2009.
- [13] W. Jiang *et al.*, “Robust dictionary learning with capped l1-norm,” in *International Joint Conference on Artificial Intelligence*, 2015, Conference Proceedings, pp. 3590–3596.
- [14] S. Mukherjee *et al.*, “L1-K-SVD: A robust dictionary learning algorithm with simultaneous update,” *Signal Processing*, vol. 123, pp. 42–52, 2016.
- [15] Q. Pan *et al.*, “Robust non-negative dictionary learning,” in *AAAI*, 2014, Conference Proceedings, pp. 2027–2033.
- [16] C. Lu *et al.*, “Online robust dictionary learning,” in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, Conference Proceedings, pp. 415–422.
- [17] N. Wang *et al.*, “Online robust non-negative dictionary learning for visual tracking,” in *Proceedings of the IEEE international conference on computer vision*, 2013, Conference Proceedings, pp. 657–664.
- [18] C. Zhao *et al.*, “Background subtraction via robust dictionary learning,” *EURASIP Journal on Image and Video Processing*, vol. 2011, no. 1, p. 972961, 2011.
- [19] S. P. Kasiviswanathan *et al.*, “Online l1-dictionary learning with application to novel document detection,” in *Advances in Neural Information Processing Systems*, 2012, Conference Proceedings, pp. 2258–2266.
- [20] J.-J. Fuchs, “An inverse problem approach to robust regression,” in *icassp. IEEE*, 1999, Conference Proceedings, pp. 1809–1812.
- [21] G. Mateos and G. B. Giannakis, “Robust pca as bilinear decomposition with outlier-sparsity regularization,” *IEEE Transactions on Signal Processing*, vol. 60, no. 10, pp. 5176–5190, 2012.
- [22] P. A. Forero *et al.*, “Sparsity-driven laplacian-regularized outlier identification for dictionary learning,” *IEEE Transactions on Signal Processing*, vol. 65, no. 14, pp. 3803–3817, 2017.
- [23] A. Cichocki and S. I. Amari, “Families of alpha-Beta and Gamma-divergences: Flexible and robust measures of similarities,” *Entropy*, vol. 12, 2010.
- [24] R. Maronna *et al.*, *Robust Statistics: Theory and Methods*. Wiley, 2006.
- [25] A. K. Seghouane and M. Hanif, “A sequential dictionary learning algorithm with enforced sparsity,” *IEEE International Conference on Acoustic Speech and signal Processing, ICASSP*, pp. 3876–3880, 2015.
- [26] M. Sadeghi *et al.*, “Learning overcomplete dictionaries based on atom by atom updating,” *IEEE Transactions on Signal Processing*, vol. 62, pp. 883–891, 2014.
- [27] S. Ravishanker *et al.*, “Efficient sum of outer products dictionary learning (SOUP-DIL) and its application to inverse problems,” *IEEE Transactions on Computational Imaging*, pp. 1–16, 2017.
- [28] A. K. Seghouane and A. Iqbal, “Consistent adaptive sequential dictionary learning,” *Signal Processing*, vol. 99, pp. 2055–2065, 2018.
- [29] A. Iqbal and A. K. Seghouane, “Robust dictionary learning using α -divergence,” *IEEE International Conference on Acoustic Speech and signal Processing, ICASSP*, pp. 2972–2976, 2019.
- [30] H. Lee *et al.*, “Efficient sparse coding algorithms,” *Advances in Neural Information Processing Systems*, vol. 19, pp. 801–808, 2007.
- [31] M. S. Lewicki and T. J. Sejnowski, “Learning overcomplete representations,” *Neural Computation*, vol. 12, pp. 337–365, 2000.
- [32] S. Lesage *et al.*, “Learning unions of orthonormal bases with thresholded singular value decomposition,” in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing*, pp. 293–296, 2005.
- [33] J. Mairal *et al.*, *Online Learning for Matrix Factorization and Sparse Coding*. JMLR.org, 2010, pp. 19–60.
- [34] D. Barchiesi and M. D. Plumbley, “Learning incoherent dictionaries for sparse approximation using iterative projections and rotations,” *IEEE Transactions on Signal Processing*, vol. 61, pp. 2055–2065, 2013.
- [35] S. Ubaru *et al.*, “Improving the incoherence of learned dictionary via rank shrinkage,” *Neural Computation*, vol. 29, pp. 263–285, 2017.
- [36] J. Gorski *et al.*, “Biconvex sets and optimization with biconvex functions: a survey and extensions,” *Math. Meth. Oper. Res.*, vol. 66, pp. 373–407, 2007.
- [37] M. R. Osborne *et al.*, “A new approach to variable selection in least squares problems,” *IMA Journal of Numerical Analysis*, vol. 20, pp. 389–403, 2000.
- [38] W. Fu, “Penalized regressions: The bridge versus the Lasso,” *Journal of Computational and Graphical Statistics*, vol. 7, pp. 397–416, 1998.
- [39] J. Friedman *et al.*, “Pathwise coordinate optimization,” *The Annals of Applied Statistics*, vol. 1, pp. 302–322, 2007.
- [40] G. H. Golub and C. f. Van Loan, *Matrix Computations*. Johns Hopkins, 1996.
- [41] V. P. Godambe, *Estimating Functions*. Oxford University Press, 1991.
- [42] A. W. V. der Vaart, *Asymptotic statistics*. Cambridge university press, 2000.
- [43] S. I. Amari, *Differential-Geometrical Methods in Statistics*. Springer Verlag, 1985.
- [44] “Usps handwritten digits dataset.” [Online]. Available: <https://cs.nyu.edu/~roweis/data.html>
- [45] Y. LeCun *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [46] J. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *Information Theory, IEEE Transactions on*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [47] “UCSD anomaly detection dataset.” [Online]. Available: <http://www.svcl.ucsd.edu/projects/anomaly/>