DECONET: an Unfolding Network for Analysis-based Compressed Sensing with Generalization Error Estimates

Vicky Kouni and Yannis Panagakis

Abstract—We present a new deep unfolding network for analysis-sparsity-based Compressed Sensing. The proposed network coined Decoding Network (DECONET) jointly learns a decoder that reconstructs vectors from their incomplete, noisy measurements and a redundant sparsifying analysis operator, which is shared across the layers of DECONET. Moreover, we formulate the hypothesis class of DECONET and estimate its associated Rademacher complexity. Then, we use this estimate to deliver meaningful upper bounds for the generalization error of DECONET. Finally, the validity of our theoretical results is assessed and comparisons to state-of-the-art unfolding networks are made, on both synthetic and real-world datasets. Experimental results indicate that our proposed network outperforms the baselines, consistently for all datasets, and its behaviour complies with our theoretical findings.

Index Terms—Compressed Sensing, analysis sparsity, unfolding network, generalization error, Rademacher complexity.

I. INTRODUCTION

Compressed Sensing (CS) [1] is a modern technique to reconstruct signals $x \in \mathbb{R}^n$ from few linear and possibly corrupted observations $y = Ax + e \in \mathbb{R}^m$, $m < n$. Iterative methods applied on CS are by now widely used [2, 3, 4]. Nevertheless, deep neural networks (DNNs) have become very popular for tackling sparse recovery problems like CS [5, 6], since they significantly reduce the time complexity and increase the quality of the reconstruction. A new line of research lies on merging DNNs and optimization algorithms, leading to the so-called deep unfolding/unrolling [7, 8]. The latter pertains to interpreting the iterations of well-known iterative algorithms as layers of a DNN, which reconstructs $x$ from $y$.

Deep unfolding networks have become increasingly popular in the last few years [9], [10], [11], because – in contrast with traditional DNNs – they are interpretable, integrate prior knowledge about the signal structure [12], and have relatively few trainable parameters [13]. Especially in the case of CS, many unfolding networks have proven to work particularly well. For example, [14], [15], [16], [17], [18], [19], [20] interpret the iterations of well-studied optimization algorithms as layers of a neural network, which learns a decoder for CS, i.e., a function that reconstructs $x$ from $y$. Additionally, some of these networks jointly learn a sparsifying transform for $x$. This sparsifier may either be a nonlinear transform [16] or an orthogonal matrix [20] – integrating that way a dictionary learning technique. The latter has shown promising results when employed in model-based CS [21, 22, 23]; hence, it looks appealing to combine it with unfolding networks. Furthermore, research community focuses lately on the generalization error [24, 25] of deep unfolding networks [20]. Despite recent results of this kind, estimating the generalization error of unfolding networks is still in its infancy.

In fact, generalization error bounds are only provided for unfolding networks that promote synthesis sparsity (by means of the dictionary learning technique) in CS. On the other hand, analysis sparsity differs significantly [26] and it is more advantageous for CS [27], since it models sparse signals more flexibly, by leveraging the redundancy of the involved analysis operators (cf. Section II). To the best of our knowledge, only one unfolding network takes advantage of analysis sparsity [28] (in terms of analysis operator learning [29]), but its generalization ability is not mathematically explained.

In this paper, we are inspired by the articles [18], [20], [28]. These publications propose ISTA-based [2] and ADMM-based [4] unfolding networks, which jointly learn a decoder for CS and a sparsifying transform. Particularly, the learnable sparsifiers of [18, 20] promote synthesis sparsity, while [28] employs its handier analysis counterpart. The deficiency of [18, 28] lies on the fact that their proposed frameworks are not accompanied by a generalization analysis, whereas [20] provides generalization error bounds for the proposed network. Similarly, we develop a new unfolding network based on an optimal analysis-$l_1$ algorithm [30] and call it Decoding Network (DECONET). The latter jointly learns a decoder for CS and a redundant sparsifying analysis operator; thus, we address the CS problem under the analysis sparsity model. Our novelty lies on estimating the generalization error of the proposed analysis-based unfolding network. To that end, we upper bound the generalization error of DECONET in terms of the Rademacher complexity [31] of the associated hypothesis class. In the end, we numerically test the validity of our theory and compare our proposed network to the state-of-the-art (SotA) unfolding networks of [20] and [28], on real-world and synthetic data. In all datasets, our proposed neural architecture outperforms the baselines in terms of generalization error, which scales in accordance with our theoretical results.
Our key contributions are listed below.

1) After differentiating synthesis from analysis sparsity in CS and presenting example unfolding networks in Section II, we develop a new unfolding network dubbed DECONET. The latter jointly learns a decoder that solves the analysis-based CS problem and a redundant sparsifying analysis operator \( W \in \mathbb{R}^{N \times n} \) (\( n < N \)), that is shared across the layers of DECONET.

2) We introduce in Section III the hypothesis class – parameterized by \( W \) – of all the decoders DECONET can realize and restrict \( W \) to be bounded in this class, so that we impose a realistic structural constraint on the operator.

3) Later in Section III, we estimate the generalization error of DECONET using a chaining technique. Our results showcase that the redundancy \( N \) of \( W \) and the number of layers \( L \) affect the generalization ability of DECONET; roughly speaking, the generalization error scales like \( \sqrt{NL} \) (see Theorem III.12 and Corollary III.13). To the best of our knowledge, we are the first to study the generalization ability of an unfolding network for analysis-based CS.

4) We confirm the validity of our theoretical guarantees in Section IV, by testing DECONET on a synthetic dataset and two real-world image datasets (MNIST [32] and CIFAR10 [33]). We also compare DECONET to two SotA unfolding networks: a recent variant of ISTA-net [20] and ADMM-DAD net [28]. Our experiments demonstrate that a) the generalization error of DECONET scales correctly with our theoretical findings b) DECONET outperforms both baseline, consistently for all datasets.

**Notation.** We denote the set of real, positive numbers by \( \mathbb{R}_+ \). For a sequence \( \{a_n\} \) that is upper bounded by \( M > 0 \), we write \( \{a_n\} \leq M \). For a matrix \( A \in \mathbb{R}^{n \times n} \), we write \( \|A\|_{\text{Frobenius}} \) for its Frobenius norm. Moreover, we write \( \|A\|_{2 \rightarrow 2} \approx 1 \) if \( \|A\|_{2 \rightarrow 2} \leq 1 \), but there exists \( C > 0 \) such that \( C\|A\|_{2 \rightarrow 2} \leq 1 \). For a family of vectors \( \{w_i\}_{i=1}^N \in \mathbb{R}^n \), its associated analysis operator is given by \( Wf := \{f(w_i)\}_{i=1}^N \), where \( f \in \mathbb{R}^n \). For matrices \( A_1, A_2 \in \mathbb{R}^{N \times N} \), we denote by \( [A_1: A_2] \in \mathbb{R}^{2N \times N} \) their concatenation with respect to the first dimension, while we denote by \( [A_1 | A_2] \in \mathbb{R}^{N \times 2N} \) their concatenation with respect to the second dimension. We write \( O_{N \times N} \) for a real-valued \( N \times N \) matrix filled with zeros and \( I_{N \times N} \) for the \( N \times N \) identity matrix. We denote by \( \text{diag}(\alpha) \) a square diagonal matrix having \( \alpha \in \mathbb{R} \) in its main diagonal and zero elsewhere. For \( x \in \mathbb{R} \), \( \tau > 0 \), the soft thresholding operator \( S_\tau : \mathbb{R} \to \mathbb{R} \) is defined as

\[
S_\tau(x) = S(x, \tau) = \begin{cases} \text{sign}(x)(|x| - \tau), & |x| \geq \tau \\ 0, & \text{otherwise} \end{cases}
\]

or in closed form \( S(x, \tau) = \text{sign}(x) \max(0, |x| - \tau) \). For \( x \in \mathbb{R}^n \), the soft thresholding operator acts componentwise, i.e. \( (S_\tau(x_i))_i = S_\tau(x_i) \). For \( y \in \mathbb{R}^n \), \( \tau > 0 \), the mapping

\[
P_G(\tau; y) = \underset{x \in \mathbb{R}^n}{\text{argmin}} \left\{ \tau G(x) + \frac{1}{2} \|x - y\|_2^2 \right\},
\]

is the proximal mapping associated to the convex function \( G \).

For \( G(\cdot) = \| \cdot \|_1 \), (2) coincides with (1). For \( x \in \mathbb{R} \), \( \tau > 0 \), the truncation operator \( T_\tau : \mathbb{R} \to \mathbb{R} \) is defined as

\[
T_\tau(x) = \text{sign}(x) \min\{|x|, \tau\} = \begin{cases} r\text{sign}(x), & |x| \geq \tau \\ x, & \text{otherwise} \end{cases}
\]

For \( x \in \mathbb{R}^n \), the truncation operator acts componentwise and is 1-Lipschitz. For two functions \( f, g : \mathbb{R}^n \to \mathbb{R}^n \), we write their composition as \( f \circ g : \mathbb{R}^n \to \mathbb{R}^n \) and if there exists some constant \( C > 0 \) such that \( f(x) \leq Cg(x) \), then we write \( f(x) \lesssim g(x) \).

A. Synthesis Sparsity in CS and ISTA-based Unfolding

Under the synthesis sparsity model [36, 38, 39, 40], signals are considered to be sparse when synthesized by a few column vectors taken from a large matrix (dictionary), which is
typically assumed to be orthogonal, i.e. $D \in \mathbb{R}^{n \times n}$, with $DD^T = I_{n \times n}$ (e.g. $D$ may be the discrete cosine transform), so that $x = Dz$. Now, employing synthesis sparsity in CS, we aim to recover $x$ from $y$. A common way to do so is by solving the $l_1$-minimization problem

$$
\min_{z \in \mathbb{R}^n} \|z\|_1 \quad \text{s. t.} \quad \|y - ADz\|_2 \leq \varepsilon. \tag{4}
$$

Towards that end, numerous iterative algorithms [2, 3, 41] have emerged. Typically, they consist of an iterative scheme that incorporates a proximal mapping and after a number of iterations and under certain conditions, they converge to a minimizer $\hat{z}$ of (4). For example, ISTA uses the proximal mapping\(^1\) (2) to yield the following iterative scheme

$$
z_{k+1} = S_\tau(z_k + \tau(AD)^T(y - ADz)) = S_\tau((I - D^TAD)z + \tau(AD)^Ty), \tag{5}
$$

for $k = 0, 1, \ldots, z_0 = 0$, with $\tau, \lambda > 0$ being parameters of the algorithm. If $\tau \|AD\|_2^2 \lambda < 1$ [2], $z_k$ converges to a minimizer $\hat{z}$ of (4), so that the reconstructed $\hat{z}$ is simply given by $\hat{x} = D\hat{z}$. As stated in [20], under the assumption that $D$ is learned from a training set, the iterative scheme of (5) can be interpreted as a layer of a neural network (whose trainable parameters are the entries of $D$) with weight matrix $I - D^TAD$, bias $\tau(AD)^Ty$ and activation function $S_\tau$. Then, the composition of a given number of layers and the consequent application of $D$ constitutes the decoder implemented by the ISTA-based network, which outputs $\hat{x} \approx x$.

B. Analysis Sparsity in CS and ADMM-based Unfolding

Despite its success, synthesis sparsity has a “twin”, i.e., the analysis sparsity model [42, 43, 44], in which one assumes that there exists a redundant analysis operator $W \in \mathbb{R}^{N \times n}$ ($N > n$), so that $WX$ is sparse. For example, $W$ may be the analysis operator associated to a frame [45, 46] or a finite difference operator [47]. The associated optimization problem for CS is the analysis $l_1$-minimization problem

$$
\min_{x \in \mathbb{R}^n} \|WX\|_1 \quad \text{s. t.} \quad \|Ax - y\|_2 \leq \varepsilon. \tag{6}
$$

From now on, whenever we speak about the redundancy of an analysis operator, we mean the number of its rows $N$.

Analysis sparsity has become popular, due to some benefits it has compared to its synthesis counterpart. For example, it is computationally more appealing to solve the optimization algorithm of analysis CS, since the actual optimization takes place in the ambient space [48] and the algorithm involved may need less measurements for perfect reconstruction, if one uses a redundant transform instead of an orthogonal one [43]. Nevertheless, choosing the appropriate iterative algorithm for solving (6) may be a tricky task. The reason is that most thresholding algorithms cannot handle analysis sparsity, since the proximal mapping associated to $\|W(\cdot)\|_1$ does not have a closed-form type. To tackle this issue and solve (6), one may employ the so-called ADMM [4] algorithm, which uses the following iterative scheme

$$
x^{k+1} = (A^T A + \rho W^T W)^{-1} (A^T y + \rho W^T(z^k - u^k)) \tag{7}
$$

$$
z^{k+1} = S_{\lambda/\rho}(Wx^{k+1} - u^k) \tag{8}
$$

$$
u^{k+1} = u^k + Wx^{k+1} - z^{k+1}, \tag{9}
$$

with $k \in \mathbb{N}$, dual variables $z, u \in \mathbb{R}^N$, initial points $(x^0, z^0, u^0) = (0, 0, 0)$, penalty parameter $\rho > 0$ and regularization parameter $\lambda > 0$. As shown in [4], the iterates (7) - (9) converge to a solution $p^*$ of

$$
\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - y\|_2^2 + \lambda \|z\|_1 \quad \text{s. t.} \quad Wx - z = 0. \tag{10}
$$

In [28], the updates (7) - (9) are formulated as a neural network (whose trainable parameters are the entries of $W$) with $L$ layers, defined as

$$
f_1(y) = I_1 b(y) + I_2 S_{\lambda/\rho}(b(y)), \tag{11}
$$

$$
f_k(v) = \tilde{\Theta} v + I_1 b + I_2 S_{\lambda/\rho}(\tilde{\Theta} v + b), \quad k = 2, \ldots, L, \tag{12}
$$

with $v \in \mathbb{R}^{2N \times 1}$ being an intermediate variable, weight matrices $\tilde{\Theta}, \Theta$ (depending on $A, W, \rho$), bias $b$ (depending on $A, W, \rho, y$) and activation function $S_{\lambda/\rho}$. The resulting learned decoder that reconstructs $x$ from $y$ emerges by applying an affine map on the composition of a given number of layers.

C. Towards a New Unfolding Network for Analysis CS

Although the generic ADMM-based unfolding network of [28] seems promising, it involves the costly computation of $(A^T A + \rho W^T W)^{-1}$. Therefore, we opt for solving (6) with the optimal first-order analysis-$l_1$ algorithm described in [30], which uses transposes – instead of inverses – of the involved matrices. We will briefly describe the steps leading to the derivation of the aforementioned algorithm, as these are stated in [30]. First, an equivalent to (6) smoothed formulation is given as follows:

$$
\min_{x \in \mathbb{R}^n} \|WX\|_1 + \frac{\mu}{2} \|x - x_0\|_2^2 \quad \text{s. t.} \quad \|Ax - y\|_2 \leq \varepsilon, \tag{13}
$$

where $\mu \in \mathbb{R}_+$ is the so-called smoothing parameter and $x_0 \in \mathbb{R}^n$ is an initial guess on $x$. Second, the dual of (13) is determined to be

$$
\text{maximize} \quad \langle y, z^2 \rangle - \varepsilon \|z^2\|_2 \quad \text{s. t.} \quad A^T z^2 - W^T z^1 = 0, \quad \|z^1\|_\infty \leq 1, \tag{14}
$$

where $z^1 \in \mathbb{R}^N$, $z^2 \in \mathbb{R}^m$ are dual variables. The aforementioned formulations – combined with a collection of arguments and computations – lead to Algorithm 1, which constitutes a variant of an optimal first-order method. For reasons of convenience, we call this algorithm analysis conic form (ACF) from now on. ACF also involves step sizes $\{t_k^1\}, \{t_k^2\} > 0$ and a step size multiplier $0 < \{\theta_k\}$. A standard setup for ACF employs update rules such that $0 < \{t_k^1\}_{k \geq 0}$, $\{t_k^2\}_{k \geq 0} \leq 1$, $0 \{\theta_k\}_{k \geq 0} \leq 1$.

The dual function $g_y$ corresponding to (14) has a Lipschitz continuous gradient, hence ACF converges [30] to a solution $\hat{x}_\mu$ of (13), for which we have $\hat{x}_\mu \approx x$, where $\hat{x}$ is an optimal solution of (6). The authors of [30] clarify that when

\(^1\)We remind that the proximal mapping with respect to $\|\cdot\|_1$ coincides to the soft thresholding operator (1)
Algorithm 1: ACF

\[ \begin{aligned}
\text{Input: } x_0 &\in \mathbb{R}^n, z_0^1 \in \mathbb{R}^N, z_0^2 \in \mathbb{R}^m, \mu \in \mathbb{R}_+, \text{ step sizes} \left\{ t_k^1 \right\}, \left\{ t_k^2 \right\} \\
\text{Output: solution } \hat{x}_p \text{ of (13)} \\
n &\left \{ \begin{array}{l}
0 \\
\text{for iterations } k = 0, 1, \ldots \text{ do }
\end{array} \right.
\end{aligned} \]

\[ \begin{aligned}
x_k &\leftarrow x_0 + \mu \cdot (1 - \theta_k) W^T u_k^1 + \theta_k W z_k^2 - (1 - \theta_k) A \hat{u}_k^1 - \theta_k A \hat{z}_k^2 \\
z_{k+1}^1 &\leftarrow T((1 - \theta_k) u_k^1 + \theta_k z_k^1 - \theta_k - t_k^1 W x_k) + \theta_k (1 - t_k^1) \varepsilon \\
z_{k+1}^2 &\leftarrow S((1 - \theta_k) u_k^2 + \theta_k z_k^2 - \theta_k - t_k^2 (y - Ax_k)) + \theta_k (1 - t_k^2) \varepsilon \\
u_k^1 &\leftarrow (1 - \theta_k) u_k^1 + \theta_k z_{k+1}^1 \\
u_k^2 &\leftarrow (1 - \theta_k) u_k^2 + \theta_k z_{k+1}^2 \\
\theta_{k+1} &\leftarrow 2 / (1 + (1 + \langle \theta_k \rangle^2)^{1/2})
\end{aligned} \]

We observe that (15) can be interpreted as a layer of a neural network, with weights \( G^1, G^2 \), biases \( b^1, b^2 \) and activation functions \( T, S \). Nevertheless, this interpretation of ACF as a DNN does not account for any trainable parameters. We cope with this issue by considering \( W \) to be unknown and learned from a training sequence \( S = \{(x_i, y_i)\} \) with i.i.d. samples drawn from an unknown distribution \( D^\ast \). Hence, the trainable parameters are the entries of \( W \). Additionally, we make the realistic assumption that \( W \) is bounded with respect to the operator norm, i.e., \( W \in B_\lambda \), for some \( \lambda > 0 \). Now, based on (15), we formulate ACF as a neural network with \( L \) layers/iterations, defined as

\[ \begin{aligned}
f_1(x) &= \sigma(y) \\
f_k(v) &= D_{k-1} v + \Theta_{k-1} \sigma(v), \quad k = 2, \ldots, L,
\end{aligned} \]

where

\[ \begin{aligned}
\sigma(y) &= (T(-t_k^0 W x_0, t_k^0), S(t_k^0 (y - Ax_0), t_k^0)), \\
T(-t_k^0 W x_0, t_k^0), S(t_k^0 (y - Ax_0), t_k^0))^T, \\
\sigma(v) &= (T(G_k^1 v - b_k^1), S(G_k^2 v - b_k^2), \\
T(G_k^1 v - b_k^1, S(G_k^2 v - b_k^2))^T,
\end{aligned} \]

for \( k = 2, \ldots, L \). We denote the composition of \( L \) such layers (all having the same \( W \)) as

\[ f_L^L(y) = f_L \circ f_{L - 1} \circ \cdots \circ f_1(y). \]

The latter constitutes the realization of a neural network with \( L \) layers, that reconstructs the intermediate variable \( v \) from \( y \). Thus, we call (26) intermediate decoder. In order to get the solution \( \hat{x} \), we apply an affine map \( \Phi : \mathbb{R}^{(2N+2m) \times 1} \rightarrow \mathbb{R}^{1 \times 1} \) motivated by the \( x \)-update of ACF – after the last layer, \( L \), so that \( \hat{x} := \phi(v) = \Phi u + x_0 \), where

\[ \Phi = (\mu^{-1} \theta_L W^T | - \mu^{-1} \theta_L A^T | \mu^{-1} (1 - \theta_L) W^T | - \mu^{-1} (1 - \theta_L) A^T) \in \mathbb{R}^{1 \times (2N+2m)} \]

Moreover, in order to clip the output \( \phi(f_L^L(y)) \) in case its norm falls out of a reasonable range, we apply an extra function \( \psi : \mathbb{R}^n \rightarrow \mathbb{R}^n \) after \( \phi \) and define it as

\[ \psi(x) = \left\{ \begin{array}{l}
x, \\
B_{\text{out}} \frac{x}{\|x\|_2}, \quad \|x\|_2 \leq B_{\text{out}} \\
\text{otherwise}
\end{array} \right. \]

for some constant \( B_{\text{out}} > 0 \). Now, for a fixed number of layers \( L \), the desired learnable decoder is written as

\[ \text{dec}_W^L(y) = \psi(\phi(f_L^L(y))). \]

We call Decoding Network (DECONET) the network that implements such a decoder, which is parameterized by \( W \).

III. THEORETICAL RESULTS

In this Section, we deliver meaningful – in terms of \( L \) and \( N \) – upper bounds on the generalization error of DECONET. We do so in a series of steps presented in the next subsections.

2 Formally speaking, this is a distribution over the \( x_i \) and then \( y_i = Ax_i + \varepsilon \), with fixed \( A, \varepsilon \).
A. Hypothesis Class of DECONET and Associated Rademacher Complexity

We introduce the hypothesis class
\[
\mathcal{H}^L = \{ h : \mathbb{R}^m \to \mathbb{R}^n : h(y) = \psi(f_W(\phi(y))), W \in \mathcal{B}_\Lambda \},
\]
parametrized by \( W \) and consisting of all the functions/decoders DECONET can implement. Given (30) and the training set \( \mathcal{S} \), DECONET yields a function \( h_{\mathcal{S}} \in \mathcal{H}^L \) that aims at reconstructing \( x \) from \( y \). For a loss function \( \ell : \mathcal{H}^L \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_+ \), the empirical loss of a hypothesis \( h \in \mathcal{H}^L \) is the reconstruction error on the training set, i.e.
\[
\hat{L}_{\text{train}}(h) = \frac{1}{s} \sum_{i=1}^{s} \ell(h, x_i, y_i).
\]
In this paper, we choose as loss function \( \ell \) the squared \( l_2 \)-norm; hence, (31) takes the form of the training mean-squared error (MSE):
\[
\hat{L}_{\text{train}}(h) = \frac{1}{s} \sum_{j=1}^{s} \|h(y_j) - x_j\|_2^2.
\]
The true loss is
\[
L(h) = \mathbb{E}_{(x,y) \sim \mathcal{D}}(\|h(y) - x\|_2^2).
\]
The generalization error is given as the difference\(^3\) between the empirical and true loss
\[
\text{GE}(h) = |\hat{L}_{\text{train}}(h) - L(h)|.
\]
A typical way to estimate (34) consists in upper bounding it in terms of the Rademacher complexity. The empirical Rademacher complexity is defined as
\[
\mathcal{R}_S(\ell \circ \mathcal{H}^L) = \mathbb{E} \sup_{h \in \mathcal{H}^L} \frac{1}{s} \sum_{i=1}^{s} \epsilon_i \|h(y_i) - x_i\|_2^2,
\]
where \( \epsilon \) is a Rademacher vector, that is, a vector with entries taking the values \( \pm 1 \) with equal probability. Then, the Rademacher complexity is defined as
\[
\mathcal{R}_s(\ell \circ \mathcal{H}^L) = \mathbb{E}_{s \sim \mathcal{D}^s}(\mathcal{R}_S(\ell \circ \mathcal{H}^L)).
\]
In this paper, we solely work with (35). We rely on the following Theorem that estimates (34) in terms of (35).

**Theorem III.1** ([24]). Let \( \mathcal{H} \) be a family of functions, \( \mathcal{S} \) the training set drawn from \( \mathcal{D}^s \), and \( \ell \) a real-valued bounded loss function satisfying \( |\ell(h, z)| \leq c \), for all \( h \in \mathcal{H}, z \in Z \). Then, for \( \delta \in (0, 1) \), with probability at least \( 1 - \delta \), we have for all \( h \in \mathcal{H} \)
\[
L(h) \leq \hat{L}(h) + 2\mathcal{R}_S(\ell \circ \mathcal{H}^L) + 4c \sqrt{\frac{2 \log(4d/\delta)}{s}},
\]
In order to use the previous Theorem, the loss function must be bounded. Towards this end, we make two typical (for the machine learning literature) assumptions regarding the training set \( \mathcal{S} = \{(x_i, y_i)\}_{i=1}^{s} \). Let us suppose that with overwhelming probability we have
\[
\|y_i\|_2 \leq B_{in},
\]
for some constant \( B_{in} > 0 \), \( i = 1, \ldots, s \). Moreover, we assume that for any \( h \in \mathcal{H}^L \), with overwhelming probability over \( y_i \) chosen from \( \mathcal{D} \), the following holds
\[
\|h(y_i)\|_2 \leq B_{out},
\]
by definition of \( \psi \), for some constant \( B_{out} > 0 \), for all \( i = 1, \ldots, s \). Hence, the loss function is bounded as \( \|h(y_i) - x_i\|_2^2 \leq (B_{in} + B_{out})^2 \), for all \( i = 1, \ldots, s \). Following the previous assumptions, it is easy to check that \( \|\cdot\|_2^2 \) is a Lipschitz continuous function, with Lipschitz constant \( \text{Lip}_{\|\cdot\|_2^2} = 2B_{in} + 2B_{out} \). Since \( \|\cdot\|_2^2 \) is Lipschitz, we can employ the so-called (vector-valued) contraction principle, which allows us to study \( \mathcal{R}_S(\ell \circ \mathcal{H}) \) alone:

**Lemma III.2** ([49]). Let \( \mathcal{H} \) be a set of functions \( h : \mathcal{X} \to \mathbb{R}^n \), \( f : \mathbb{R}^n \to \mathbb{R}^n \) a \( K \)-Lipschitz function and \( \mathcal{S} = \{x_i\}_{i=1}^{s} \). Then
\[
\mathbb{E} \sup_{h \in \mathcal{H}} \sum_{i=1}^{s} \epsilon_i f \circ h(x_i) \leq \sqrt{2K} \mathbb{E} \sup_{h \in \mathcal{H}} \sum_{i=1}^{s} \sum_{k=1}^{n} \epsilon_{ik} h_k(x_i),
\]
where \( (\epsilon_i), (\epsilon_{ik}) \) are both Rademacher sequences.

Applying Lemma III.2 in \( \mathcal{R}_S(\ell \circ \mathcal{H}) \) yields:
\[
\mathcal{R}_S(\ell \circ \mathcal{H}^L) \leq \sqrt{2\text{Lip}_{\|\cdot\|_2^2}^2} \mathcal{R}_S(\mathcal{H}^L) = \sqrt{2\text{Lip}_{\|\cdot\|_2^2}^2} \mathbb{E} \sup_{h \in \mathcal{H}^L} \sum_{i=1}^{s} \sum_{k=1}^{n} \epsilon_{ik} h_k(x_i) = B_{\text{out}}^\text{mat} \mathbb{E} \sup_{h \in \mathcal{H}^L} \sum_{i=1}^{s} \sum_{k=1}^{n} \epsilon_{ik} h_k(x_i),
\]
where \( B_{\text{out}}^\text{mat} = \sqrt{2(2B_{in} + 2B_{out})} \). We return to estimating (41) in Section III-D, after presenting the adequate mathematical tools in Sections III-B and III-C.

B. Boundedness of DECONET’s Outputs

We now take into account the number of training samples and pass to matrix notation. Due to (38), (39) and the Cauchy-Schwartz inequality, we get \( \|Y\|_F \leq \sqrt{s}B_{in} \) and
\[
\|h(Y)\|_F = \|\psi(f_W(\phi(Y)))\|_F \leq \sqrt{s}B_{out}.
\]
The next Lemma presents bounds on a quantity we will encounter more often in the sequel.

**Lemma III.3.** (Proof in the supplementary material) Let \( k \geq 0 \). For any \( W \in \mathcal{B}_\Lambda \), step sizes \( 0 < \{\tau_k\}, \{\tau_k^0\} \leq 1 \) with \( \tau_k^0 = \tau_k^0 \), \( \text{step size multiplier} \ \theta < 1 \) \( \text{with} \ \theta = 1 \) and smoothing parameter \( \mu > 1 \), the following holds for the matrices \( G^L_k, G^2_k \) defined in (18), (19), respectively:
\[
2\|G^L_k\|_{2 \rightarrow 2} + 2\|G^2_k\|_{2 \rightarrow 2} + 1 \leq \Gamma_k,
\]
where
\[
\Gamma_k = 2\left[c_{1,k}A^2 + c_{2,k}\|A\|_{2 \rightarrow 2} + 2\|A\|_{2 \rightarrow 2}A(c_{1,k} + c_{2,k})\right] + 1,
\]
with \( c_{k,1} = \{\theta_k^{-1} - 1\} \leq 1 \), \( c_{k,2} = \{\theta_k^{-1} - 1\} \leq 1 \), for all \( k \geq 0 \). Moreover, if \( c_{1,k}A \leq 1 \), \( c_{1,k}A^2 \leq 1 \), and
1. \(c_2,k\|A\|_{2\to 2}^2 \leq 1\), then \(\Gamma_k \leq \gamma\) for all \(k \geq 0\), with 
\[\gamma = 4(\Lambda + \|A\|_{2\to 2} + 1).\]

**Remark III.4.** In the previous Lemma, we made two simplifying yet reasonable assumptions, which we will keep for the rest of the paper. First, based on the assumptions we made for \(A\) in Section II-C, it is straightforward that \(c_2,k\|A\|_{2\to 2} \leq 1\), for all \(k \geq 0\). Second, there exist redundant analysis operators \(W\) for which \(\Lambda\) may be relatively small [46], [50], so that \(c_1,k\Lambda \leq 1\) and \(c_1,k\Lambda^2 \leq 1\), for all \(k \geq 0\).

Apart from (42), we can upper-bound the output \(f^{k}_W(Y)\) with respect to the Frobenius norm, after any number of layers \(k\) and especially for \(k < L\), so that \(\phi\) and \(\psi\) are not applied after the final layer \(L\).

**Lemma III.5.** Let \(k \in \mathbb{N}\). For any \(W \in B_{\Lambda}\), step sizes \(0 \in \{f^{1}_k\}, \{f^{2}_k\} \leq 1\) with \(t^{0}_f = t^{0}_G = 1, t^{1}_f = t^{2}_G = 0\), step size multiplier \(0 \in \{\theta_k\} \leq 1\) with \(\theta = \theta_{-1} = 1\), and smoothing parameter \(\mu > 1\), the following holds for the output of the functions \(f^{k}_W\) defined in (22) - (23):

\[\|f^{k}_W(Y)\|_F \leq 2\mu\|Y\|_F \sum_{i=0}^{k-1} \left( \binom{\|A\|_{2\to 2}(c_1,k-1)\Lambda + c_2,i-1\|A\|_{2\to 2} + c_2,i-1}{k-1} \prod_{j=0}^{k-1} \Gamma_j \right) + 2\mu\|Y\|_F \]

where \(\{\Gamma_j\}_{k \geq 0}, \{c_1,k\}_{k \geq 0}, \{c_2,k\}_{k \geq 0} \leq 1\) are defined as in Lemma III.3 and \(c_1,k-1 = c_2,-1 = 0\). Moreover, if \(c_1,k\Lambda \leq 1, c_1,k\Lambda^2 \leq 1, c_2,k\|A\|_{2\to 2} \leq 1\), then we have the simplified upper bound

\[\|f^{k}_W(Y)\|_F \leq 2\mu\|Y\|_F(\|A\|_{2\to 2} + 1)\zeta_k + 1),\]

where \(\zeta_k = \frac{k-1}{\gamma - 1}\), with \(\gamma\) defined as in Lemma III.3.

**Proof.** By definition, \(T(\cdot)\) and \(S(\cdot)\) are 1-Lipschitz functions, and for the matrices \(\Theta_k\) and \(D_k\) in (16) and (17) respectively, we have \(\|D_k\|_{2\to 2} \leq 1\) and \(\|\Theta_k\|_{2\to 2} = 1\), for any \(k \geq 1\), since \(\theta \in \{\theta_k\} \leq 1\) and \(\theta_0 = 1\). Now we use the previous statements, along with (22) and (23), to prove (45) via induction. For \(k = 1\), we have

\[\|f^{1}_W(Y)\|_F \leq 2\mu\|Y\|_F(\|A\|_{2\to 2} + 1)(\zeta_k + 1),\]

with \(\zeta_k = \frac{k-1}{\gamma - 1}\).

**C. Lipschitzness Results**

In the next Theorem, we prove that the intermediate decoder (26) is Lipschitz continuous with respect to \(W\) and explicitly calculate the Lipschitz constants, which depend on \(L\).

**Theorem III.6.** (Proof in the supplementary material) Let \(f^{L}_W\) defined as in (26), \(L \geq 2\), dictionary \(W \in B_{\Lambda}\), step sizes \(0 \in \{f^{1}_k\}_{k \geq 0}, \{f^{2}_k\}_{k \geq 0} \leq 1\) with \(t^{0}_f = t^{0}_G = 1, t^{1}_f = t^{2}_G = 0\), step size multiplier \(0 \in \{\theta_k\}_{k \geq 0} \leq 1\) with \(\theta_0 = \theta_{-1} = 1\), and smoothing parameter \(\mu > 1\). Then, for any \(W_1, W_2 \in B_{\Lambda}\), we have

\[\|f^{L}_W(Y) - f^{L}_{W_2}(Y)\|_F \leq K_L\|W_1 - W_2\|_{2\to 2},\]

where \(K_L\) is the Lipschitz constant, which depends on \(L\).
where

$$K_L = 2\mu\|Y\|_F \left( \mu^{-1}\|A\|_{2\rightarrow 2} + \sum_{k=2}^{L} \left( \max_{0 \leq i \leq L-1} \Gamma_i \right) \right)^{L-k} \sum_{i=0}^{k-2} 2 \left( \|A\|_{2\rightarrow 2}(c_{1,1} - 1 + A_{2,1} - 1) + c_{2,1} \right) \prod_{j=i}^{k-2} \Gamma_j + 2 \left( \|A\|_{2\rightarrow 2}(c_{1,2} - 1 + A_{2,1} - 1) + c_{2,2} \right) \cdot (2\Lambda c_{1,1} + \|A\|_{2\rightarrow 2}(c_{1,1} - 1 + c_{2,2} - 1)) + \|A_{2,1} - 1 + c_{1,1} - 1 \right), \quad (48)$$

with \( \{\Gamma_k\}_{k \geq 0}, \{c_{1,k}\}_{k \geq 0}, \{c_{2,k}\}_{k \geq 0} \) defined as in Lemma III.3 and \( c_{1,1} = c_{2,1} = 0 \). Moreover, if \( c_{1,k} \Lambda \leq 1, c_{2,k} \Lambda^2 \leq 1, c_{2,k} \|A\|_{2\rightarrow 2} \leq 1, \) for all \( k \geq 0 \), then we have the simplified upper bound

$$K_L \leq 2\mu\|Y\|_F \left[ \|A\|_{2\rightarrow 2}(L - 1 + \mu^{-1}) + 2(\|A\|_{2\rightarrow 2} + 1)(\|A\|_{2\rightarrow 2} + 3)\kappa_L \right], \quad (49)$$

where

$$\kappa_L = \gamma^L \left( \frac{L - 1}{\gamma - 1} + \frac{(\gamma - 2)^2}{(\gamma - 1)^2} \right) - \gamma^2 \frac{(\gamma - 2)^2}{(\gamma - 1)^2}, \quad (50)$$

with \( \gamma \) as in Lemma III.3.

With the previous Theorem in hand, we prove the Lipschitzness of the main decoder defined in (29).

**Corollary III.7.** Let \( h \in \mathcal{H}_L \) defined as in (30) with \( L \geq 2 \) and dictionary \( W \in \mathcal{B}_L \). Then, for any \( W_1, W_2 \in \mathcal{B}_L \), we have:

$$\|\psi(\phi(f_{W_2}^L(Y))) - \psi(\phi(f_{W_1}^L(Y)))\|_F \leq \mu^{-1}(L + \|A\|_{2\rightarrow 2})K_L\|W_2 - W_1\|_F, \quad (51)$$

with \( K_L \) as in Theorem III.6.

**Proof.** By definition, \( \psi \) is a 1-Lipschitz function. Moreover, as an affine map, \( \phi \) is Lipschitz continuous with Lipschitz constant \( \text{Lip}_\phi = \|\Phi\|_{2\rightarrow 2} \), with \( \Phi \) defined as in (27). We evaluate \( \|\Phi\|_{2\rightarrow 2} \):

$$\|\Phi\|_{2\rightarrow 2} \leq \mu^{-1}\theta_L\|W\|_{2\rightarrow 2} + \mu^{-1}\theta_L\|A\|_{2\rightarrow 2} + \mu^{-1}(1 - \theta_L)\|W\|_{2\rightarrow 2} + \mu^{-1}(1 - \theta_L)\|A\|_{2\rightarrow 2} \leq \mu^{-1}(L + \|A\|_{2\rightarrow 2}).$$

Combining the previous estimate with Theorem III.6, we get

$$\|\psi(\phi(f_{W_2}^L(Y))) - \psi(\phi(f_{W_1}^L(Y)))\|_F \leq \|\phi(f_{W_2}^L(Y)) - \phi(f_{W_1}^L(Y))\|_F \leq \|\Phi\|_{2\rightarrow 2}\|f_{W_2}^L(Y) - f_{W_1}^L(Y)\|_F \leq \mu^{-1}(L + \|A\|_{2\rightarrow 2})K_L\|W_2 - W_1\|_F. \quad \square$$

**D. Covering numbers and Dudley’s inequality**

For a fixed number of layers \( L \in \mathbb{N} \), we define the set \( \mathcal{M} \subset \mathbb{R}^{n \times s} \) corresponding to the hypothesis class \( \mathcal{H}_L \) to be

$$\mathcal{M} : = \{(h(y_1)|h(y_2)| \ldots |h(y_s)) : h \in \mathcal{H}_L\} = \{\psi(\phi(f_W^L(Y))) : W \in \mathcal{B}_L\}. \quad (52)$$

The column elements of each matrix in \( \mathcal{M} \) are the reconstructions given by a decoder \( h \) in \( \mathcal{H}_L \) when applied to the measurements \( y_i \). Since \( \mathcal{M} \) is parameterized by \( W \) like \( \mathcal{H}_L \), we may rewrite (41) as

$$\mathcal{R}_S(l \circ \mathcal{H}_L) \leq B_{\text{out}}^{-1} \mathbb{E} \sup_{\mathcal{M} \in \mathcal{M}} \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{n} \epsilon_{ik}M_{ik}. \quad (53)$$

Thus, we are left with estimating the Rademacher process in the right hand side of (53). The latter has subgaussian increments, hence we use Dudley’s inequality [36] to upper bound it in terms of the covering numbers of \( \mathcal{M} \). Towards that end, we calculate the radius of \( \mathcal{M} \), that is,

$$\Delta(\mathcal{M}) = \sup_{h \in \mathcal{H}_L} \left[ \mathbb{E} \left( \sum_{i=1}^{n} \sum_{k=1}^{n} \epsilon_{ik}h_k(y_i) \right)^2 \right] \leq \sup_{h \in \mathcal{H}_L} \left[ \sum_{i=1}^{n} \|h(y_i)\|_2^2 \right] \leq \frac{1}{\sqrt{2}} B_{\text{out}}. \quad (54)$$

With (54) in hand, applying Dudley’s inequality to (53) yields

$$\mathcal{R}_S(l \circ \mathcal{H}_L) \leq \frac{16(B_{\text{in}} + B_{\text{out}})}{\sqrt{2}} \int_0^{2\pi/\sqrt{\log N(\mathcal{M}, \|\cdot\|_F, \varepsilon)}} \sqrt{\log N(\mathcal{M}, \|\cdot\|_F, \varepsilon)} \, d\varepsilon. \quad (55)$$

Before continuing, we state the following auxiliary Lemma.

**Lemma III.8.** The covering number of \( B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(\Lambda) = \{X \in \mathbb{R}^{n \times n} : \|X\|_{2\rightarrow 2} \leq \Lambda, \Lambda > 0\} \) satisfies the following for any \( \varepsilon > 0 \):

$$N(B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(\Lambda), \|\cdot\|_{2 \rightarrow 2}, \varepsilon) \leq \left( 1 + \frac{2\Lambda}{\varepsilon} \right)^{N_n}. \quad (56)$$

**Proof.** For \( |\cdot| \) denoting the volume in \( \mathbb{R}^{n \times n} \), the following is an adaptation of a well-known result [51], connecting covering numbers and volume in \( \mathbb{R}^{n \times n} \):

$$N(B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(\Lambda), \|\cdot\|_{2 \rightarrow 2}, \varepsilon) \leq \frac{[B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(\Lambda) + (\varepsilon/2)B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(1)]}{|(\varepsilon/2)B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(1)|} = \frac{[(\Lambda + \varepsilon/2)B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(1)]}{|(\varepsilon/2)B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(1)|}. \quad \square$$

Hence,

$$N(B^{n \times n}_{\|\cdot\|_{2 \rightarrow 2}}(\Lambda), \|\cdot\|_{2 \rightarrow 2}, \varepsilon) \leq \left( 1 + \frac{2\Lambda}{\varepsilon} \right)^{N_n}. \quad \square$$

We can now estimate (55):
Proposition III.9. The following estimate holds for the covering numbers of \( \mathcal{M} \):
\[
\mathcal{N}(\mathcal{M}, \| \cdot \|_F, \varepsilon) \leq \left(1 + \frac{2\Lambda + \|A\|_{2 \to 2}K_L}{\mu\varepsilon}\right)^{\eta n}.
\]
(57)

Proof. We first consider the set \( \Omega = \{W : W \in \mathcal{B}_\Lambda\} \subset \mathbb{R}^{N \times n} \). Then, due to Lemma III.8, we can upper bound the covering numbers of \( \Omega \) as follows:
\[
\mathcal{N}(\Omega, \| \cdot \|_{2 \to 2}, \varepsilon) \leq \left(1 + \frac{2\Lambda}{\varepsilon}\right)^{\eta n}.
\]
(58)

Now, for the covering numbers of \( \mathcal{M} \) we have
\[
\mathcal{N}(\mathcal{M}, \| \cdot \|_F, \varepsilon)
\leq \mathcal{N}(\mu^{-1}(\Lambda + \|A\|_{2 \to 2})K_L, \Omega, \| \cdot \|_{2 \to 2}, \varepsilon)
= \mathcal{N}(\Omega, \| \cdot \|_{2 \to 2}, \varepsilon/(\mu^{-1}(\Lambda + \|A\|_{2 \to 2})K_L))
\leq \left(1 + \frac{2\mu^{-1}(\Lambda + \|A\|_{2 \to 2})K_L}{\varepsilon}\right)^{\eta n}.
\]
\[
\Box
\]

E. Generalization Error Bounds

We are now in position to deliver generalization error bounds for DECONET.

Theorem III.10. Let \( \mathcal{H}_L \) be the hypothesis class defined in (30). With probability at least \( 1 - \delta \), for all \( h \in \mathcal{H}_L \), the generalization error is bounded as
\[
\mathcal{L}(h) \leq \hat{\mathcal{L}}(h) + 8(B_{\text{in}} + B_{\text{out}})B_{\text{out}} \sqrt{\frac{Nn}{s}}
\cdot \sqrt{\log \left(e \left(1 + \frac{4\mu^{-2}(\Lambda + \|A\|_{2 \to 2})K_L}{\sqrt{s}B_{\text{out}}}\right)\right)}
+ 4(B_{\text{in}} + B_{\text{out}})^2 \frac{2\log(4/\delta)}{s},
\]
with \( K_L \) defined in (48).

Proof. We apply Proposition III.9 to (55), yielding:
\[
\mathcal{R}_S(l \circ \mathcal{H}_L)
\leq \frac{16(B_{\text{in}} + B_{\text{out}})}{s} \int_0^{\frac{\sqrt{2}B_{\text{out}}}{s}} \sqrt{\log \mathcal{N}(\mathcal{M}, \| \cdot \|_F, \varepsilon)} d\varepsilon
\leq \frac{16(B_{\text{in}} + B_{\text{out}})}{s}
\cdot \int_0^{\frac{\sqrt{2}B_{\text{out}}}{s}} \sqrt{Nn \log \left(1 + \frac{2\mu^{-2}(\Lambda + \|A\|_{2 \to 2})K_L}{\varepsilon}\right)} d\varepsilon
\leq 8(B_{\text{in}} + B_{\text{out}})B_{\text{out}} \sqrt{\frac{Nn}{s}}
\cdot \sqrt{\log \left(e \left(1 + \frac{4\mu^{-2}(\Lambda + \|A\|_{2 \to 2})K_L}{\sqrt{s}B_{\text{out}}}\right)\right)},
\]
where in the last step we used the inequality
\[
\int_0^a \sqrt{\log \left(1 + \frac{b}{t}\right)} dt \leq a \sqrt{\log(e(1 + b/a))}, \quad a, b > 0.
\]
The proof follows by employing Theorem III.1 with the upper bound \( c = (B_{\text{in}} + B_{\text{out}})^2 \) for the loss function \( \| \cdot \|_2^2 \). \( \Box \)

Similarly to Section III-B, we may further assume that for the sequences of ratios \( \{c_{1,k}\}_{k \geq 0} = \{t^2_k/\mu^2\}_{k \geq 0} \leq 1 \), \( \{c_{2,k}\}_{k \geq 0} = \{t^2_k/\mu^2\}_{k \geq 0} \leq 1 \), it holds \( c_{1,k}\Lambda \leq 1, c_{1,k}\Lambda^2 \leq 1, c_{2,k}\|A\|_{2 \to 2}^2 \leq 1 \), for all \( k \geq 0 \), so we obtain

Corollary III.11. Let \( \mathcal{H}_L \) be the hypothesis class defined in (30) and assume that \( c_{1,k}\Lambda \leq 1, c_{1,k}\Lambda^2 \leq 1, c_{2,k}\|A\|_{2 \to 2}^2 \leq 1 \), for all \( k \geq 0 \), with \( \{c_{1,k}\}, \{c_{2,k}\} \leq 1 \) defined as in Lemma III.3. With probability at least \( 1 - \delta \), for all \( h \in \mathcal{H}_L \), the generalization error is bounded as
\[
\mathcal{L}(h) \leq \hat{\mathcal{L}}(h) + 8(B_{\text{in}} + B_{\text{out}}) \left(B_{\text{out}} \sqrt{\frac{Nn}{s}}
\cdot \sqrt{\log \left(e \left(1 + \frac{\|Y\|_F(p + qL + \tau_L)}{\sqrt{s}B_{\text{out}}}\right)\right)}
+ \frac{2\log(4/\delta)}{s}\right),
\]
with \( \kappa_L \) as in Theorem III.6 and \( p, q, r > 0 \) constants depending on \( \|A\|_{2 \to 2}, \Lambda, \mu \).

Proof. The estimate easily follows from Theorems III.6 and III.10, if we set \( p := \Lambda(\Lambda + \|A\|_{2 \to 2})\|A\|_{2 \to 2}, q := p(\mu^{-1} - 1) \) and \( r := 2\mu(\|A\|_{2 \to 2} + 1)(\|A\|_{2 \to 2} + 3) \). \( \Box \)

All the previous results are summarized in

Theorem III.12. Let \( \mathcal{H}_L \) be the hypothesis class defined in (30). Assume there exist pair-samples \( \{(x_i, y_i)\}_{i=1}^{s} \), with \( y_i = Ax_i + e_i \|e\|_2 \leq \varepsilon \), for some \( \varepsilon > 0 \), that are drawn i.i.d. according to an unknown distribution \( D \), and that it holds \( \|y_i\|_2 \leq B_{\text{in}} \) almost surely with \( B_{\text{in}} = B_{\text{out}} \) in (28). Let us further assume that for step sizes \( 0 < \{t^2_k\}_{k \geq 0} \), \( \{t^2_k\}_{k \geq 0} \leq 1 \), step size multiplier \( 0 < \{\theta_k\}_{k \geq 0} \leq 1 \) and smoothing parameter \( \mu > 1 \), we have \( \mu^{-1}\theta^{-1}_k t^4_k \Lambda \leq 1, \mu^{-1}\theta^{-1}_k t^4_k \Lambda^2 \leq 1, \mu^{-1}\theta^{-1}_k t^4_k \|A\|_{2 \to 2} \leq 1 \), for all \( k \geq 0 \). Then with probability at least \( 1 - \delta \), for all \( h \in \mathcal{H}_L \), the generalization error is bounded as
\[
\mathcal{L}(h) \leq \hat{\mathcal{L}}(h) + 16B_{\text{out}}^2 \sqrt{\frac{Nn}{s}}
\cdot \sqrt{\log \left(e \left(1 + \frac{p + qL + r\kappa_L}{\sqrt{s}B_{\text{out}}}\right)\right)}
+ 16B_{\text{out}}^2 \frac{2\log(4/\delta)}{s},
\]
with \( \kappa_L \) as in Theorem III.6 and constants \( p, q, r > 0 \) as in Corollary III.11.

We also state below a key remark, regarding the generalization error bound as a function of \( L, N \) and \( s \).

Corollary III.13 (Informal). According to (50), we have that \( L \) enters at most exponentially in the definition of \( \kappa_L \). If we consider the dependence of the generalization error bound (61) only on \( L, N, s \) and treat all other terms as constants, we roughly have
\[
|\mathcal{L}(h) - \hat{\mathcal{L}}(h)| \leq \sqrt{\frac{NL}{s}}.
\]
(62)
IV. NUMERICAL EXPERIMENTS

In this Section, we examine whether our theory regarding the generalization error of DECONET is consistent to real-world applications of our framework.

A. Experimental Setup

We train and test DECONET on a synthetic dataset of random vectors, drawn from the normal distribution (70000 training and 10000 test examples), and two real-world image datasets: MNIST (60000 training and 10000 test 28 × 28 image examples) and CIFAR10 (50000 training and 10000 test 32 × 32 coloured image examples). For the CIFAR10 dataset, we transform the images into grayscale ones. We examine DECONET with varying number of layers $L$. We consider two CS ratios, i.e. $m/n = 25\%$ and $m/n = 50\%$. We choose a random Gaussian measurement matrix $A \in \mathbb{R}^{m \times n}$ and appropriately normalize it, i.e., $\tilde{A} = A/\sqrt{m}$. We add zero-mean Gaussian noise $\varepsilon$ with standard deviation std $= 10^{-4}$ to the measurements $y$, so that $y = \tilde{A}x + \varepsilon$. We set $\varepsilon = \|y - \tilde{A}x\|_2$ and $x_0 = A^T y$, which are standard algorithmic setups. We take different values of $N$ and perform two different initializations for $W \in \mathbb{R}^{N \times n}$: normal initialization and initialization based on Beta distribution [52], with varying values of Beta’s parameters $a$ and $b$. We set $\mu = 100$, initial step sizes $t_0 = t_0^2 = 1$ and step size multiplier $\theta_0 = 1$. For $t_k$, $t_k^2$, $\theta_k$, we
apply the following update rules: 

\[ t_{1_k} = \alpha t_{1_{k-1}}, \quad t_{2_k} = \beta t_{2_{k-1}}, \]

\[ \theta_k = \theta_{k-1}, \quad \theta' = \frac{1}{1 + \sqrt{\mu / \tilde{L}}}, \]

where \( (\alpha, \beta) \in (0, 1) \times (0, 1) \), \( \theta' = \frac{1}{1 + \sqrt{\mu / \tilde{L}}} \), respectively, and \( \tilde{L} \) is an upper bound on the smoothing parameter \( \mu \); we set \( \tilde{L} = 1000 \). All networks are implemented in PyTorch [53] and trained using the Adam optimizer [54], with batch size 128. For our experiments, we report the test MSE defined by

\[ L_{\text{test}} = \frac{1}{d} \sum_{i=1}^{d} \| h(\tilde{y}_i) - \tilde{x}_i \|_2^2, \quad (63) \]

where \( D = \{(\tilde{y}_i, \tilde{x}_i)\}_{i=1}^{d} \) is a set of \( d \) test data, not used in the training phase. We also report the empirical generalization error (EGE) defined by

\[ L_{\text{gen}} = |L_{\text{test}} - L_{\text{train}}|, \quad (64) \]

where \( L_{\text{train}} \) is the train MSE defined in (32). Since test MSE approximates the true loss, we use (64) – which can be explicitly computed – to approximate the generalization error of (34). We train all networks, on all datasets, employing an early stopping technique [55] with respect to (64). We repeat all the experiments at least 10 times and average the results over the runs. We compare the EGE of DECONET to the EGE achieved by two SotA unfolding networks serving as baselines: a recent variant of ISTA-net [20] and ADMM-DAD net [28]. Both baselines jointly learn a decoder for CS and a sparsifying transform. Nevertheless, ISTA-net solves the CS problem employing synthesis sparsity, since the learnable sparsifier is orthogonal, while ADMM-DAD involves analysis sparsity, by learning a redundant analysis operator. We aim to see how the EGE is affected by each of the two sparsity models and if DECONET outperforms ADMM-DAD. For both baselines, we set the best hyper-parameters proposed by the original authors.

B. Experimental Results

We test DECONET on all datasets under multiple experimental scenarios.

1) Fixed CS ratio with varying \( N/n \) and \( L \), for different initializations, on image datasets: We examine the performance of 10- and 50-layer DECONET for a fixed 25% CS ratio, varying redundancy ratio \( N/n \) and both normal and Beta initializations for \( W \). We report the results for MNIST and CIFAR10 in Fig. 1. As illustrated in Fig. 1a, the test MSEs, achieved by 10- and 50-layer DECONET on both datasets, drop as \( L \) and \( N/n \) increase, for both types of initialization. The decays seem reasonable, if one considers a standard analysis CS scenario: the reconstruction quality and performance of the analysis-l1 algorithm, typically benefit from the (high) redundancy offered by the involved analysis operator, especially as the number of iterations/layers increases. Furthermore, Fig. 1b demonstrates that the EGE of DECONET increases as both \( L \) and \( N/n \) increase, for both normal and Beta initialization. For the latter, we observe that the different values of its parameters affect the generalization ability of DECONET on both image datasets. The overall performance of DECONET confirms our theoretical results depicted in Section III-E, since EGEs seem to scale like \( \sqrt{NL} \).

2) Fixed CS ratio, with varying \( N \) and \( L \), on image datasets: We examine the generalization ability of DECONET for \( m = n/2 \), with increasing number of layers \( L \), under different choices of \( N \) and normal initialization. Inspired by

---

4To the best of author’s knowledge, ADMM-DAD is the only (other than DECONET) unfolding network that entails analysis sparsity for solving the CS problem.
frames with redundancy ratio $N/n \notin \mathbb{N}$ \[56\], we consider $N$ of the form

$$N = pn + q, \quad p, q \in \mathbb{N},$$ \[65\]

We report the results in Fig. 2 for MNIST and CIFAR10. Similarly to Section IV-B1, we observe that the empirical generalization error increases in $L$ and $N$, for both datasets. Even though the upper bound in (61) depends on other terms too, the empirical generalization error appears to grow at the rate of $\sqrt{NL}$. The behaviour of DECONET again conforms with our theoretical results presented in Section III. One may also notice that – in general – we choose different $N$ for each of the two datasets. This is simply due to (65), i.e., $N$ depends on the vectorized ambient dimension $n$, which is different for each of the two datasets.

3) Fixed CS ratio with varying $n$, $N$, $L$, on synthetic dataset: Similar to Sections IV-B1 and IV-B2, we examine the generalization error that DECONET achieves on a synthetic dataset of random vectors, with a normally initialized $W$. We present the results\(^5\) in Fig. 3. Particularly, the top plot of Fig. 3 demonstrates how the EGE for 10-layer DECONET scales, for different values of the ambient dimension $n$, with 25\% CS ratio, as $N/n$ increases. On the other hand, as depicted in the bottom plot of Fig. 3, we revisit the experimental reasoning of Section IV-B2, with fixed $n = 300$ and $m = n/2$. In both subplots, we see that the EGEs achieved by DECONET comply with our theory.

4) Comparison to baselines: We examine how analysis and synthesis sparsity models affect the generalization ability of CS-oriented unfolding networks. Towards this end, we compare the proposed DECONET’s decoder to ISTA-net’s and ADMM-DAD’s decoders. For the image datasets, the comparisons are made for 10, 20 and 30 layers, with 25\% and 50\% CS ratio, and fixed $N = 37145$ for both DECONET’s and ADMM-DAD’s sparsifiers. For the synthetic dataset, we fix $N = 12000$ (for both DECONET’s and ADMM-DAD’s sparsifiers), $n = 300$, $m = n/2$, and vary $L$. We report the empirical generalization errors for the image datasets in Table II and for the synthetic dataset in Fig. 4. The aforementioned table and figure demonstrate that our proposed decoder outperforms both baseline decoders, consistently for all datasets. In fact, both DECONET and ADMM-DAD outperform the synthesis-sparsity-based ISTA-net. This behaviour indicates that learning a redundant sparsifier instead of an orthogonal one, improves the performance of a CS-oriented unfolding network. Nevertheless, our proposed network outperforms ADMM-DAD, which is considered to be a SotA unfolding network for analysis CS. Second, our theoretical results on the generalization error of DECONET seem to align with the experiments, since the EGE of DECONET increases as $L$ also increases. Furthermore, we observe that the EGE of DECONET on the CIFAR10 dataset decreases as the number of measurements $m$ increases.

V. CONCLUSION AND FUTURE WORK

In this paper we derived DECONET, a new deep unfolding network for solving the analysis-sparsity-based Compressed Sensing problem. DECONET jointly learns a decoder for CS and a redundant sparsifying analysis operator. Furthermore, we estimated the generalization error of DECONET, in terms of the Rademacher complexity of the associated hypothesis class. Our generalization error bounds roughly scale like the square root of the product between the number of layers and the redundancy of the learnable sparsifier. To the best of our knowledge, this is the first result of its kind for unfolding networks solving the analysis CS problem. Furthermore, we conducted experiments that confirmed the validity of our theoretical results and compared DECONET to state-of-the-art CS-oriented unfolding networks. Our proposed network

![Comparison with baselines, synthetic dataset](image)

Fig. 4: Performance plot for all decoders, with fixed $m$, $n$, $N$, on a synthetic dataset.

\(^5\) all the plots regarding the synthetic data are made in logarithmic scale.

### Table II: Empirical generalization errors for 10-, 20- and 30-layer decoders, with 25\% and 50\% CS ratios, and fixed $N = 37145$ for DECONET’s and ADMM-DAD’s sparsifiers. Bold letters indicate the best performance among the three decoders.

| Dataset  | MNIST | CIFAR10 |
|----------|-------|---------|
| Layers   |       |         |
| Decoder  |       |         |
| DECONET  |       |         |
| ADMM-DAD |       |         |
| ISTA-net |       |         |

| 25\% CS ratio | 50\% CS ratio |
|---------------|---------------|
| $L = 10$     | $L = 10$     |
| $L = 20$     | $L = 20$     |
| $L = 30$     | $L = 30$     |
| $N = 10000$  | $N = 10000$  |
| $N = 20000$  | $N = 20000$  |
| $N = 30000$  | $N = 30000$  |
| $N = 40000$  | $N = 40000$  |
| $N = 50000$  | $N = 50000$  |

| Encoder     |       |         |
|-------------|-------|---------|
| Decoder     |       |         |
| DECONET     |       |         |
| ADMM-DAD    |       |         |
| ISTA-net    |       |         |
outperformed the baselines, consistently for synthetic and real-world datasets. As a future direction, we would like to examine the performance of our proposed framework on speech datasets and experiment with different values of the non-learnable parameters. Additionally, it would be interesting to further characterize (e.g. in terms of structure) the sparsifying transform that DECONET learns.

ACKNOWLEDGEMENTS

V. Kouni would like to thank Holger Rauhut for his advice and helpful conversations around the subject covered in this paper.

REFERENCES

[1] E. J. Candès, J. Romberg, and T. Tao. “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information”. In: IEEE Trans. Inf. Theory 52.2 (2006), pp. 489–509.
[2] I. Daubechies, M. Defrise, and C. De Mol. “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint”. In: Commun. Pure and Appl. Math. 57.11 (2004), pp. 1413–1457.
[3] R. Chartrand and W. Yin. “Iteratively reweighted algorithms for compressive sensing”. In: Int. Conf. Acoust., Speech and Signal Process. IEEE. 2008, pp. 3869–3872.
[4] S. Boyd, N. Parikh, and E. Chu. Distributed optimization and statistical learning via the alternating direction method of multipliers. Now Publishers Inc, 2011.
[5] H. Yao et al. “Dr2-net: Deep residual reconstruction network for image compressive sensing”. In: Neurocomputing 359 (2019), pp. 483–493.
[6] G. Yang et al. “DAGAN: Deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction”. In: IEEE Trans. Med. Imag. 37.6 (2017), pp. 1310–1321.
[7] J. R. Hershey, J. Le Roux, and F. Weninger. “Deep unfolding: Model-based inspiration of novel deep architectures”. In: arXiv preprint arXiv:1409.2574 (2014).
[8] V. Monga, Y. Li, and Y. C Eldar. “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing”. In: IEEE Signal Process. Mag. 38.2 (2021), pp. 18–44.
[9] K. Gregor and Y. LeCun. “Learning fast approximations of sparse coding”. In: Proc. 27th Int. Conf. Mach. Learn. 2010, pp. 399–406.
[10] J. Zhang et al. “Deep Unfolding With Weighted $l_2$ Minimization for Compressive Sensing”. In: IEEE Internet of Things J. 8.4 (2020), pp. 3027–3041.
[11] C. Bertocchi et al. “Deep unfolding of a proximal interior point method for image restoration”. In: Inverse Problems 36.3 (2020), p. 034005.
[12] Y. Yang et al. “A Robust Deep Unfolded Network for Sparse Signal Recovery from Noisy Binary Measurements”. In: Eur. Signal Process. Conf. IEEE. 2021, pp. 2060–2064.
[13] A. P. Sabulal and S. Bhashyam. “Joint Sparse Recovery Using Deep Unfolding With Application to Massive Random Access”. In: Int. Conf. Acoust., Speech and Signal Process. 2020, pp. 5050–5054.
[14] M. Borgerding, P. Schniter, and S. Rangan. “AMP-Inspired Deep Networks for Sparse Linear Inverse Problems”. In: IEEE Trans. Signal Process. 65.16 (2017), pp. 4293–4308.
[15] W. Pu, Y. C. Eldar, and M. RD. Rodrigues. “Optimization Guarantees for ISTA and ADMM Based Unfolded Networks”. In: Int. Conf. Acoust., Speech and Signal Process. IEEE. 2022, pp. 8687–8691.
[16] J. Zhang and B. Ghanem. “ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing”. In: Proc. IEEE Comput. Vision and Pattern Recognit. 2018, pp. 1828–1837.
[17] Y. Khalifa, Z. Zhang, and E. Sejdić. “Sparse recovery of time-frequency representations via recurrent neural networks”. In: 22nd Int. Conf. Digit. Signal Process. IEEE. 2017, pp. 1–5.
[18] J. Sun et al. “Deep ADMM-Net for compressive sensing MRI”. In: Advances Neural Inf. Process. Syst. 29 (2016).
[19] P. Xiao, B. Liao, and N. Deligiannis. “Deepfpc: A deep unfolded network for sparse signal recovery from 1-bit measurements with application to doa estimation”. In: Signal Process. 176 (2020), p. 107699.
[20] A. Behboodi, H. Rauhut, and E. Schnoor. “Compressive Sensing and Neural Networks from a Statistical Learning Perspective”. In: arXiv preprint arXiv:2010.15658 (2020).
[21] Y. Shen et al. “Image reconstruction algorithm from compressed sensing measurements by dictionary learning”. In: Neurocomputing 151 (2015), pp. 1153–1162.
[22] Z. Li, H. Huang, and S. Misra. “Compressed sensing via dictionary learning and approximate message passing for multimedia Internet of Things”. In: IEEE Internet of Things J. 4.2 (2016), pp. 505–512.
[23] H. Zayyani, M. Korki, and F. Marvasti. “Dictionary learning for blind one bit compressed sensing”. In: IEEE Signal Process. Lett. 23.2 (2015), pp. 187–191.
[24] S. Shalev-Shwartz and S. Ben-David. Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.
[25] A. Xu and M. Raginsky. “Information-theoretic analysis of generalization capability of learning algorithms”. In: Advances Neural Inf. Process. Syst. 30 (2017).
[26] S. Nam et al. “The csparse analysis model and algorithms”. In: Appl. and Comput. Harmon. Anal. 34.1 (2013), pp. 30–56.
[27] H. Cherkaoui et al. “Analysis vs synthesis-based regularization for combined compressed sensing and parallel MRI reconstruction at 7 tesla”. In: Eur. Signal Process. Conf. IEEE. 2018, pp. 36–40.
[28] V. Kouni et al. “ADMM-DAD Net: A Deep Unfolding Network for Analysis Compressed Sensing”. In: Int. Conf. Acoust., Speech and Signal Process. IEEE. 2022, pp. 1506–1510.
