The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

A Provable Approach for Double-Sparse Coding

Thanh V. Nguyen
ECE Department
Iowa State University
thanhng@iastate.edu

Raymond K. W. Wong
Statistics Department
Texas A&M University
raywong@tamu.edu

Chinmay Hegde *
ECE Department
Iowa State University
chinmay@iastate.edu

Abstract

Sparse coding is a crucial subroutine in algorithms for various signal processing, deep learning, and other machine learning applications. The central goal is to learn an overcomplete dictionary that can sparsely represent a given dataset. However, storage, transmission, and processing of the learned dictionary can be untenably high if the data dimension is high. In this paper, we consider the double-sparsity model introduced by Rubinstein, Zibulevsky, and Elad (2010) where the dictionary itself is the product of a fixed, known basis and a data-adaptive sparse component. First, we introduce a simple algorithm for double-sparse coding that can be amenable to efficient implementation via neural architectures. Second, we theoretically analyze its performance and demonstrate asymptotic sample complexity and running time benefits over existing (provable) approaches for sparse coding. To our knowledge, our work introduces the first computationally efficient algorithm for double-sparse coding that enjoys rigorous statistical guarantees. Finally, we support our analysis via several numerical experiments on simulated data, confirming that our method can indeed be useful in problem sizes encountered in practical applications.

Introduction

We consider the problem of *dictionary learning* (also known as sparse coding), a common and powerful technique in unsupervised feature learning. The high-level idea of sparse coding is to represent a set of data vectors in terms of *sparse* linear combinations of atoms from a learned basis (or dictionary). Sparse coding has a rich history in diverse fields such as image processing, machine learning, and neuroscience (Krim et al. 1999; Elad and Aharon 2006; Rubinstein, Bruckstein, and Elad 2010; Mairal et al. 2009). Sparse coding forms a core component of several neural learning systems, both biological (Olshausen and Field 1997) and artificial (Gregor and LeCun 2010; Boureau et al. 2010; Mazumdar and Rawat 2017).

Formally, suppose we are given p data samples $Y = [y^{(1)}, y^{(2)}, \ldots, y^{(p)}] \in \mathbb{R}^{n\times p}$. We wish to find a dictionary $D \in \mathbb{R}^{n\times m}$ (with $n < m$) and corresponding sparse code vectors $X = [x^{(1)}, x^{(2)}, \ldots, x^{(p)}] \in \mathbb{R}^{m\times p}$ such that the representation DX fits the data samples as well as possible. The typical approach is to pose the dictionary (and codes) as the solution to the constrained optimization problem:

$$\begin{aligned} \min_{D,X} \mathcal{L}(D, X) &= \frac{1}{2}\sum_{j=1}^{p} \|y^{(j)} - Dx^{(j)}\|_2^2, \\ \text{s.t. } &\sum_{j=1}^{p} \mathcal{S}(x^{(j)}) \le S, \end{aligned} \tag{1}$$

where $\mathcal{S}(\cdot)$ is some sparsity-inducing penalty function, such as the ℓ_1-norm. However, even a cursory attempt at solving (1) reveals several conceptual obstacles:

Theoretical challenges. The constrained optimization problem (1) involves a non-convex (bilinear) objective function, as well as potentially non-convex constraints depending on the function $\mathcal{S}$. Therefore, design and analysis of *provably* correct algorithms for this problem can be difficult. Indeed, the vast majority of practical approaches for sparse coding are based on heuristics; barring a few recent papers from the learning theory community (Spielman, Wang, and Wright 2012; Agarwal et al. 2014; Arora et al. 2015; Sun, Qu, and Wright 2015), very few methods come equipped with global correctness guarantees.

Challenges in applications. Even if we ignore theoretical correctness issues and somehow are able to learn good enough sparse codes, we often find that applications using such learned sparse codes encounter *memory* and *running-time* issues. Indeed, in the overcomplete case, merely storing the learned dictionary D incurs $mn = \Omega(n^2)$ memory cost, which is prohibitive when n is large. In practical applications such as image analysis, one typically resorts to chopping the data into smaller blocks (e.g., partitioning image data into patches) to make the problem manageable.

An approach used to resolve those practical computational difficulties is to assume some type of structure in the (learned) dictionary D; e.g., the dictionary is assumed to be either separable, or obey a convolutional structure. One such variant is *double-sparse* coding (Rubinstein, Zibulevsky, and Elad 2010; Sulam et al. 2016) where the dictionary D *itself* exhibits a sparse structure. More precisely,

$$D = \Phi A,$$

*This work is supported in part by the National Science Foundation under the grants CCF-1566281 and DMS-1612985.

Setting	Reference	Sample complexity (w/o noise)	Sample complexity (w/ noise)	Upper bound on running time	Expt
Regular	MOD (Engan, Aase, and Husoy 1999)	✗	✗	✗	✓
	K-SVD (Aharon, Elad, and Bruckstein 2006)	✗	✗	✗	✓
	(Spielman, Wang, and Wright 2012)	$O(n^2 \log n)$	✗	$\widetilde{\Omega}(n^4)$	✓
	(Arora, Ge, and Moitra 2014)	$\widetilde{O}(m^2/k^2)$	✗	$\widetilde{O}(np^2)$	✗
	(Gribonval, Jenatton, and Bach 2015)	$O(nm^3)$	$O(nm^3)$	✗	✗
	(Arora et al. 2015)	$\widetilde{O}(mk)$	✗	$\widetilde{O}(mn^2p)$	✗
Double Sparse	Double Sparsity (Rubinstein, Zibulevsky, and Elad 2010)	✗	✗	✗	✓
	(Gribonval et al. 2015)	$\widetilde{O}(mr)$	$\widetilde{O}(mr)$	✗	✗
	Trainlets (Sulam et al. 2016)	✗	✗	✗	✓
	This paper	$\widetilde{O}(mr)$	$\widetilde{O}(mr + \sigma_\varepsilon^2 \frac{mnr}{k})$	$\widetilde{O}(mnp)$	✓

Table 1: Comparison of various sparse coding techniques. Expt: whether numerical experiments have been conducted. ✗ in all other columns indicates no provable guarantees. Here, n is the signal dimension, and m is the number of atoms. The sparsity levels for A and x are r and k respectively, and p is the sample size.

with a known "base dictionary" $\Phi \in \mathbb{R}^{n\times n}$ and a learned column-sparse "synthesis" matrix $A \in \mathbb{R}^{n\times m}$. The base dictionary Φ is typically any orthonormal basis (such as the canonical or wavelet basis) chosen according to domain knowledge, while the synthesis matrix A is *column-wise sparse* and is learned from the data. Such a double-sparsity assumption is appealing conceptually, since it lets us combine the knowledge of good dictionaries Φ to synthesize new representations tailored to specific data families. Moreover, the sparse structure of the synthesis matrix produces more interpretable features than the regular model; see (Rubinstein, Zibulevsky, and Elad 2010; Sulam et al. 2016) for an extensive discussion and illustration. If the columns of A has only (say) $r \ll n$ non-zero elements, then the overall memory burden of storing and transmitting A is $O(mr)$, which is much lower than that for general unstructured dictionaries. Moreover, this approach performs comparably with (regular) sparse coding approaches, as demonstrated by the extensive empirical evaluations in (Sulam et al. 2016). However, two obstacles remain: the associated *training* algorithms used to learn double-sparse codes incur significant running time, and no rigorous theoretical analysis of their performance has been reported in the literature.

Our Contributions

We provide a new algorithmic approach to double-sparse coding. To the best of our knowledge, our approach is the first method that enjoys *provable* statistical and algorithmic guarantees for the double-sparse coding problem. In addition, our approach enjoys three benefits: (i) our method is *tractable* as well as *neurally plausible*, i.e., its execution can plausibly be achieved using a neural network architecture; (ii) our method enjoys *noise robustness* guarantees; (iii) we demonstrate *practical relevance* via several simulations.

Inspired by the aforementioned recent theoretical papers in sparse coding, we assume a learning-theoretic setup where the data samples arise from a ground-truth generative model. Informally, suppose there exists a true (but unknown) synthesis matrix $A^* \in \mathbb{R}^{n\times m}$ whose columns have only r non-zero elements, and the i^{th} data sample is generated as:

$$y^{(i)} = \Phi A^* x^{*(i)} + \text{ noise}, \quad i = 1, 2, \ldots, p,$$

where the code vector $x^{*(i)}$ is independently drawn from a distribution supported on the set of k-sparse vectors. We desire to learn the matrix A^*. We suppose that the synthesis matrix A^* is *incoherent* (the columns of A^* are sufficiently close to orthogonal) and has bounded spectral norm, that m is at most a constant multiple of n, and that the noise is sub-Gaussian. All these assumptions are standard[1].

First, we propose and analyze an algorithm that produces a coarse estimate of the synthesis matrix that is sufficiently close to the ground truth A^*. Our method builds upon the method of *spectral initialization* that have recently gained popularity in non-convex machine learning (Zhang et al. 2016; Wang, Zhang, and Gu 2016).

Second, given such a coarse estimate of the synthesis matrix A^*, we propose and analyze a gradient descent-style algorithm to refine this estimate. This algorithm is simpler than previously studied double-sparse coding algorithms that rely on alternating minimization (such as the Trainlets approach of (Sulam et al. 2016)), while still giving good statistical performance.

Put together, the above constitutes the first provably polynomial-time method for double-sparse coding. In particular, in the absence of noise, we prove that $p = \Omega(mr \text{ polylog } n)$ samples are sufficient to obtain a good enough estimate in the initialization, and also to obtain guaranteed linear convergence during descent to provably recover A^*. See Table 1 for the summary and a comparison with the existing work. Indeed, our sample complexity result matches with what is achieved in (Gribonval et al. 2015). Nevertheless, we provide a practical polynomial-time algorithm for learning the sparse dictionary whereas Gribonval et al. only study properties of a theoretical estimator. Also, our approach results in strict improvement in sample complexity, as well as running time over rigorous methods for (regular) sparse coding, such as (Arora et al. 2015).

We analyze our approach in a more realistic setting with the presence of additive noise, and demonstrate its stability. While our analysis mainly consists of sufficiency results and involves several (absolute) unspecified constants, in practice we have found that these constants are reasonable. We jus-

[1]We clarify the generative model in concrete terms below.

tify our observations by reporting a suite of numerical experiments on synthetic test datasets.

Techniques

The remainder of the paper is fairly technical; therefore, for clarity let us provide some non-rigorous intuition for our approach. At a high level, our method extends the neural sparse coding approach of (Arora et al. 2015) to the double-sparse case. A major barrier in the analysis of sparse coding algorithms is that the gradient of $\mathcal{L}$ in (1) with respect to D inherently depends on the codes of the training samples (i.e., the columns of X), but these codes are unknown *a priori*. However, the main insight in (Arora et al. 2015) is that within a small enough neighborhood of the true dictionary, an approximated version of X^* can be estimated, and therefore the overall method is similar to performing *approximate* gradient descent towards the population parameter A^*. Regarding the actual algorithm as its noisy variation allows us to overcome the finite-sample variability of the loss, and obtain a descent property directly related to A^*.

The descent stage of our approach leverages this intuition. However, instead of standard gradient descent, we perform approximate *projected* gradient descent so that the column-wise r-sparsity property is enforced in each new estimate of A^*. This extra projection step is critical in showing sample complexity improvement over (regular) sparse coding methods. The key novelty is in figuring out how to perform the projection in each gradient iteration. For this purpose, we develop a novel initialization algorithm that identifies the locations of the non-zeroes in A^* even before commencing the descent phase. This is non-trivially different from previous rigorous methods for sparse coding, and the analysis is somewhat more involved.

In (Arora et al. 2015), (the principal eigenvector of) the weighted covariance matrix of y, given by a suitable weighted average of outer products $y_i y_i^T$, is shown to provide a coarse estimate of a given dictionary atom. We leverage this idea and rigorously show that the diagonal of the weighted covariance matrix serves as a good indicator of the support of a column in A^*. The success relies on the concentration of the diagonal vector with dimension n, instead of the covariance matrix with dimensions $n \times n$. With the support selected, our scheme only utilizes a *truncated* weighted covariance matrix with dimensions $r \times r$. This initialization scheme enables us to effectively reduce the dimension of the problem, and therefore leads to significant improvement in sample complexity and running time over previous (provable) sparse coding methods when the data representation sparsity k is much smaller than m.

Further, we rigorously analyze the proposed algorithms in the presence of noise with a bounded expected norm. Our analysis shows that our method is stable, and in the case of i.i.d. Gaussian noise with bounded expected ℓ_2-norms, is at least a polynomial factor better than previous polynomial time algorithms for sparse coding in terms of running time. Our analysis of the descent stage follows from (Arora et al. 2015), where the descent property is first shown under an ideal algorithm which uses the expectation of the noisy (approximate) gradient, and is later established to the practical case via a concentration argument. Our novel initialization algorithm allows an accurate determination of the support of A^*, and therefore, for each column of A^*, we can focus on an r-dimensional subvector of the noisy (approximate) gradient vector, rather than the full n-dimensional vector. This allows us to sharpen the sample complexity beyond what has been established in the earlier work.

Setup and Definitions

Notation. Let $[m] \triangleq \{1, 2, \ldots, m\}$ for some integer m. For any vector $x = [x_1, x_2, \ldots, x_m]^T \in \mathbb{R}^m$, let $\mathrm{supp}(x) \triangleq \{i \in [m] : x_i \neq 0\}$. Given any subset $S \subseteq [m]$, x_S corresponds to the sub-vector of x indexed by the elements of S. For any matrix $A \in \mathbb{R}^{n \times m}$, we use $A_{\bullet i}$ and $A_{j\bullet}^T$ to represent the i^{th} column and the j^{th} row respectively. For some appropriate sets R and S, let $A_{R\bullet}$ (respectively, $A_{\bullet S}$) be the sub-matrix of A with rows (respectively columns) indexed by the elements in R (respectively S). For the i^{th} column $A_{\bullet i}$, use $A_{R,i}$ to denote the sub-vector indexed by the elements of R. Use $A_{R\bullet}^T$ to indicate $(A_{R\bullet})^T$. Let $\circ$ and $\mathrm{sgn}(\cdot)$ represent the (element-wise) Hadamard operator and sign function. Further, $\mathrm{threshold}_K(x)$ is a thresholding operator that replaces any elements of x with magnitude less than K by zero.

The ℓ_2-norm $\|x\|$ for a vector x and the spectral norm $\|A\|$ for a matrix A are used extensively in this paper. In some cases, we also utilize the Frobenius norm $\|A\|_F$ and a special matrix operator norm $\|A\|_{1,2} \triangleq \max_{\|x\|_1 \leq 1} \|Ax\|$.

For clarity purposes, we adopt big-Oh notation extensively. The symbols $\widetilde{\Omega}(\cdot)$ and $\widetilde{O}(\cdot)$ represent $\Omega(\cdot)$ and $O(\cdot)$ up to a multiplicative poly-logarithmic factor of n respectively. Throughout the paper, we use the phrase "with high probability" (abbreviated to w.h.p.) to describe an event with failure probability of $O(n^{-\omega(1)})$. In addition, $g(n) = O^*(f(n))$ means $g(n) \leq Kf(n)$ for some small enough constant K.

Model. Suppose that the observed samples are given by

$$y^{(i)} = Dx^{*(i)} + \varepsilon, \;\; i = 1, \ldots, p;$$

i.e., we are given p samples of y generated from a fixed (but unknown) dictionary D where the sparse code x^* and the error ε are drawn from a joint distribution $\mathcal{D}$ specified below. In the double-sparse setting, the dictionary is assumed to follow a decomposition $D = \Phi A^*$, where $\Phi \in \mathbb{R}^{n \times n}$ is a known orthonormal basis matrix and A^* is an unknown, ground truth synthesis matrix. Our approach relies upon the following assumptions on the synthesis dictionary A^*:

A1 The dimensions of A^* obey $m = O(n)$.

A2 A^* is μ-incoherent, i.e., for $i \neq j$, $|\langle A_{\bullet i}^*, A_{\bullet j}^* \rangle| \leq \mu/\sqrt{n}$.

A3 $A_{\bullet i}^*$ has exactly r non-zero elements, and is normalized such that $\|A_{\bullet i}^*\| = 1$ for all i. Moreover, $|A_{ij}^*| \geq \tau$ for $A_{ij}^* \neq 0$ and $\tau = \Omega(1/\sqrt{r})$.

A4 A^* has bounded spectral norm: $\|A^*\| \leq O(\sqrt{m/n})$.

These assumptions are standard. In Assumption **A2**, the incoherence μ is $O(1/\log n)$ with high probability for a normal random matrix (Arora, Ge, and Moitra 2014). Assump-

tion **A3** is a common assumption for sparse signal recovery [2]. Assumption **A4** is also standard (Arora et al. 2015). In addition to Assumptions **A1-A4**, we make the following distributional assumptions on $\mathcal{D}$:

B1 The support $S = \text{supp}(x^*)$ is of size at most k; its indices are uniformly drawn without replacement from $[m]$.

B2 The nonzero entries x^*_S are pairwise independent and sub-Gaussian conditioned on the support S, with $\mathbb{E}[x^*_i | i \in S] = 0$ and $\mathbb{E}[x^{*2}_i | i \in S] = 1$.

B3 For $i \in S$, $|x^*_i| \geq C$ where $0 < C \leq 1$.

B4 The additive noise ε has i.i.d. Gaussian entries with variance σ^2_ε with $\sigma_\varepsilon = O(1/\sqrt{n})$.

Similar sub-Gaussian models for $\mathcal{D}$ have been previously considered in (Jenatton, Gribonval, and Bach 2012).

For the rest of the paper, for notational simplicity we set $\Phi = I_n$, i.e., the identity matrix. This does not affect anything, since one can study the equivalent problem:

$$y' = A^* x^* + \varepsilon',$$

where $y' = \Phi^T y$ and $\varepsilon' = \Phi^T \varepsilon$. Due to the Gaussian assumption on ε, it follows that ε' also has independent Gaussian entries. The analysis can be extended to sub-Gaussian noise with several minor (but tedious) changes.

Our goal is to devise an algorithm that produces an provably "good" estimate of A^*. For this, we need to define a suitable measure of "goodness". We use the following notion of distance that measures the maximal column-wise difference in ℓ_2-norm under a suitable transformation.

Definition 1 ((δ, κ)-nearness)**.** *A is said to be δ-close to A^* if there is a permutation $\pi : [m] \to [m]$ and a sign flip $\sigma : [m] : \{\pm 1\}$ such that $\|\sigma(i) A_{\bullet \pi(i)} - A^*_{\bullet i}\| \leq \delta$ for every i. In addition, A is said to be (δ, κ)-near to A^* if $\|A_{\bullet \pi} - A^*\| \leq \kappa \|A^*\|$ also holds.*

For notational simplicity, in our theorems we simply replace π and σ in Definition 1 with the identity permutation $\pi(i) = i$ and the positive sign $\sigma(\cdot) = +1$ while keeping in mind that in reality we are referring to finding one element of the equivalence class of all permutations and sign flip transforms of A^*.

We will also need some technical tools from (Arora et al. 2015) to analyze gradient descent-style methods. Consider an iterative algorithm that looks for a desired solution $z^* \in \mathbb{R}^n$ to optimize some function $f(z)$. Suppose that the algorithm produces a sequence of estimates $z^1, \ldots, z^s$ via the update rule:

$$z^{s+1} = z^s - \eta g^s,$$

for some vector g^s and scalar step size η. The goal is to characterize "good" directions g^s such that the sequence converges to z^* under the Euclidean distance. The following gives one such sufficient condition for g^s.

[2] The requirement of exactly r non-zero elements is merely for simplicity and there is no technical difficulty to extend our algorithms and corresponding analyses to the case with at most r non-zero elements.

Definition 2. *A vector g^s at the s^{th} iteration is $(\alpha, \beta, \gamma_s)$-correlated with a desired solution z^* if*

$$\langle g^s, z^s - z^* \rangle \geq \alpha \|z^s - z^*\|^2 + \beta \|g^s\|^2 - \gamma_s.$$

We know from convex optimization that if f is 2α-strongly convex and $1/2\beta$-smooth, and g^s is chosen as the gradient $\nabla_z f(z)$, then g^s is $(\alpha, \beta, 0)$-correlated with z^*. In our setting, the desired solution corresponds to A^*, the ground-truth synthesis matrix. In (Arora et al. 2015), it is shown that $g^s = \mathbb{E}_y[(A^s x - y)\text{sgn}(x)^T]$, where $x = \text{threshold}_{C/2}((A^s)^T y)$ indeed satisfies Definition 2. This g^s is a population quantity and not explicitly available, but one can estimate such g^s using an empirical average. The corresponding estimator $\widehat{g}^s$ is a random variable, so we also need a related *correlated-with-high-probability* condition:

Definition 3. *A direction $\widehat{g}^s$ at the s^{th} iteration is $(\alpha, \beta, \gamma_s)$-correlated-w.h.p. with a desired solution z^* if, w.h.p.,*

$$\langle \widehat{g}^s, z^s - z^* \rangle \geq \alpha \|z^s - z^*\|^2 + \beta \|\widehat{g}^s\|^2 - \gamma_s.$$

From Definition 2, one can establish a form of descent property in each update step, as shown in Theorem 1.

Theorem 1 (Convergence of approximate gradient descent)**.** *Suppose that g^s satisfies the condition described in Definition 2 for $s = 1, 2, \ldots, T$. Moreover, $0 < \eta \leq 2\beta$ and $\gamma = \max_{s=1}^T \gamma_s$. Then, the following holds for all s:*

$$\|z^{s+1} - z^*\|^2 \leq (1 - 2\alpha\eta) \|z^s - z^*\|^2 + 2\eta\gamma_s.$$

In particular, the above update converges geometrically to z^* with an error γ/α. That is,

$$\|z^{s+1} - z^*\|^2 \leq (1 - 2\alpha\eta)^s \|z^0 - z^*\|^2 + 2\gamma/\alpha.$$

We can obtain a similar result for Definition 3 except that $\|z^{s+1} - z^*\|^2$ is replaced with its expectation.

Armed with the above tools, we are now ready to introduce our method. As discussed above, our approach consists of two stages: an initialization algorithm that produces a coarse estimate of A^*, and a descent-style algorithm that refines this estimate to accurately recover A^*.

Stage 1: Initialization

The first stage of our approach iteratively estimates the columns of A^* (up to sign flips) in a manner similar to (Arora et al. 2015). However, their initialization algorithm incurs severe computational costs in terms of running time. More precisely, the expected value of the running time is $\widetilde{\Omega}(mn^2 p)$, which is unrealistic for large m and n.

In contrast, we leverage the double-sparsity assumption in our generative model to obtain a more efficient approach. The key ingredient of our method is a novel spectral procedure that gives us an estimate of the column supports purely from the observed samples. The full algorithm, that we call *Truncated Pairwise Reweighting*, is listed in pseudocode form as Algorithm 1.

We first state a theoretical result characterizing the performance of Algorithm 1.

Algorithm 1 Truncated Pairwise Reweighting

Initialize $L = \emptyset$

Randomly divide p samples into two disjoint sets $\mathcal{P}_1$ and $\mathcal{P}_2$ of sizes p_1 and p_2 respectively

While $|L| < m$. Pick u and v from $\mathcal{P}_1$ at random

For every $l = 1, 2, \ldots, n$, compute

$$\widehat{e}_l = \frac{1}{p_2} \sum_{i=1}^{p_2} \langle y^{(i)}, u\rangle \langle y^{(i)}, v\rangle (y_l^{(i)})^2$$

Sort $\widehat{e}_1, \widehat{e}_2, \ldots, \widehat{e}_n$ in descending order

If $\widehat{e}_{(r)} \geq \Omega(k/mr) \wedge \widehat{e}_{(r+1)}/\widehat{e}_{(r)} < O^*(r/\log^2 n)$

Let $\widehat{R}$ be set of the r largest entries of $\widehat{e}$

$\widehat{M}_{u,v} = \frac{1}{p_2} \sum_{i=1}^{p_2} \langle y^{(i)}, u\rangle \langle y^{(i)}, v\rangle y^{(i)}_{\widehat{R}} (y^{(i)}_{\widehat{R}})^T$

$\delta_1, \delta_2 \leftarrow$ top singular values of $\widehat{M}_{u,v}$

$z_{\widehat{R}} \leftarrow$ top singular vector of $\widehat{M}_{u,v}$

If $\delta_1 \geq \Omega(k/m)$ and $\delta_2 < O^*(k/m \log n)$

If dist$(\pm z, l) > 1/\log n$ for any $l \in L$

Update $L = L \cup \{z\}$

Return $A^0 = (L_1, \ldots, L_m)$

Theorem 2. *Suppose that Assumptions* **B1-B4** *hold and Assumptions* **A1-A4** *hold with parameters* $\mu = O^*\big(\frac{\sqrt{n}}{k \log^3 n}\big)$, $k = O^*\big(\frac{\sqrt{n}}{\log n}\big)$ *and* $r = o(\log^2 n)$. *Then, with high probability, Algorithm 1 returns an initial estimate* A^0 *whose columns share the same support as* A^* *and is* $(\delta, 2)$*-near to* A^* *with* $\delta = O^*(1/\log n)$ *if* $p_1 = \widetilde{\Omega}(m)$ *and* $p_2 = \widetilde{\Omega}(mr)$.

The formal proof is available in our extended version (Nguyen, Wong, and Hegde 2017). To provide some intuition about the working of the algorithm (and proof of Theorem 2), let us consider the setting where we have access to infinitely many samples. Of course, this setting is fictional. However, the analysis of this case is much simpler since we can deal with expected values rather than empirical averages. Moreover, the analysis reveals several key lemmas, which we will reuse extensively for proving Theorem 2.

First, we give some intuition behind the definition of the "scores", $\widehat{e}_l$. Fix a sample $y = A^* x^* + \varepsilon_y$ from the available training set, and consider two other samples

$$u = A^* \alpha + \varepsilon_u, \;\; v = A^* \alpha' + \varepsilon_v.$$

Consider the (very coarse) estimate for the sparse code of u with respect to A^*:

$$\beta = A^{*T} u = A^{*T} A^* \alpha + A^{*T} \varepsilon_u.$$

As long as A^* is incoherent enough and $\varepsilon_u, \varepsilon_y$ is small, the estimate β "looks" like α in the following sense:

$$\langle y, u\rangle \approx \langle x^*, \beta\rangle \approx \langle x^*, \alpha\rangle.$$

Moreover, the above inner products are large only if α and x^* share some elements in their supports; else, they are likely to be small. Likewise, the weight $\langle y, u\rangle\langle y, v\rangle$ is large only when x^* shares common elements with both α and α'. The following lemma leverages this intuition; given sufficiently many samples, $\widehat{e}_l$ gives an indicator of how large the "overlap" between α and α' is.

Lemma 1. *Fix samples* u *and* v. *Suppose that* $y = A^* x^* + \varepsilon$ *is a random sample independent of* u *and* v, *whose codes* α *and* α' *have supports* U *and* V *respectively. Then*

$$e_l \triangleq \mathbb{E}[\langle y, u\rangle \langle y, v\rangle y_l^2] = \sum_{i \in U \cap V} q_i c_i \beta_i \beta_i' A_{li}^{*2} + E,$$

where $q_i = \mathbb{P}[i \in S]$, $q_{ij} = \mathbb{P}[i, j \in S]$ *and* $c_i = \mathbb{E}[x_i^4 | i \in S]$. *Also,* E *has absolute value* $O^*(k/m \log^2 n)$ *w.h.p.*

Now, suppose for a moment that u and v share exactly one common atom in their codes, i.e., $U \cap V = \{i\}$. Lemma 1 suggests that e_l is proportional to A_{li}^{*2}; therefore, the scores e_l corresponding to the r largest coefficients of the shared atom will dominate the rest. This lets us isolate the support, R, of the shared atom. We still need a mechanism to estimate its non-zero coefficients. This is handled in the following two Lemmas, which shows that the spectrum of a certain (truncated) weighted covariance matrix reveals this information. This step is reminiscent of covariance-thresholding methods for sparse PCA (Johnstone and Lu 2004; Deshpande and Montanari 2014), and distinguishes our approach from that in (Arora et al. 2015).

Lemma 2. *The truncated re-weighting matrix obeys:*

$$M_{u,v}^R \triangleq \mathbb{E}[\langle y, u\rangle \langle y, v\rangle y_R y_R^T] = \sum_{i \in U \cap V} q_i c_i \beta_i \beta_i' A_{R,i}^* A_{R,i}^{*T} + E',$$

where E' *have spectrum norm at most* $O^*(k/m \log n)$ *w.h.p.*

Lemma 3. *If* $U \cap V = \{i\}$, *then the* r *largest entries of* e_l *are of magnitude at least* $\Omega(k/mr)$ *and are supported on* R. *Moreover, the top singular vector of* $M_{u,v}^R$ *is* δ*-close to* $A_{R,i}^*$ *for* $\delta = O^*(1/\log n)$.

Using the same argument for bounding E in Lemma 1, we can see that $M_0 \triangleq q_i c_i \beta_i \beta_i' A_{R,i}^* A_{R,i}^{*T}$ has norm at least $\Omega(k/m)$ when u and v share a unique element i. Therefore, the spectral norm of M_0 dominates those of the perturbation term E'. Thus, given R, we can use the first singular vector of $M_{u,v}^R$ as an estimate of $A_{R,i}^*$.

The question remains when and how we can certify that u and v share a unique single element in the support of their code vectors. Fortunately, this condition can be confirmed by checking the decay of the singular values of the (truncated) covariance matrix. This is quantified as follows.

Lemma 4. *If the top singular value of* $M_{u,v}$ *is at least* $\Omega(k/m)$ *and the second largest one is at most* $O^*(k/m \log n)$, *then* u *and* v *share a unique dictionary element with high probability.*

The above discussion assumes infinitely many available samples. However, we can derive analogous finite-sample lemmas which hold w.h.p. via concentration arguments. See the appendix for details. Similar to (Arora et al. 2015), our

Algorithm 2 Double-Sparse Coding Descent Algorithm

Initialize A^0 is $(\delta, 2)$-near to A^*. $H = (h_{ij})_{n\times m}$ where $h_{ij} = 1$ if $i \in \text{supp}(A^0_{\bullet j})$ and 0 otherwise.
Repeat for $s = 0, 1, \ldots, T$
Encode: $x^{(i)} = \text{threshold}_{C/2}((A^s)^T y^{(i)})$
Update: $A^{s+1} = \mathcal{P}_H(A^s - \eta\widehat{g}^s) = A^s - \eta\mathcal{P}_H(\widehat{g}^s)$, where $\widehat{g}^s = \frac{1}{p}\sum_{i=1}^{p}(A^s x^{(i)} - y^{(i)})\text{sgn}(x^{(i)})^T$ and $\mathcal{P}_H(G) = H \circ G$

algorithm requires $\widetilde{O}(m)$ iterations to estimate all the atoms, and hence the expected running time is $\widetilde{O}(mnp)$.

Lemma 3 indicates that the support, as well as a coarse (δ-close) estimate, of each column of A^* can be estimated using our proposed initialization method. We now show how to refine this estimate using a descent-style method.

Stage 2: Descent

We adapt the neural sparse coding approach of (Arora et al. 2015) to obtain an improved estimate of A^*. As mentioned earlier, at a high level the algorithm is akin to performing approximate gradient descent. The insight is that within a small enough neighborhood (in the sense of δ-closeness) of the true A^*, an estimate of the ground-truth code vectors, X^*, can be constructed using a neurally plausible algorithm. It can be used to construct a noisy approximate gradient $\widehat{g}^s$.

The innovation, in our case, is the double-sparsity model since we know *a priori* that A^* is itself sparse. Under sufficiently many samples, the support of A^* can be deduced from the initialization stage; therefore we perform an extra *projection* step in each iteration of gradient descent. In this sense, our method is non-trivially different from (Arora et al. 2015). The full algorithm is presented as Algorithm 2.

As discussed in the Setup section above, convergence of noisy approximate gradient descent can be achieved as long as $\widehat{g}^s$ is correlated-w.h.p. with the true solution. However, an analogous convergence result for projected gradient descent does not exist in the literature. We fill this gap via a careful analysis. Due to the projection, we only require the correlated-w.h.p. property for a *part* of $\widehat{g}^s$ with A^* when it is restricted to some support set. The descent property is still achieved via Theorem 3. Due to the various perturbation terms, $\widehat{g}$ is only a biased estimate of $\nabla_A\mathcal{L}(A, X)$; therefore, we can only refine the estimate of A^* until the column-wise error is of the order of $O(\sqrt{k/n})$. The performance of Algorithm 2 can be characterized via the following theorem.

Theorem 3. *Suppose that the initial estimate A^0 has the correct column supports and is $(\delta, 2)$-near to A^* with $\delta = O^*(1/\log n)$. If Algorithm 2 is provided with $p = \widetilde{\Omega}(m + \sigma_\varepsilon^2\frac{mnr}{k})$ samples at each step and $\eta = \Theta(m/k)$, then*

$$\mathbb{E}[\|A^s_{\bullet i} - A^*_{\bullet i}\|^2] \le (1-\rho)^s\|A^0_{\bullet i} - A^*_{\bullet i}\|^2 + O(k/n)$$

for some $0 < \rho < 1/2$ and for $s = 1, 2, \ldots, T$. Consequently, A^s converges to A^* geometrically until column-wise error is $O\big(\sqrt{k/n}\big)$.

The formal proof of Theorem 3 is available in our extended version (Nguyen, Wong, and Hegde 2017). Here, we shed some light on the analysis techniques by studying the case of infinite samples. Therefore, the estimate $\widehat{g}^s$ can be replaced by its expectation,

$$g^s \triangleq \mathbb{E}[(A^s x - y)\text{sgn}(x)^T].$$

Let us focus on the i^{th} column. Given the knowlege of the support R of $A^*_{\bullet i}$, we only have to restrict our focus to $g^s_{R,i}$. A key component is to establish the $(\alpha, \beta, \gamma_s)$-correlation of g^s_{Ri} with $A^*_{R,i}$ so as to obtain a descent property, similar to Theorem 3, for infinite number of samples. To this end, we establish the following lemma, using the same strategy as in (Arora et al. 2015).

Lemma 5. *Suppose that the initial estimate A^0 has the correct column supports and is $(\delta, 2)$-near to A^* with $\delta = O^*(1/\log n)$. The update is of the form $g^s_{R,i} = p_i q_i(\lambda^s_i A^s_{R,i} - A^*_{R,i} + \xi^s_i \pm \zeta)$ where $R = \text{supp}(A^*_{\bullet i})$ and*

$$\xi^s_i = A^s_{R,-i}\text{diag}(q_{ij})(A^s_{\bullet -i})^T A^*_{\bullet i}/q_i$$

and $\lambda^s_i = \langle A^s_{\bullet i}, A^*_{\bullet i}\rangle$. In addition, $\|\xi^s_i\| \le O(k/n)$ and ζ is negligible.

Intuitively, Lemma 5 suggests that $g^s_{R,i}$ is almost equal to $p_i q_i(A^s_{R,i} - A^*_{R,i})$ (since $\lambda^s_i \approx 1$), which is a desired direction. Then, we can prove the correlation and descent results accordingly:

Lemma 6. *If A^s is $(\delta, 2)$-near to A^* with $\delta = O^*(1/\log n)$ and $R = \text{supp}(A^*_{\bullet i})$, then $2g^s_{R_i,i}$ is $(\alpha, 1/2\alpha, \epsilon^2/\alpha)$-correlated with $A^*_{R,i}$ by*

$$\langle 2g^s_{R,i}, A^s_{R,i} - A^*_{R,i}\rangle \ge \alpha\|A^s_{R,i} - A^*_{R,i}\|^2 + 1/(2\alpha)\|g^s_{R,i}\|^2 - \epsilon^2/\alpha$$

where $\epsilon = O(k^2/mn)$. In particular, $g^s_{R,i}$ is (α, β, γ)-correlated with $A^*_{R,i}$ for $\alpha = \Omega(k/m), \beta = \Omega(m/k)$ and $\gamma = O(k^3/mn^2)$.

After the results under infinite samples are achieved, we study the concentration of the empirical average $\widehat{g}^s$ to its mean. Again, due to the knowledge of the column supports, for each column of $\widehat{g}^s$, we only have to establish such concentration over a r-dimensional sub-vector. This helps to achieve a better sample complexity especially when r is small. To sum up, the respective sample complexities for the descent and the initialization stage are $\widetilde{O}(m + \sigma_\varepsilon^2\frac{mnr}{k})$ and $\widetilde{O}(mr)$. Overall, the sample complexity $\widetilde{O}(mr + \sigma_\varepsilon^2\frac{mnr}{k})$ sufficiently guarantees the success of our approach.

Regarding the running time, the running time per iteration of Algorithm 2 is $O(m\max(k, r)p)$ due to the sparsity of both A and x. The main botteneck is at the initialization stage with the expected running time is $\widetilde{O}(mnp)$. Consequently, the total computational complexity of our approach is $\widetilde{O}(mnp)$.

Empirical Study

We compare our method with three different methods for both standard sparse and double-sparse coding. For the standard approach, we implement the algorithm proposed in

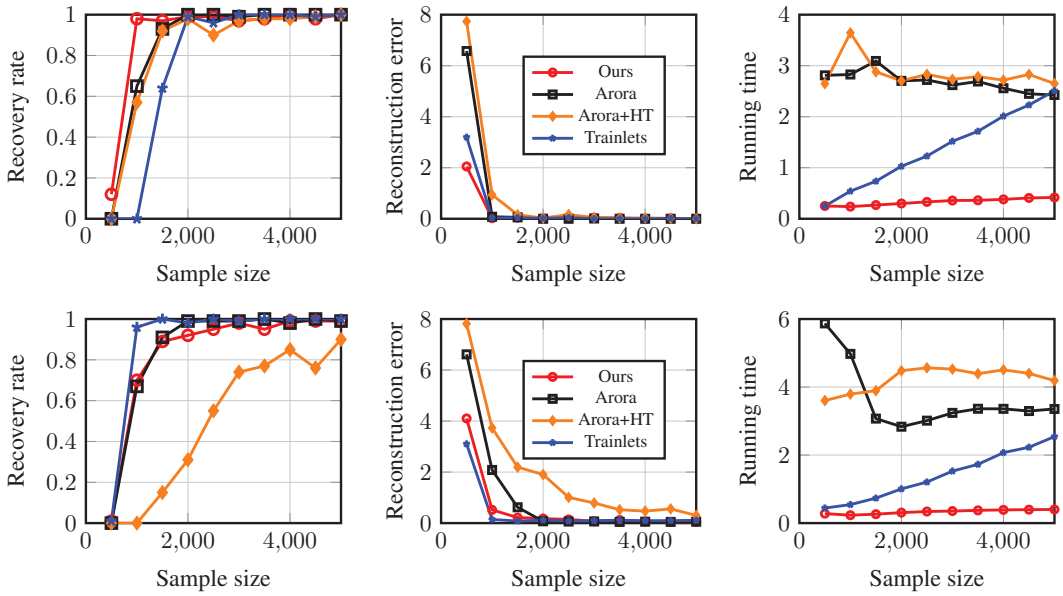

Figure 1: (top) The performance of four methods on three metrics (recovery rate, reconstruction error and running time) in sample size in the noiseless case. (bottom) The same metrics are measured for the noisy case.

(Arora et al. 2015), which currently is the best theoretically sound method for provable sparse coding. However, since their method does not explicitly leverage the double-sparsity model, we also implement a heuristic modification that performs a hard thresholding (HT)-based post-processing step in the initialization and learning procedures (which we dub *Arora + HT*). The final comparison is the *Trainlets* approach of (Sulam et al. 2016).

We generate a synthetic training dataset according to the model described in the Setup. The base dictionary Φ is the identity matrix of size $n = 64$ and the square synthesis matrix A^* is a block diagonal matrix with 32 blocks. Each 2×2 block is of form $[1\ 1; 1\ -1]$ (i.e., the column sparsity $r = 2$) . The support of x^* is drawn uniformly over all 6-dimensional subsets of $[m]$, and the nonzero coefficients are randomly set to ± 1 with equal probability. In our simulations with noise, we add Gaussian noise ε with entrywise variance $\sigma_\varepsilon^2 = 0.01$ to each of those above samples.

For all the approaches except Trainlets, we use $T = 2000$ iterations for the initialization procedure, and set the number of steps in the descent stage to 25. Since Trainlets does not have a specified initialization procedure, we initialize it with a random Gaussian matrix upon which column-wise sparse thresholding is then performed. The learning step of Trainlets[3] is executed for 50 iterations, which tolerates its initialization deficiency. For each Monte Carlo trial, we uniformly draw p samples, feed these samples to the four different algorithms, and observe their ability to reconstruct A^*. Matlab implementation of our algorithms is available online[4].

[3]We use the implementation of Trainlets's provided at http://jsulam.cswp.cs.technion.ac.il/home/software/.

[4]https://github.com/thanh-isu/double-sparse-coding

We evaluate these approaches on three metrics as a function of the number of available samples: (i) fraction of trials in which each algorithm successfully recovers the ground truth A^*; (ii) reconstruction error; and (iii) running time. The synthesis matrix is said to be "successfully recovered" if the Frobenius norm of the difference between the estimate $\widehat{A}$ and the ground truth A^* is smaller than a threshold which is set to 10^{-4} in the noiseless case, and to 0.5 in the other. All three metrics are averaged over 100 Monte Carlo simulations. As discussed above, the Frobenius norm is only meaningful under a suitable permutation and sign flip transformation linking $\widehat{A}$ and A^*. We estimate this transformation using a simple maximum weight matching algorithm. Specifically, we construct a weighted bipartite graph with nodes representing columns of A^* and $\widehat{A}$ and adjacency matrix defined as $G = |A^{*T}\widehat{A}|$, where $|\cdot|$ is taken element-wise. We compute the optimal matching using the Hungarian algorithm, and then estimate the sign flips by looking at the sign of the inner products between the matched columns.

The results of our experiments are shown in Figure 1 with the top and bottom rows respectively for the noiseless and noisy cases. The two leftmost figures suggest that all algorithms exhibit a "phase transitions" in sample complexity that occurs in the range of 500-2000 samples. In the noiseless case, our method achieves the phase transition with the fewest number of samples. In the noisy case, our method nearly matches the best sample complexity performance (next to Trainlets, which is a heuristic and computationally expensive). Our method achieves the best performance in terms of (wall-clock) running time in all cases. To conclude, simulation suggests that our method enjoys the sample efficiency, robustness and computational efficacy, and strongly supports our theory.

References

Agarwal, A.; Anandkumar, A.; Jain, P.; Netrapalli, P.; and Tandon, R. 2014. Learning sparsely used overcomplete dictionaries. In *Conference on Learning Theory*, 123–137.

Aharon, M.; Elad, M.; and Bruckstein, A. 2006. rmk-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing* 54(11):4311–4322.

Arora, S.; Ge, R.; Ma, T.; and Moitra, A. 2015. Simple, efficient, and neural algorithms for sparse coding. In *Conference on Learning Theory*, 113–149.

Arora, S.; Ge, R.; and Moitra, A. 2014. New algorithms for learning incoherent and overcomplete dictionaries. In *Conference on Learning Theory*, 779–806.

Boureau, Y.-L.; Bach, F.; LeCun, Y.; and Ponce, J. 2010. Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2559–2566. IEEE.

Deshpande, Y., and Montanari, A. 2014. Sparse pca via covariance thresholding. In *Advances in Neural Information Processing Systems*, 334–342.

Elad, M., and Aharon, M. 2006. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing* 15(12):3736–3745.

Engan, K.; Aase, S. O.; and Husoy, J. H. 1999. Method of optimal directions for frame design. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 5, 2443–2446. IEEE.

Gregor, K., and LeCun, Y. 2010. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 399–406.

Gribonval, R.; Jenatton, R.; Bach, F.; Kleinsteuber, M.; and Seibert, M. 2015. Sample complexity of dictionary learning and other matrix factorizations. *IEEE Transactions on Information Theory* 61(6):3469–3486.

Gribonval, R.; Jenatton, R.; and Bach, F. 2015. Sparse and spurious: dictionary learning with noise and outliers. *IEEE Transactions on Information Theory* 61(11):6298–6319.

Jenatton, R.; Gribonval, R.; and Bach, F. 2012. Local stability and robustness of sparse dictionary learning in the presence of noise. *arXiv preprint arXiv:1210.0685*.

Johnstone, I. M., and Lu, A. Y. 2004. Sparse principal components analysis. *Unpublished manuscript* 7.

Krim, H.; Tucker, D.; Mallat, S.; and Donoho, D. 1999. On denoising and best signal representation. *IEEE Transactions on Information Theory* 45(7):2225–2238.

Mairal, J.; Bach, F.; Ponce, J.; and Sapiro, G. 2009. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, 689–696.

Mazumdar, A., and Rawat, A. S. 2017. Associative memory using dictionary learning and expander decoding. In *AAAI*, 267–273.

Nguyen, T.; Wong, R. K. W.; and Hegde, C. 2017. A provable approach for double-sparse coding. *arXiv preprint arXiv:1711.03638*.

Olshausen, B. A., and Field, D. J. 1997. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research* 37(23):3311–3325.

Rubinstein, R.; Bruckstein, A. M.; and Elad, M. 2010. Dictionaries for sparse representation modeling. *Proceedings of the IEEE* 98(6):1045–1057.

Rubinstein, R.; Zibulevsky, M.; and Elad, M. 2010. Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on Signal Processing* 58(3):1553–1564.

Spielman, D. A.; Wang, H.; and Wright, J. 2012. Exact recovery of sparsely-used dictionaries. In *Conference on Learning Theory*, 37–1.

Sulam, J.; Ophir, B.; Zibulevsky, M.; and Elad, M. 2016. Trainlets: Dictionary learning in high dimensions. *IEEE Transactions on Signal Processing* 64(12):3180–3193.

Sun, J.; Qu, Q.; and Wright, J. 2015. Complete dictionary recovery using nonconvex optimization. In *Proceedings of the 33rd International Conference on Machine Learning*, 2351–2360.

Wang, L.; Zhang, X.; and Gu, Q. 2016. A unified computational and statistical framework for nonconvex low-rank matrix estimation. *arXiv preprint arXiv:1610.05275*.

Zhang, Y.; Chen, X.; Zhou, D.; and Jordan, M. I. 2016. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. *The Journal of Machine Learning Research* 17(1):3537–3580.