On the Convergence of Group-Sparse Autoencoders

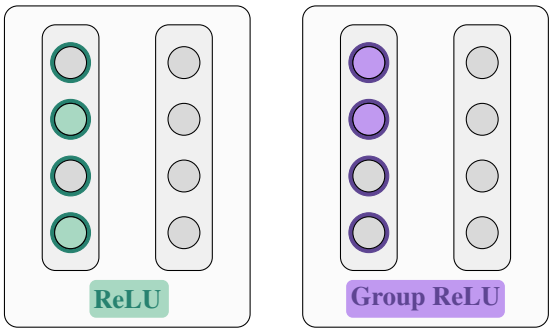
Emmanouil Theodosis [1] **Bahareh Tolooshams** [*1] **Pranay Tankala** [*1] **Abiy Tasissa** [2] **Demba Ba** [1]

Abstract

Recent approaches in the theoretical analysis of model-based deep learning architectures have studied the convergence of gradient descent in shallow ReLU networks that arise from generative models whose hidden layers are sparse. Motivated by the success of architectures that impose structured forms of sparsity, we introduce and study a group-sparse autoencoder that accounts for a variety of generative models, and utilizes a group-sparse ReLU activation function to force the nonzero units at a given layer to occur in blocks. For clustering models, inputs that result in the same group of active units belong to the same cluster. We proceed to analyze the gradient dynamics of a shallow instance of the proposed autoencoder, trained with data adhering to a group-sparse generative model. In this setting, we theoretically prove the convergence of the network parameters to a neighborhood of the generating matrix. We validate our model through numerical analysis and highlight the superior performance of networks with a group-sparse ReLU compared to networks that utilize traditional ReLUs, both in sparse coding and in parameter recovery tasks. We also provide real data experiments to corroborate the simulated results, and emphasize the clustering capabilities of structured sparsity models.

1. Introduction

Model-based learning approaches (Bora et al., 2017; Simon & Elad, 2019; Shlezinger et al., 2020; Tasissa et al., 2020) address one of the fundamental problems of deep learning; that is, current high-performing architectures lack explainability. The model-based learning paradigm addresses this by invoking domain knowledge in order to constrain the neural architectures, thus making them amenable to interpretation. Within this context, *unfolded networks* (Hershey et al., 2014; Tolooshams et al., 2018; Monga et al., 2020), popularized by the seminal work of Gregor & LeCun (2010), unroll the steps of iterative optimization algorithms to form a neural network. Compared to their unconstrained counterparts, these structured architectures, which combine ideas from the signal processing and deep learning communities, enjoy a de facto reduction in the number of trainable parameters while maintaining competitive performance (Simon & Elad, 2019; Tolooshams et al., 2020b).



(a) Network with *ReLU* activation.

(b) Network with *Group ReLU* activation.

Figure 1: In sparsity-promoting models (**left**), arbitrary neurons can activate. In contrast, in group-sparse networks (**right**), the activated neurons are structured.

LISTA (Gregor & LeCun, 2010), which enforces sparsity on the units of deep layers, is based on the unfolding of the Iterative Shrinkage Thresholding Algorithm (ISTA), a sparse coding optimization algorithm. Sparsity-focused generative models (Tibshirani, 1996) are most frequently employed due to their experimentally and theoretically proven generalization power (Mairal et al., 2009; Mehta & Gray, 2013). In addition, because they can significantly reduce the number of nonzero coefficients—units active at a given layer—sparse models have also been used to speed up inference in deep neural networks (Liu et al., 2015; Sze et al., 2020). Recent research has deviated from the traditional sparse coding model; certain works reconsidered the sparsity-promoting minimization (Huang & Tran, 2018), where others focused on exploring different generative models (Scardapane et al., 2017; Narang et al., 2017; del Aguila Pla & Jaldén, 2018). Within the latter class, works studying *group sparsity* (Yuan

[*]Equal contribution [1]School of Engineering and Applied Sciences, Harvard University, Cambridge, MA [2]Department of Mathematics, Tufts University, Medford, MA. Correspondence to: Emmanouil Theodosis <etheodosis@g.harvard.edu>.

& Lin, 2006; Eldar et al., 2010) have been rather prolific. In addition to minimizing the number of non-zero coefficients, group sparsity forces them to occur in blocks (see Figure 1). As we argue in the sequel, we can interpret inputs that share active groups as belonging to the same class or cluster. The groupings manifest themselves either as a direct arrangement of the hidden units of neural networks into blocks (Wen et al., 2016; Yoon & Hwang, 2017), or as a clustering of data that, a priori, share similar characteristics, such as patches of natural images (Lecouat et al., 2020). Enforcing group structure has proved practical in applications and outperforms approaches based on the traditional notion of sparsity.

Despite the success of model-based unfolded architectures, their theoretical analysis is still nascent. The theoretical convergence of sparsity-based generative models has been studied in a *supervised* setting, where data and their sparse representations are readily available (Chen et al., 2018; Liu et al., 2019). The majority of recent work, motivated by the general framework for the convergence of *unsupervised* autoencoders introduced by Arora et al. (2015), provide guarantees under which gradient descent will converge. For example, Rangamani et al. (2018) theoretically prove that the gradient of the loss function vanishes around critical points, and introduce a proxy gradient to approximate the true expectation. Most recently, Nguyen et al. (2019) showed that a multitude of shallow models (derived from sparse coding) conform to the framework and converge to the generative model when trained with gradient descent. However, all of the existing literature is limited to models that rely on traditional notions of sparsity and, thus, does not account for generative models with structured notions of sparsity, such as ones with group-structured activations.

Our work poses the following question: given data generated according to a group-sparse model, *can a shallow, group-sparse autoencoder recover the generating matrix, as well as nonzero blocks and their activations?* We answer this question in the affirmative, and our overall contributions are summarized as follows:

Introduction of group-sparse autoencoders. Motivated by the connection between clustering and group-sparsity, we introduce a *group-sparse* autoencoder architecture, that we subsequently analyze theoretically and experimentally.

Convergence of gradient descent. Under mild assumptions introduced in Section 3, we prove that shallow group-sparse autoencoders, when trained with gradient descent, converge to a *neighborhood* of the generative model.

Recovering cluster membership. In Section 3 we show that the encoder's output demonstrably identifies nonzero blocks and their activations. Through the connection we establish between group-sparsity and clustering (Section 2), this result suggests that, under mild conditions, we can recover cluster membership in unions of subspaces models. Finally, we showcase the clustering structure of group-sparsity in our experimental section.

Group-sparse ReLU outperforms ReLU. Through our experimental validation in Section 4, we demonstrate that a network with group-sparse activations shows superior performance compared to networks that promote traditional notions of sparsity. In particular, in both simulated and real data experiments, the group-sparse autoencoder outperforms a sparse autoencoder, both in recovering generative models and in learning interpretable dictionaries.

In what follows we denote matrices with capital bold-faced letters, vectors with lowercase bold-faced letters, and scalars with lowercase letters. $\boldsymbol{A}$ is a matrix, $\boldsymbol{A}_S$ is a sub-matrix of $\boldsymbol{A}$ indexed by S, $\boldsymbol{a}_i$ is the i-th column of $\boldsymbol{A}$, and a_{ij} is the element in its i-th row and j-th column. $\|\boldsymbol{x}\|_p$ denotes the ℓ_p norm of $\boldsymbol{x}$ and $\|\boldsymbol{A}\|_2, \|\boldsymbol{A}\|_F$ denote the spectral and Frobenius norms, respectively, of $\boldsymbol{A}$, while $\sigma_1(\boldsymbol{A})$ denotes its maximum singular value. Finally, $[N] = \{1, \dots, N\}$.

2. Group-sparsity in dictionary learning

In model-based approaches the observed data $\{\boldsymbol{y}_i\}_{i=1}^N \in \mathcal{Y}$[1] are assumed to adhere to a generative model. Formally, we assume that the data satisfy

$$\boldsymbol{y}_i = f_{\theta^*}(\boldsymbol{x}_i^*), \tag{1}$$

where $\boldsymbol{x}_i^* \in \mathcal{X}$ is a latent vector and f_θ, parametrized by θ, comes from a function class $\mathcal{F}$ that describes the relation between the data $\boldsymbol{y}_i$ and the latent variables $\boldsymbol{x}_i^*$. Most frequent are models of *linear* relations, where the function f_θ is of the form $f_\theta \colon \boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$, parametrized by $\theta = \{\boldsymbol{A}\}$.

2.1. Group-sparse generative model

Consider a generative model where each observation[2] $\boldsymbol{y}$ belongs to the union of one, or more, subspaces (Gribonval & Nielsen, 2003). In this general group-sparse model the observed data satisfy

$$\boldsymbol{y} = \boldsymbol{A}^*\boldsymbol{x}^* = \sum_{g \in S} \boldsymbol{A}_g^* \boldsymbol{x}_g^*, \tag{2}$$

where $S \subset [\Gamma]$ denotes the *group support* (i.e. which of the Γ groups are active), and the latent vector has the form $\boldsymbol{x}^* = [\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \dots, \boldsymbol{x}_\Gamma^*]^T$. Gaussian mixture models, sparse models, and nonnegative sparse models (Nguyen et al., 2019) can readily be derived as special cases of the highly-expressive generative model from (2). The group-sparse

[1] We intentionally do not write $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$, as the setting we are considering is *strictly* unsupervised.

[2] For the rest of the text, we drop the index i to reduce clutter.

prior assumes that the latent representation $\boldsymbol{x}$ is sparse, and that in nonzero entries occur in blocks (groups). The model also implies a decomposition of $\boldsymbol{A}^*$ into sub-matrices $\boldsymbol{A}_1^*, \boldsymbol{A}_2^*, \ldots, \boldsymbol{A}_\Gamma^*$ such that $\boldsymbol{A}^* = [\boldsymbol{A}_1^* \boldsymbol{A}_2^* \ldots \boldsymbol{A}_\Gamma^*]$, where we assume that each group $\boldsymbol{A}_g^*$ has exactly d elements. Without additional structure, the generative model may not yield a unique solution; for example, Eldar et al. (2010) impose orthonormality on $\boldsymbol{A}_g^*$ to ensure uniqueness.

An analogue to the *coherence* of a dictionary in sparse models (defined as $\mu = \max_{i \neq j} |\boldsymbol{a}_i^{*T} \boldsymbol{a}_j^*|$; the inner-product with the largest magnitude in $\boldsymbol{A}^*$) is the *block coherence* of $\boldsymbol{A}^*$

$$\mu_B = \max_{g \neq h} \frac{1}{d} \|\boldsymbol{A}_g^{*T} \boldsymbol{A}_h^*\|_2. \tag{3}$$

Intuitively, coherence metrics give a sense of how correlated the different columns, or groups, of $\boldsymbol{A}^*$ are and directly affect the ability to recover latent vectors. Assuming normalized groups, as we will in this work, it holds that $0 \leq \mu_B \leq \mu \leq 1$.

2.2. Group-sparse dictionary learning

Assuming a linear underlying generative model, dictionary learning (Aharon et al., 2006; Mairal et al., 2012) sets out to learn a dictionary $\boldsymbol{A}$ such that every vector $\boldsymbol{y}$ in a data set adopts a sparse representation as a linear combination of the columns of $\boldsymbol{A}$ using a vector $\boldsymbol{x}$. In group-sparse settings, given the dictionary $\boldsymbol{A}$, *group-sparse coding* lets us find $\boldsymbol{x}$ as the solution to the optimization problem

$$\min_{\boldsymbol{x} \in \mathcal{X}} \quad \|\boldsymbol{x}\|_{\ell_0/\ell_2}, \qquad \text{s.t. } \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}. \tag{4}$$

The ℓ_0/ℓ_2, expressed as the ℓ_0 pseudo-norm of the vector of ℓ_2 norms $[\|\boldsymbol{x}_1\|_2, \|\boldsymbol{x}_2\|_2, \ldots, \|\boldsymbol{x}_\Gamma\|_2]^T$, norm minimizes the number of active groups. The combinatorial nature of ℓ_0 pseudo-norm makes this optimization intractable in practice. A popular approach utilizes the ℓ_1 norm instead, as a tractable convex relaxation of the optimization of (4), yielding

$$\min_{\boldsymbol{x} \in \mathcal{X}} \quad \|\boldsymbol{x}\|_{\ell_1/\ell_2}, \qquad \text{s.t. } \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \tag{5}$$

where $\|\boldsymbol{x}\|_{\ell_1/\ell_2} = \sum_{g \in S} \|\boldsymbol{x}_g\|_2$. Both the optimizations of (4) and (5) require the recovery of latent codes $\boldsymbol{x}$ that lead to an exact reconstruction of the data $\boldsymbol{y}$. The following *unconstrained* optimization problem enables a trade-off between exact recovery and the group-sparsity of the latent codes

$$\min_{\boldsymbol{x} \in \mathcal{X}} \quad \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \sum_{g \in S} \|\boldsymbol{x}_g\|_2. \tag{6}$$

Optimization objectives of the form $\frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \Omega(\boldsymbol{x})$ have been studied extensively in the literature and can be directly solved via the theory of proximal operators (Bach et al., 2011; Parikh & Boyd, 2014). Pertinent to the current discussion, the proximal operator promoting group-sparse structures can be derived as

$$\sigma_\lambda(\boldsymbol{x}_g) = \left(1 - \frac{\lambda}{\|\boldsymbol{x}_g\|_2}\right)_+ \boldsymbol{x}_g, \tag{7}$$

where $(\cdot)_+ = \max(\cdot, 0)$. Note that this proximal operator bears a striking similarity to $\mathrm{ReLU}(x) = \max(x, 0)$. Indeed, we can consider (7) as a generalization of ReLU (informally termed "Group ReLU"), where the thresholding is applied in a structured way, instead of an element-wise fashion. Dictionary learning can then be performed by solving

$$(\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{x}}) = \operatorname*{arg\,min}_{\boldsymbol{A} \in \mathcal{A}, \boldsymbol{x} \in \mathcal{X}} \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \sum_{g \in S} \|\boldsymbol{x}_g\|_2, \tag{8}$$

a nonconvex optimization problem. A popular approach, termed *alternating minimization* (Tseng, 1991; Agarwal et al., 2016), cycles between group-sparse coding and dictionary update steps. Formally, the group-sparse coding step considers the dictionary $\widehat{\boldsymbol{A}}^k$ fixed and solves

$$\widehat{\boldsymbol{x}}^{k+1} = \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathcal{X}} \quad \frac{1}{2} \|\boldsymbol{y} - \widehat{\boldsymbol{A}}^k \boldsymbol{x}\|_2^2 + \lambda \sum_{g \in S} \|\boldsymbol{x}_g\|_2, \tag{9}$$

followed by an optimization to find the optimal dictionary $\widehat{\boldsymbol{A}}^{k+1}$ given an estimate of the latent code $\widehat{\boldsymbol{x}}^{k+1}$

$$\widehat{\boldsymbol{A}}^{k+1} = \operatorname*{arg\,min}_{\boldsymbol{A} \in \mathcal{A}} \quad \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}^{k+1}\|_2^2, \tag{10}$$

where (9) and (10) are performed in an alternating manner until convergence, yielding $(\widehat{\boldsymbol{A}}^{\text{OPT}}, \widehat{\boldsymbol{x}}^{\text{OPT}})$.

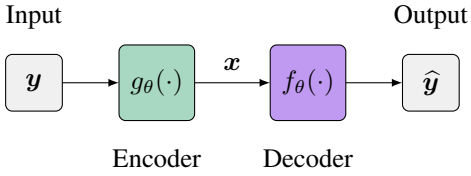


Figure 2: Diagram of an autoencoder architecture.

3. Group-sparse autoencoders and descent

3.1. Group-sparse autoencoder

The proximal operator of (7) implies an iterative optimization algorithm to recover the group-sparse codes $\boldsymbol{x}^*$. We unfold this optimization scheme into an autoencoder architecture (Gregor & LeCun, 2010; Tolooshams et al., 2020b), illustrated in Figure 2, which implicitly recovers the sparse codes. Formally, we denote the encoder as a function $g_\theta \colon \mathcal{R}^n \to \mathcal{R}^m$ such that

$$g_\theta(\boldsymbol{y}) = \sigma_\lambda(\boldsymbol{A}^T \boldsymbol{y}) = \boldsymbol{x}, \tag{11}$$

and the corresponding decoder $f_\theta \colon \mathcal{R}^m \to \mathcal{R}^n$

$$f_\theta(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} = \widehat{\boldsymbol{y}}. \tag{12}$$

Notice that we deliberately use the same subscript θ (here $\theta = \{\boldsymbol{A}\}$), as the parametrization is common for the two systems, i.e. the weights are *tied*. We train the architecture by minimizing the loss function[3] $\mathcal{L}(\theta) = \frac{1}{2}\|\boldsymbol{y} - \widehat{\boldsymbol{y}}\|_2^2$. We assume that the data are being sampled from the generative model indicated by (2) with $\boldsymbol{A} \in \mathcal{R}^{n\times m}$, which implies that $\boldsymbol{y} = \boldsymbol{A}^*\boldsymbol{x}^* = f_{\theta^*}(\boldsymbol{x}^*)$. Accounting for all training examples $\{\boldsymbol{y}_i\}_{i=1}^N$, we end up minimizing the cost

$$\frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(\theta) = \tfrac{1}{2N}\sum_{i=1}^{N}\|\boldsymbol{y}_i - \widehat{\boldsymbol{y}}_i\|_2^2. \tag{13}$$

The dictionary updates are performed by backpropagating through the architecture via gradient descent, yielding updates of the form

$$\boldsymbol{A}^{k+1} = \boldsymbol{A}^k - \eta\nabla_{\boldsymbol{A}^k}\mathcal{L}(\boldsymbol{A}^k). \tag{14}$$

In the next section, we will unfold a single iteration of the optimization and study a *shallow* version of this network.

3.2. Theoretical analysis

Having introduced the shallow group-sparse autoencoder in the previous subsections, we will now analyze the theoretical properties of the architecture. We will organize our analysis in two parts; first, we will derive conditions that guarantee the recovery of the group support by applying the proximal operator of (7). Then, assuming that the group memberships are correctly recovered, we will argue convergence of gradient descent by proving that the gradient of the loss function is aligned with $\boldsymbol{A}_g - \boldsymbol{A}_g^*$.

Assumptions. We will delineate our critical assumptions here, for completeness; we will try our best to indicate where in the analysis each of the assumptions is invoked, as to also highlight why such a choice was made.

1. *Bounded norms:* We assume that each group of the generating code $\boldsymbol{x}^*$ has bounded norm; i.e. for every g in S it holds that

$$B_{\min} \leq \|\boldsymbol{x}_g^*\|_2 \leq B_{\max}. \tag{15}$$

Note that this condition is significantly looser than similar conditions relating to sparse coding (Arora et al., 2015; Nguyen et al., 2019). Precisely, as we will show experimentally in Section 4, the norm condition is more easily satisfied than a direct bound on the codes of the generative model.

2. *Group-sparse support:* Each group g has size $\mathrm{card}(g) = d$ and the number of non-zero groups is at most $\mathrm{card}(S) = \gamma$; the overall number of groups is Γ. Given that $\boldsymbol{A} \in \mathcal{R}^{n\times m}$, this implies that $m = d\cdot\Gamma$. We assume that the support S is uniformly distributed, i.e. $\mathbb{P}[g \in S] = p_g = \Theta(\frac{\gamma}{\Gamma})$, $\mathbb{P}[g, h \in S] = p_{gh} = \Theta(\frac{\gamma^2}{\Gamma^2})$, etc.

3.*Group-sparse code covariance:* Given two groups in the group support $v, h \in S$, we assume it holds

$$\mathbb{E}\left[\boldsymbol{x}_v^*\boldsymbol{x}_h^{*T} \mid v, h \in S\right] = \begin{cases}\boldsymbol{I}, & v = h,\\ \boldsymbol{0}, & v \neq h.\end{cases} \tag{16}$$

4. *Model conditions:* We assume that our initialization, and therefore the subsequent updates, are relatively close to the generating matrix $\boldsymbol{A}^*$; formally, we assume that for every group $g \in S$ it holds that

$$\|\boldsymbol{A}_g - \boldsymbol{A}_g^*\|_F \leq \delta, \tag{17}$$

for some $\delta \in [0, 1]$. We also assume that the norm $\|\boldsymbol{A}_g^T\boldsymbol{A}_g^*\|_F$ is bounded, i.e.

$$\|\boldsymbol{A}_g^T\boldsymbol{A}_g^*\|_F \leq \zeta, \tag{18}$$

by some ζ that will be clarified later in this section. Finally, we assume that the weight matrix has orthonormal groups, i.e. $\boldsymbol{A}_g^T\boldsymbol{A}_g = \boldsymbol{I}$. While that assumption might seem restrictive, it is fairly common in the literature of group sparsity (Eldar et al., 2010).

Support recovery. To show that the support is correctly recovered, we will show that for two groups $g, v \in \Gamma$ with $g \in S$ and $v \notin S$ it holds that $\|\boldsymbol{A}_g^T\boldsymbol{y}\|_2 \geq \lambda$ and $\|\boldsymbol{A}_v^T\boldsymbol{y}\|_2 \leq \lambda$, for some threshold λ. Then the analysis consists of finding a suitable λ guaranteeing that applying the proximal operator of (7) will attenuate the group g, but completely block group v. To that end, we decompose $\boldsymbol{A}_g^T\boldsymbol{y}$ as follows

$$\boldsymbol{A}_g^T\boldsymbol{y} = \boldsymbol{A}_g^T\boldsymbol{A}_g^*\boldsymbol{x}_g^* + \sum_{h\neq g\in S}\boldsymbol{A}_g^T\boldsymbol{A}_h^*\boldsymbol{x}_h^*, \tag{19}$$

that is, each group $g \in S$ comprises the contribution stemming from its own generating group and the cross-contribution of the other terms in the support S. Similarly, for a group $v \notin S$, we will only have contributions from the "cross-terms", i.e. $\boldsymbol{A}_v^T\boldsymbol{y} = \sum_{h\in S}\boldsymbol{A}_v^T\boldsymbol{A}_h^*\boldsymbol{x}_h^*$. Therefore, to achieve support recovery we will show that the term $\boldsymbol{A}_g^T\boldsymbol{A}_g^*\boldsymbol{x}_g^*$ is large (in terms of norm) compared to the norm of $\sum_{h\in S}\boldsymbol{A}_v^T\boldsymbol{A}_h^*\boldsymbol{x}_h^*$ (with $v \notin S$). The following two propositions bound the relevant norms.

Proposition 1 (Group-norm lower bound)**.** *The norm of the term $\boldsymbol{A}_g^T\boldsymbol{A}_g^*\boldsymbol{x}_g^*$ is lower-bounded by*

$$\|\boldsymbol{A}_g^T\boldsymbol{A}_g^*\boldsymbol{x}_g^*\|_2 \geq B_{\min}(1-\delta). \tag{20}$$

Proposition 2 (Cross-term upper bound)**.** *The norm of the term $\sum_{h\in S}\boldsymbol{A}_v^T\boldsymbol{A}_h^*\boldsymbol{x}_h^*$ is upper-bounded by*

$$\|\sum_{h\in S}\boldsymbol{A}_v^T\boldsymbol{A}_h^*\boldsymbol{x}_h^*\|_2 \leq \gamma B_{\max}(\mu_B + \delta). \tag{21}$$

[3] Note that the loss function we are minimizing to train the autoencoder is *different* from the minimization objective of (6), and instead similar to (10).

The proofs for Propositions 1 and 2 can be found in Appendix A. Using these results, we can now lower-bound the norm of a group $g \in S$ as

$$\|\boldsymbol{A}_g^T \boldsymbol{y}\|_2 \geq \|\boldsymbol{A}_g^T \boldsymbol{A}_g^* \boldsymbol{x}_g^*\|_2 - \|\sum_{h \neq g \in S} \boldsymbol{A}_g^T \boldsymbol{A}_h^* \boldsymbol{x}_h^*\|_2 \quad (22)$$

$$\geq B_{\min}(1-\delta) - \gamma B_{\max}(\mu_B + \delta). \quad (23)$$

Similarly, we have $\|\boldsymbol{A}_v^T \boldsymbol{y}\|_2 \leq \gamma B_{\max}(\mu_B + \delta)$, for $v \notin S$. If we choose the threshold λ to be in the range $[\gamma B_{\max}(\mu_B + \delta), B_{\min}(1-\delta) - \gamma B_{\max}(\mu_B + \delta)]$, then the support is correctly recovered. Connecting this result to the clustering narrative, cluster membership is recovered in a *single* step of our architecture. Formally, we state the following lemma.

Lemma 1 (Conditions for support recovery)**.** *Assume* $\gamma \leq \log n$ *and* $\max(\mu_B, \delta) = O(\frac{1}{\log n})$*. Then the range* $[\gamma B_{\max}(\mu_B + \delta), B_{\min}(1-\delta) - \gamma B_{\max}(\mu_B + \delta)]$ *is non-empty, and any value* λ *within that range will correctly recover the support.*

Descent property. Having shown that the true groups are immediately recovered after a single application of the proximal operator, we will now proceed in arguing descent. In order to do that, we will first compute the gradient of the loss function with respect to each group $g \in S$, then, to deal with the fact that both the support and the group codes are unknown, take the expectation of the group (which invokes the *infinite data* assumption). Having computed the gradient $\boldsymbol{G}_g$ of the loss function, we show that it is sufficiently aligned with $\boldsymbol{A}_g - \boldsymbol{A}_g^*$. This, in turn, is enough to guarantee that the error $\|\boldsymbol{A}_g^k - \boldsymbol{A}_g^*\|_F$ between the estimated group weights $\boldsymbol{A}_g^k$ at iteration k and the generating matrix $\boldsymbol{A}_g^*$ is bounded, resulting to the convergence of the weights to a *neighborhood* of $\boldsymbol{A}_g^*$.

The gradient of the loss function $\mathcal{L}(\theta)$ with respect to a group $\boldsymbol{A}_g$ for some $g \in \Gamma$ is given by[4]

$$\begin{aligned}\nabla_{\boldsymbol{A}_g} \mathcal{L}(\theta) = &-\left(\boldsymbol{y} - \boldsymbol{A}\sigma_\lambda(\boldsymbol{A}^T\boldsymbol{y})\right)\sigma_\lambda^T(\boldsymbol{A}_g^T\boldsymbol{y}) \\ &- \boldsymbol{y}\left(\boldsymbol{y} - \boldsymbol{A}\sigma_\lambda(\boldsymbol{A}^T\boldsymbol{y})\right)^T \boldsymbol{A}_g \operatorname{diag}(\sigma_\lambda'(\boldsymbol{A}_g^T\boldsymbol{y})).\end{aligned} \quad (24)$$

Considering the terms $\sigma_\lambda^T(\boldsymbol{A}_g^T\boldsymbol{y})$ and $\sigma_\lambda'(\boldsymbol{A}_g^T\boldsymbol{y})$, we can see that they will either be zero (when $g \notin S$), or they will be scaled versions of $\boldsymbol{A}_g^T\boldsymbol{y}$ and $\mathbf{1}$, respectively (where $\mathbf{1}$ denotes a vector of ones). Formally, we make the following substitutions

$$\begin{aligned}\sigma_\lambda(\boldsymbol{A}_g^T\boldsymbol{y}) &= \mathbb{1}_{\boldsymbol{x}_g \neq 0}\left(1 - \frac{\lambda}{\|\boldsymbol{A}_g^T\boldsymbol{y}\|_2}\right)\boldsymbol{A}_g^T\boldsymbol{y}, \\ \sigma_\lambda'(\boldsymbol{A}_g^T\boldsymbol{y}) &= \mathbb{1}_{\boldsymbol{x}_g \neq 0}\left(1 - \frac{\lambda}{\|\boldsymbol{A}_g^T\boldsymbol{y}\|_2}\right)\mathbf{1}.\end{aligned} \quad (25)$$

[4]The complete derivation of the gradient can be found in Appendix B.

The substitutions of (25) result in a different gradient $\nabla_{\boldsymbol{A}_g}\widetilde{\mathcal{L}}(\theta)$ that was shown to be a good approximation of $\nabla_{\boldsymbol{A}_g}\mathcal{L}(\theta)$ (Rangamani et al., 2018). We now need to deal with the last nonlinear term, $\sigma_\lambda(\boldsymbol{A}^T\boldsymbol{y})$. Since (7) attenuates the elements of each group in the support S by a different term depending on the corresponding group norm, let us define $\tau_g = \left(1 - \frac{\lambda}{\|\boldsymbol{A}_g^T\boldsymbol{y}\|_2}\right)$ for every $g \in S$. Then, we can define a vector $\boldsymbol{\tau}$ such that

$$\boldsymbol{\tau} = \big[\underbrace{\tau_1 \quad \dots \quad \tau_1}_{d} \quad \tau_2 \quad \dots \quad \underbrace{\tau_\gamma \quad \dots \quad \tau_\gamma}_{d}\big]^T. \quad (26)$$

As suggested, the vector $\boldsymbol{\tau}$ is scaling every element of the group g with the correct scaling of τ_g. We can then write $\sigma_\lambda(\boldsymbol{A}^T\boldsymbol{y}) = \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\boldsymbol{y}$, and then the approximate gradient $\nabla_{\boldsymbol{A}_g}\widetilde{\mathcal{L}}(\theta)$ becomes

$$\begin{aligned}\nabla_{\boldsymbol{A}_g}\widetilde{\mathcal{L}}(\theta) = \\ &- \mathbb{1}_{\boldsymbol{x}_g \neq 0} \cdot \tau_g \Big[\left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)\boldsymbol{y}\boldsymbol{y}^T \\ &+ \boldsymbol{y}\boldsymbol{y}^T\left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)^T\Big]\boldsymbol{A}_g.\end{aligned} \quad (27)$$

At this point, we will take the expectation of (27). The reasoning for this is that the true codes $\boldsymbol{x}_g^*$ (and their supports) are unknown, and thus we have the expected gradient $\boldsymbol{G}_g$

$$\begin{aligned}\boldsymbol{G}_g = \\ &- \mathbb{E}\Big[\mathbb{1}_{\boldsymbol{x}_g^* \neq 0} \cdot \tau_g \left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{A}_g\Big] \\ &- \mathbb{E}\Big[\mathbb{1}_{\boldsymbol{x}_g^* \neq 0} \cdot \tau_g \boldsymbol{y}\boldsymbol{y}^T\left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)^T\boldsymbol{A}_g\Big] \\ &+ \epsilon \\ &= \boldsymbol{G}_g^{(1)} + \boldsymbol{G}_g^{(2)} + \epsilon,\end{aligned} \quad (28)$$

where $\mathbb{1}_{\boldsymbol{x}_g^*}$ replaced $\mathbb{1}_{\boldsymbol{x}_g}$, as we showed that the support is recovered with a single application of the proximal operator. Regardless, this introduces an error term ϵ, that is, however, bounded (and specifically has a norm of order $O(n^{-w(1)})$ (Nguyen et al., 2019)). In order to deal with the unknown support, we will use Adam's Law and compute $\boldsymbol{G}_g^{(i)}$ as $\mathbb{E}\left[\boldsymbol{G}_{g|S}^{(i)}\right]$, where

$$\boldsymbol{G}_{g|S}^{(i)} = -\mathbb{E}\left[\mathbb{1}_{\boldsymbol{x}_g^* \neq 0} \cdot \tau_g \left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{A}_g \mid S\right]. \quad (29)$$

Then, noting that $\boldsymbol{y}\boldsymbol{y}^T = \sum_{h,v \in S} \boldsymbol{A}_h^*\boldsymbol{x}_h^*\boldsymbol{x}_v^{*T}\boldsymbol{A}_v^{*T}$ and invoking the assumption of the code covariances we can finally write $\boldsymbol{G}_{g|S}^{(i)}$ as

$$\begin{aligned}\boldsymbol{G}_{g|\Gamma}^{(1)} &= -\tau_g\left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)\boldsymbol{A}_g^*\boldsymbol{A}_g^{*T}\boldsymbol{A}_g + \boldsymbol{P}_1, \\ \boldsymbol{G}_{g|S}^{(2)} &= -\tau_g\boldsymbol{A}_g^*\boldsymbol{A}_g^{*T}\left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau})\boldsymbol{A}_S^T\right)\boldsymbol{A}_g + \boldsymbol{P}_2,\end{aligned} \quad (30)$$

where the $\boldsymbol{P}_i$ are cross-terms given by

$$\boldsymbol{P}_1 = -\tau_g \left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau}) \boldsymbol{A}_S^T\right) \sum_{h \neq g \in S} \boldsymbol{A}_h^* \boldsymbol{A}_h^{*T} \boldsymbol{A}_g,$$
$$\boldsymbol{P}_2 = -\tau_g \sum_{h \neq g \in S} \boldsymbol{A}_h^* \boldsymbol{A}_h^{*T} \left(\boldsymbol{I} - \boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau}) \boldsymbol{A}_S^T\right) \boldsymbol{A}_g. \tag{31}$$

Then, noting that $\boldsymbol{A}_S \operatorname{diag}(\boldsymbol{\tau}) \boldsymbol{A}_S^T = \sum_{h \in S} \tau_h \boldsymbol{A}_h \boldsymbol{A}_h^T$ and skipping intermediate steps we can finally write the expected gradient $\boldsymbol{G}_g$ as

$$\boldsymbol{G}_g = \tau_g p_g (2 - \tau_g)(\boldsymbol{A}_g - \boldsymbol{A}_g^*) \boldsymbol{A}_g^{*T} \boldsymbol{A}_g + \boldsymbol{V} + \epsilon, \tag{32}$$

where the matrix $\boldsymbol{V}$ is an amalgam of $\mathbb{E}[\boldsymbol{P}_1 + \boldsymbol{P}_2]$ and lower probability terms. We can then prove the following theorem.

Theorem 1 (Gradient direction)**.** *Consider the expected gradient of* (32)*. The* inner product *between the i-th columns of $\boldsymbol{G}_g$ and $\boldsymbol{A}_g - \boldsymbol{A}_g^*$ is lower-bounded by*

$$2\langle \boldsymbol{g}_i, \boldsymbol{a}_i - \boldsymbol{a}_i^* \rangle \geq \tau_g (2 - \tau_g) p_g \alpha_i \|\boldsymbol{a}_i - \boldsymbol{a}_i^*\|_2^2 + \frac{1}{\tau_g (2 - \tau_g) p_g \alpha_i} \|\boldsymbol{g}_i\|_2^2 - \frac{1}{\tau_g (2 - \tau_g) p_g \alpha_i} \|\boldsymbol{v}_i\|_2^2 - O((\mu_B + \delta)^2 \tfrac{\gamma^5}{\Gamma^3}), \tag{33}$$

where $\alpha_i = \boldsymbol{a}_i^{*T} \boldsymbol{a}_i$ and $\|\boldsymbol{v}_i\|_2$ satisfies

$$\|\boldsymbol{v}_i\|_2 \leq \tau_g (2 - \tau_g) p_g (\alpha_{\max} \sqrt{d^2 + 1} + \delta) \|\boldsymbol{A}_g - \boldsymbol{A}_g^*\|_F \tag{34}$$

with $\alpha_{\max} = \max_i |\boldsymbol{a}_i^{*T} \boldsymbol{a}_i|$ (and we similarly denote $\alpha_{\min} = \min_i |\boldsymbol{a}_i^{*T} \boldsymbol{a}_i|$).

Intuitively, Theorem 1 states that the gradient $\boldsymbol{g}_i$ "points" at the same direction as $\boldsymbol{a}_i - \boldsymbol{a}_i^*$, and thus moving along the opposite direction will get us closer to the generative model of $\boldsymbol{A}_g$. We are finally able to state our main result.[5]

Theorem 2 (Convergence to a neighborhood)**.** *Suppose that the learning rate η is upper bounded by $\frac{1}{\tau_g (2 - \tau_g) p_g \alpha_{\max}}$, the norm $\|\boldsymbol{A}_g^T \boldsymbol{A}_g^*\|_F^2$ is upper bounded by $\frac{1}{\tau_g^2}[3(2 - \tau_g)(\tau_g - \frac{2}{3})n + (\tau_g - 1)^2 \delta^2]$, and that $\boldsymbol{G}_g$ is "aligned" with $\boldsymbol{A}_g$, i.e.*

$$2\langle \boldsymbol{g}_i, \boldsymbol{a}_i - \boldsymbol{a}_i^* \rangle \geq \kappa \|\boldsymbol{a}_i - \boldsymbol{a}_i^*\|_2^2 + \nu \|\boldsymbol{g}_i\|_2^2 - \xi \|\boldsymbol{v}_i\|_2^2 - \varepsilon,$$

where κ, ν, ξ, and ε are given by Theorem 1. Then it follows that

$$\|\boldsymbol{A}_g^{k+1} - \boldsymbol{A}_g^*\|_F^2 \leq (1 - \rho) \|\boldsymbol{A}_g^k - \boldsymbol{A}_g^*\|_F^2 + \eta \gamma \varepsilon,$$

where $\rho = \eta \tau_g (2 - \tau_g) p_g (\alpha_{\min} - \frac{d (\alpha_{\max} \sqrt{d^2 + 1} + \delta)^2}{\alpha_{\min}})$. If we further consider the assumptions of Lemma 1 then

$$\|\boldsymbol{A}_g^{k+1} - \boldsymbol{A}_g^*\|_F^2 \leq (1 - \rho) \|\boldsymbol{A}_g^k - \boldsymbol{A}_g^*\|_F^2 + O(\tfrac{\log^3 n}{\Gamma^2}).$$

[5] The proof of both Theorems 1 and 2 are given in Appendix C.

This concludes the proof of descent; we showed that, when training with gradient descent, the weights of a shallow group-sparse autoencoder will, in Frobenius norm, get closer to the weights of the generating model. However, because of the error term ε, the convergence isn't to the exact weights; it is rather to a *neighborhood* of the generative model.

Comparison with prior work. Existing literature has analyzed the convergence of gradient descent assuming a sparse generative model (Arora et al., 2015; Rangamani et al., 2018; Nguyen et al., 2019). As our study assumes a group-sparse model, it is natural to ask how these assumptions and analyses differ. We identify two main differences in favor of the group-sparse approach:
(a) Works using traditional notions of sparsity (Arora et al., 2015; Rangamani et al., 2018; Nguyen et al., 2019) require bounds on the true codes of the generative model. Instead, our formulation requires a bound on the *norms* of the true codes; a significantly looser assumption. As a result, a group-sparse autoencoder is able to recover the support of the true codes $\boldsymbol{x}^*$ with a significantly higher success rate (Section 4).
(b) Sparse approaches (Nguyen et al., 2019) require the magnitude of the bias to *diminish* at every iteration. In stark contrast, our analysis makes no such assumption; we are able to satisfy both support recovery and convergence using a *constant* choice of bias.

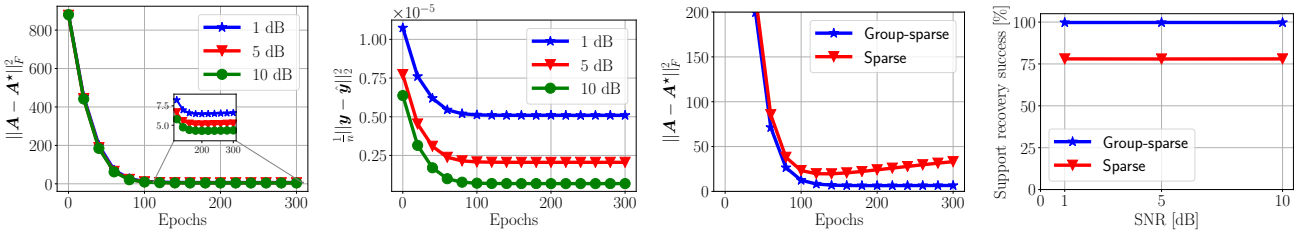
4. Experimental validation

We support our theoretical results via numerical simulations (Section 4.1) and evaluate the performance of group-sparse learning on a clustering task (Section 4.2). We highlight the superior performance of group-sparse coding compared to traditional sparse coding, and the natural clustering induced by the group-sparse autoencoder in real-data experiments.

4.1. Simulation experiments

In this section we show that training our proposed architecture recovers the parameters of the generative model and showcase superior performance to a traditional sparse autoencoder, both in recovering the generating dictionary, and in the ability of the encoder to identify nonzero blocks of units and their activations.

Data generation. We generated a set of data points $\{\boldsymbol{y}_i\}_{i=1}^N \in \mathcal{R}^{950}$ consisting of $N = 10{,}000$ examples following the noisy group-sparse generative model $\boldsymbol{y}_i = \boldsymbol{A}^* \boldsymbol{x}_i^* + \boldsymbol{z}_i$, where the data are corrupted by strong additive zero-mean Gaussian noise with signal-to-noise-ratio (SNR) equal to 1, 5, and 10 dB. We let $\boldsymbol{A}^* \in \mathcal{R}^{n \times m}$ to comprise $\Gamma = 500$ groups, each of size $d = 2$ (i.e. $m = 1000$). We sampled the entries of each column $\boldsymbol{a}_i^*$ from a zero-mean



(a) Recovery of the generating matrix (in terms of the Frobenius norm) for various noise levels.

(b) Training loss of the group-sparse autoencoder across epochs for different noise levels.

(c) Comparing a group-sparse and a sparse network in the recovery of the generating matrix.

(d) Success rate in recovering the correct support for the group-sparse and sparse architectures.

Figure 3: Numerical simulations highlighting the superior performance of the proposed model versus a sparse autoencoder.

Gaussian distribution and normalized each column. The codes $\boldsymbol{x}_i^* \in \mathbb{R}^m$ contains 3 active groups chosen uniformly at random from the set $[\Gamma]$ (i.e. the code is 6-sparse). We sampled the code entries of active groups according to the $\mathcal{N}(0,1)$ distribution; after normalizing the vector of coefficients $\boldsymbol{x}_g$ in an active group, we scaled them with a factor $c \sim \text{Uniform}(4,5)$.

Training. We initialized the weights of a group-sparse autoencoder with an extreme perturbation of the generating dictionary (i.e. $\boldsymbol{A} = \boldsymbol{A^*} + \boldsymbol{B}$ with $\boldsymbol{B} \sim \mathcal{N}(\boldsymbol{0}, \sigma_B^2 \boldsymbol{I})$), where the average correlation between the columns of $\boldsymbol{A}^*$ and $\boldsymbol{A}$ is approximately 0.15. We chose this initialization procedure over a random initialization to make sure the weights are far from $\boldsymbol{A^*}$, but also to minimize the possibility of column permutations (so that we can compare $\boldsymbol{A}$ to $\boldsymbol{A}^*$ without solving the combinatorial problem of matching columns) (Agarwal et al., 2016; Tolooshams et al., 2020a). We trained the architecture for 300 epochs with full-batch gradient descent using the Adam optimizer with a learning rate $\eta = 10^{-3}$ and bias $\lambda = 2$. During training, we did not constrain the weights to have unit norm. We measured the distance between the network weights and the generating ones at each iteration using the Frobenius norm $\|\boldsymbol{A} - \boldsymbol{A}^*\|_F^2$ of the difference between the normalized weights.

Results. Figure 3a supports our theory by demonstrating the convergence of $\boldsymbol{A} \to \boldsymbol{A}^*$ and how the error $\|\boldsymbol{A} - \boldsymbol{A}^*\|_F^2$ decreases as a function of epochs for various SNRs. This figure extends our theory and highlights the robustness of the network to extreme noise. Figure 3b shows the reconstruction loss as a function of epochs for various SNRs.

We compared the group-sparse autoencoder with a sparse autoencoder for all SNRs. For a SNR of 1 dB, Figure 3c shows that by taking into account the structured sparsity of the data, we estimate the generating dictionary better; the group-sparse network convergences to a closer neighbourhood of the dictionary than the sparse one. We attribute this superiority to the better sparse coding performance (i.e. better recovery of the support at the encoder) of the group-sparse autoencoder compared to that of the sparse network. Indeed, based on our theory, to ensure the correct recovery of the support, the group-sparse analysis only requires a bound on the *norm* of each group. In contrast, the sparse autoencoder of Nguyen et al. (2019), enforces conditions on the energy of individual entries of the code. Figure 3d examines the codes at the output of the encoder; we observe that unlike the sparse network, the group-sparse autoencoder has successfully recovered the correct support.

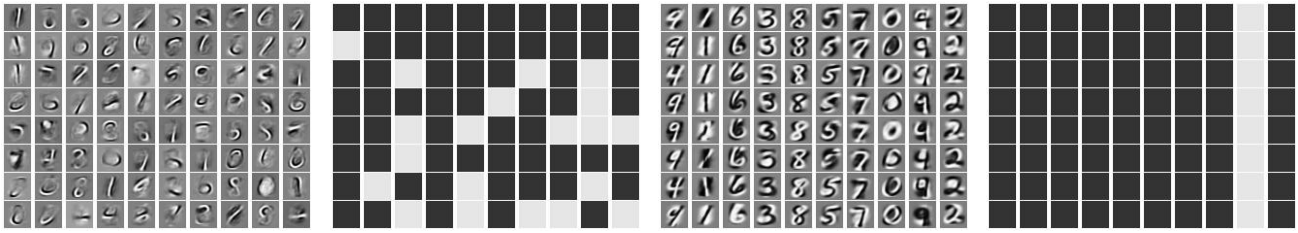
4.2. Real-data experiments

In this section, we demonstrate the clustering capability of our proposed group-sparse autoencoder on the MNIST Harwritten Digit dataset.

Training. We train our architecture on 60,000 grayscale images of size 28×28 from the MNIST dataset. We set the number of groups to $\Gamma = 10$ each of size $s = 16$ and we unfold our encoder for 15 iterations. We set the group-sparsity-inducing bias to $\lambda = 0.2$ and train the network for 300 epochs with the Adam optimizer using a learning rate $\eta = 10^{-3}$. Using the same approach, settings, and a sparsity-inducing bias of $\lambda = 0.03$, we train a sparse autoencoder (Tolooshams et al., 2018; Nguyen et al., 2019) with $\boldsymbol{A} \in \mathcal{R}^{784 \times 160}$ as baseline.

Results. We observe that the weights learned by the group-sparse network (Figure 4c) are more interpretable than those learned by a sparse network (Figure 4a). The sparse dictionaries of Figure 4a resemble the individual pen strokes of handwritten digits and do not reflect the underlying class membership. On the other hand, the group-sparse model is able to learn higher level features, shown in Figure 4c, that naturally resemble members of the cluster and reflect the ten distinct digit classes of the dataset.

Figures 4b and 4d shows examples of the sparse supports recovered using a sparse and a group-sparse architecture, respectively. We clearly observe that while the sparse support is spread throughout the coding vector, in stark contrast the group-sparse one is heavily structured. This meaningful structured connectivity (i.e., grouping of the dictionary



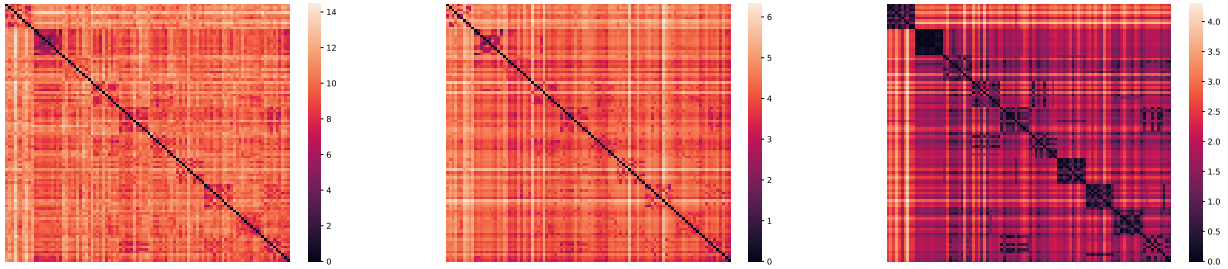
(a) Learned representations using a sparse autoencoder.

(b) Activation pattern of a code for the sparse architecture.

(c) Learned representations using a group-sparse autoencoder.

(d) Activation pattern of a code for the group-sparse architecture.

Figure 4: Representations and code activation patterns learned by the group-sparse and sparse autoencoders.



(a) Similarity matrix of MNIST images when representing the data using their raw format (pixel basis).

(b) Similarity matrix of MNIST images on the latent representation learned by a sparse autoencoder.

(c) Similarity matrix of MNIST images on the latent representation learned by a group-sparse autoencoder

Figure 5: Pairwise distances between the representations of MNIST test images for different latent representations.

	k-means	Spectral clustering
Raw representation	54.46	63.04
Sparse codes	54.14	55.49
Sparse codes$_+$	60.94	56.98
Group-sparse codes	**74.5**	**79.4**
Group-sparse codes$_+$	**80.09**	**80.27**

Table 1: Test accuracy on the MNIST dataset ($N = 10{,}000$) for all 10 classes. The subscript $(\cdot)_+$ indicates that the coefficients were made nonnegative and scaled to sum to 1.

atoms) is especially noteworthy considering that the network was trained *without* the use of the ground truth class label information.

Figure 5 shows pairwise distances between 100 MNIST test images of each of the 10 digit classes with respect to the pixel basis, the sparse dictionary of Figure 4a, and the group-sparse dictionary of Figure 4c. We report that the similarity structure of the group-sparse dictionary lends itself most readily to standard similarity-based clustering algorithms. The reported performance gains attained by using group-sparsity in application domains such as clustering highlight the importance more structured notions of sparsity.

Finally, in Table 1 we evaluate the efficacy of the dictionaries in Figures 4a and 4c when used as a pre-processing step for clustering. In particular, we performed clustering using k-means with 10 centroids and spectral clustering on the similarity graph of 1,000 nearest neighbors. The input to these algorithms is the representation of the digits in the pixel basis (first row), the latent representations learned with a sparse autoencoder (second and third rows), or the group-sparse autoencoder (fourth and fifth rows). We observed that performing clustering on the group-structured representations dramatically improved performance over baselines, and over a sparse approach. This further solidifies our findings that group-sparse architectures perform a natural grouping of the unlabeled data.

5. Conclusions

In this work we studied the gradient dynamics of a group-sparse, shallow autoencoder. Motivated by the connection between group-sparsity and cluster membership, we first introduced the group-sparse generative family and highlighted its expressivity. We then proceeded into proving, under mild conditions, that training the architecture with gradient descent will result into the convergence of the weight matrix to a neighborhood of the true weights. Finally, we provided numerical and real data experiments to support the developed theory.

References

Agarwal, A., Anandkumar, A., Jain, P., and Netrapalli, P. Learning sparsely used overcomplete dictionaries via alternating minimization. *SIAM Journal on Optimization*, 26(4):2775–2799, 2016.

Aharon, M., Elad, M., and Bruckstein, A. k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

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

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2011.

Bora, A., Jalal, A., Price, E., and Dimakis, A. Compressed sensing using generative models. In *International Conference on Machine Learning*, 2017.

Chen, X., Liu, J., Wang, Z., and Yin, W. Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds. In *Advances in Neural Information Processing Systems*, 2018.

del Aguila Pla, P. and Jaldén, J. Cell detection by functional inverse diffusion and non-negative group sparsity–part ii: Proximal optimization and performance evaluation. *IEEE Transactions on Signal Processing*, 20(20):5422–5437, 2018.

Eldar, Y., Kuppinger, P., and Bölcskei, H. Block-sparse signals: Uncertainty relations and efficient recovery. *IEEE Transactions on Signal Processing*, 58(6):3042–3054, 2010.

Gregor, K. and LeCun, Y. Learning fast approximations of sparse coding. In *International Conference on Machine Learning*, 2010.

Gribonval, R. and Nielsen, M. Sparse representations in unions of bases. *IEEE Transactions on Information Theory*, 49(12):3320–3325, 2003.

Hershey, J., Le Roux, J., and Weninger, F. Deep unfolding: Model-based inspiration of novel deep architectures. In *arXiv*, 2014.

Huang, S. and Tran, T. Sparse signal recovery via generalized entropy functions minimization. *IEEE Transactions on Signal Processing*, 67(5):1322–1337, 2018.

Lecouat, B., Ponce, J., and Mairal, J. Fully trainable and interpretable non-local sparse models for image restoration. In *European Conference on Computer Vision*, 2020.

Liu, B., Wang, M., Foroosh, H., Tappen, M., and Penksy, M. Sparse convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition*, 2015.

Liu, J., Chen, X., Wang, Z., and Yin, W. ALISTA: Analytic weights are as good as learned weights in LISTA. In *International Conference on Learning Representations*, 2019.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online dictionary learning for sparse coding. In *International Conference on Machine Learning*, 2009.

Mairal, J., Bach, F., and Ponce, J. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012.

Mehta, N. and Gray, A. Sparsity-based generalization bounds for predictive sparse coding. In *International Conference on Machine Learning*, 2013.

Monga, V., Li, Y., and Eldar, Y. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. In *arXiv*, 2020.

Narang, S., Undersander, E., and Diamos, G. Block-sparse recurrent neural networks. In *arXiv*, 2017.

Nguyen, T., Wong, R., and Hegde, C. On the dynamics of gradient descent for autoencoders. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Parikh, N. and Boyd, S. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014.

Rangamani, A., Mukherjee, A., Basu, A., Arora, A., and Ganapathi, T. Sparse coding and autoencoders. In *IEEE International Symposium on Information Theory*, 2018.

Scardapane, S., Comminiello, D., Hussain, A., and Uncini, A. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

Shlezinger, N., Whang, J., Eldar, Y., and Dimakis, A. Model-based deep learning. In *arXiv*, 2020.

Simon, D. and Elad, M. Rethinking the CSC model for natural images. In *Advances in Neural Information Processing Systems*, 2019.

Sze, V., Chen, Y.-H., Yang, T.-J., and Emer, J. *Efficient processing of deep neural networks*. Morgan & Claypool Publishers, 2020.

Tasissa, A., Theodosis, E., Tolooshams, B., and Ba, D. Towards improving discriminative reconstruction via simultaneous dense and sparse coding. In *arXiv*, 2020.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1): 267–288, 1996.

Tolooshams, B., Dey, S., and Ba, D. Scalable convolutional dictionary learning with constrained recurrent sparse auto-encoders. In *International Workshop on Machine Learning for Signal Processing*, 2018.

Tolooshams, B., Dey, S., and Ba, D. Deep residual autoencoders for expectation maximization-inspired dictionary learning. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2020a.

Tolooshams, B., Song, A., Temereanca, S., and Ba, D. Convolutional dictionary learning based auto-encoders for natural exponential-family distributions. In *International Conference on Machine Learning*, 2020b.

Tseng, P. Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. *SIAM Journal of Control and Optimization*, 29(1):119–138, 1991.

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, 2016.

Yoon, J. and Hwang, S. J. Combined group and exclusive sparsity for deep neural networks. In *International Conference on Machine Learning*, 2017.

Yuan, M. and Lin, Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society*, 68(1):49–67, 2006.