Sparse Coding and Autoencoders

Akshay Rangamani[*]
ECE Department
Johns Hopkins University
rangamani.akshay@jhu.edu

Anirbit Mukherjee[*]
AMS Department
Johns Hopkins University
amukhe14@jhu.edu

Amitabh Basu
AMS Department
Johns Hopkins University
basu.amitabh@jhu.edu

Tejaswini Ganapathy
Senior Data Scientist, Infrastructure Engineering,
Salesforce, San Francisco Bay Area
tganapathi@salesforce.com

Ashish Arora
ECE Department
Johns Hopkins University
aarora8@jhu.edu

Sang (Peter) Chin
Department of Computer Science
Boston University
spchin@cs.bu.edu

Trac D. Tran
ECE Department
Johns Hopkins University
trac@jhu.edu

Abstract

In this work we study the landscape of squared loss of an *Autoencoder* when the data generative model is that of *"Sparse Coding"/"Dictionary Learning"*. The neural net considered is mapping $\mathbb{R}^n \to \mathbb{R}^n$ and has a single ReLU activation layer of size $h > n$. The net has access to vectors $y \in \mathbb{R}^n$ obtained as $y = A^*x^*$ where $x^* \in \mathbb{R}^h$ are sparse high dimensional vectors and $A^* \in \mathbb{R}^{n \times h}$ is an overcomplete incoherent matrix. Under very mild distributional assumptions on x^*, we prove that the norm of the expected gradient of the squared loss function is asymptotically (in sparse code dimension) negligible for *all* points in a small neighborhood of A^*. This is supported with experimental evidence using synthetic data. We conduct experiments to suggest that A^* sits at the bottom of a well in the landscape and we also give experiments showing that gradient descent on this loss function gets columnwise very close to the original dictionary even with far enough initialization. Along the way we prove that a layer of ReLU gates can be set up to automatically recover the support of the sparse codes. Since this property holds independent of the loss function we believe that it could be of independent interest.

A full version of this paper is accessible at: `https://arxiv.org/abs/1708.03735`

1 Introduction

In *Dictionary Learning/Sparse Coding* one receives samples of vectors $y \in \mathbb{R}^n$ that have been generated as $y = A^*x^*$ where $A^* \in \mathbb{R}^{n \times h}$ and $x^* \in \mathbb{R}^h$ and $h > n$. We typically assume that the number of non-zero entries in x^* to be no larger than some sub-linear function of the dimension h and that A^* satisfies certain incoherence properties. The question now is to recover A^* from samples of y. There have been renewed investigations into the hardness of this problem [23] and

[*]Equal Contribution

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

many former results have recently been reviewed in these lectures [10]. Ever since the groundbreaking paper by Olshausen and Field([16]) (a recent review by the same authors can be found in [17]), over the years many algorithms have been developed to solve sparse coding in various cases [9, 2, 19, 7, 1, 3, 4, 5, 20, 21, 6]. A detailed comparison among these various approaches can be found in [7].
Olshausen and Field had, as early as in 1996, already made the connection between sparse coding and training neural architectures and in today's terminology this is very naturally reminiscent of the architecture of an autoencoder [15]. Autoencoder architectures have very recently been reviewed here, [8]. Provable training of neural nets has been a long standing open question and many recent works have focussed on such proofs for nets with one hidden layer [11, 12, 22, 24]. However, to the best of our knowledge its not clear if the techniques in these papers can be adapted to establish the behaviour of a loss function (or gradient descent on it) in the vicinity of a pre-specified point like the unknown dictionary. In this work we report on our progress towards bridging the above mentioned mathematical gap between autoencoders and sparse coding.

2 Introducing the neural architecture and the distributional assumptions

For any $n, h \in \{1, 2, ..\}$, we consider autoencoders which are fully connected $\mathbb{R}^n \to \mathbb{R}^n$ neural networks with a single hidden layer of h activations. We focus on networks that use the Rectified Linear Unit (ReLU) activation which is the function $\text{ReLU} : \mathbb{R}^h \to \mathbb{R}^h$ mapping $\vec{x} \to (\max\{0, x_i\})_{i=1}^h$. In this case, the autoencoder can be seen as computing the following function $\hat{y}(W, y, \epsilon)$ as follows,

$$\begin{aligned} r &= \text{ReLU}\left(Wy - \epsilon\right) \\ \hat{y} &= W^\top r \end{aligned} \tag{1}$$

Here $y \in \mathbb{R}^n$ is the input to the autoencoder, $W \in \mathbb{R}^{h \times n}$ is the linear transformation implemented by the first layer, $r \in \mathbb{R}^h$ is the output of the layer of activations, $\epsilon \in \mathbb{R}^h$ is the bias vector and $\hat{y} \in \mathbb{R}^n$ is the output of the autoencoder. Note that we impose the condition that the second layer of weights is simply the transpose of the first layer. We shall define the columns of $W^\top$ (rows of W) as $\{W_i\}_{i=1}^h$.

Assumptions on the dictionary and the sparse code. We assume that our signal y is generated using sparse linear combinations of atoms/vectors of an overcomplete dictionary, i.e., $y = A^* x^*$, where $A^* \in \mathbb{R}^{n \times h}$ is a dictionary, and x^* is a compactly supported non-negative sparse code parameterized as $x^* \in [a(h), b(h)]^h$ with $a(h) > 0$. x^* is assumed to have at most $k = h^p$ (for some $0 < p < 1$) non-zero elements. The columns of the original dictionary A^* (labeled as $\{A_i^*\}_{i=1}^h$) are assumed to be normalized and also satisfying the incoherence property that $\max_{\substack{i,j=1,..,h \\ i \neq j}} |\langle A_i^*, A_j^* \rangle| \leq \frac{\mu}{\sqrt{n}} = h^{-\xi}$ for some $\xi > 0$.
We assume that the sparse code x^* is sampled from a distribution with the following properties. We fix a set of possible supports of x^*, denoted by $\mathbb{S} \subseteq 2^{[h]}$, where each element of $\mathbb{S}$ has at most $k = h^p$ elements. We consider any arbitrary discrete probability distribution $D_{\mathbb{S}}$ on $\mathbb{S}$ such that the probability $q_1 := \mathbb{P}_{S \sim \mathbb{S}}[i \in S]$ is independent of $i \in [h]$, and the probability $q_2 := \mathbb{P}_{S \in \mathbb{S}}[i, j \in S]$ is independent of $i, j \in [h]$. A special case is when $\mathbb{S}$ is the set of all subsets of size k, and $D_{\mathbb{S}}$ is the uniform distribution on $\mathbb{S}$. For every $S \in \mathbb{S}$ there is a distribution say D_S on $(\mathbb{R}^{\geq 0})^h$ which is supported on vectors whose support is contained in S and which is uncorrelated for pairs of coordinates $i, j \in S$. Further, we assume that the distributions D_S are such that each coordinate i is compactly supported over an interval $[a(h), b(h)]$, where $a(h)$ and $b(h)$ are independent of both i and S but will be functions of h. Moreover, $m_1(h) := \mathbb{E}_{x^* \sim D_S}[x_i^*]$, and $m_2(h) := \mathbb{E}_{x^* \sim D_S}[x_i^{*2}]$ are assumed to be independent of both i and S but allowed to depend on h. For ease of notation henceforth we will keep the h dependence of these variables implicit and refer to them as a, b, m_1 and m_2. All of our results will hold in the special case when a, b, m_1, m_2 are constants (no dependence on h).

The coordinates of x^* are assumed to be i.i.d random variables and we denote the first and the second moments of their distribution as m_1 and m_2 respectively. Some further details about the distribution of x^* can be found in the full version of the paper, [18]. In particular all of our results will hold in the special case when a, b, m_1 and m_2 are all constants (no dependence on h).

3 Main Results

3.1 Recovery of the support of the sparse code by a layer of ReLUs

First we prove the following theorem which precisely quantifies the sense in which a layer of ReLU gates is able to recover the support of the sparse code when the weight matrix of the deep net is close to the original dictionary.

Theorem 3.1 *(**Recovering the Sparse Code Support at the Hidden Layer**) Let each column of $W^\top$ be within a δ-ball of the corresponding column of A^*, where $\delta = O\left(h^{-p-\nu^2}\right)$ for some $\nu > 0$, such that $p + \nu^2 < \xi$. We further assume that $a = \Omega\left(bh^{-\nu^2}\right)$. Let the bias of the hidden layer of the autoencoder, as defined in* (1) *be $\epsilon = 2m_1 k\left(\delta + \frac{\mu}{\sqrt{n}}\right)$. Let r be the vector defined in* (1)*. Then $r_i \neq 0$ if $i \in$ supp(x^*), and $r_i = 0$ if $i \notin$ supp(x^*) with probability at least $1 - \exp\left(-\frac{2h^p m_1^2}{(b-a)^2}\right)$ (with respect to the distribution on x^*).*

As long as $\frac{h^p m_1^2}{(b-a)^2}$ is large, i.e., an increasing function of h, we can interpret this as saying that the probability of the adverse event is small, and we have successfully achieved support recovery at the hidden layer in the limit of large sparse code dimension.

3.2 Asymptotic Criticality of the Autoencoder around A^*

In this work we analyze the following standard squared loss function for the autoencoder,

$$L = \frac{1}{2}||\hat{y} - y||^2 \tag{2}$$

In the above we continue to use the variables as defined in equation 1. If we consider a generative model in which A^* is a square, orthogonal matrix and x^* is a non-negative vector (not necessarily sparse), it is easily seen that the standard squared reconstruction error loss function for the autoencorder has a global minimum at $W = A^{*\top}$. However in our generative model A^* is an incoherent and overcomplete dictionary.

Theorem 3.2 *(**The Main Theorem**) Assume that the hypotheses of Theorem 3.1 hold, and $p < \min\{\frac{1}{2}, \nu^2\}$ (and hence $\xi > 2p$). Further, assume that the distribution parameters are such that exp $\left(\frac{h^p m_1^2}{2(b-a)^2}\right)$ is superpolynomial in h (which holds, for example, when m_1, a, b are $O(1)$). Then for $i = 1, \ldots, h$,*

$$\left\| \mathbb{E}\left[\frac{\partial L}{\partial W_i}\right] \right\|_2 \leq o\left(\frac{\max\{m_1^2, m_2\}}{h^{1-p}}\right).$$

Thus we have shown that this δ ball around A^* is asymptotically critical for the squared loss function for this autoencoder.

4 A Layer of ReLU Gates can Recover the Support of the Sparse Code (Proof of Theorem 3.1)

Most sparse coding algorithms are based on an alternating minimization approach, where one iteratively finds a sparse code based on the current estimate of the dictionary, and then uses the estimated sparse code to update the dictionary. The analogue of the sparse coding step in an autoencoder, is the passing through the hidden layer of activations of a certain affine transformation (W which behaves as the current estimate of the dictionary) of the input vectors. We show that under certain stochastic assumptions, the hidden layer of ReLU gates in an autoencoder recovers with high probability the support of the sparse vector which corresponds to the present input.

Proof 4.1 (Proof of Theorem 3.1) *From the model assumptions, we know that the dictionary A^* is incoherent, and has unit norm columns. So, $|\langle A_i^*, A_j^* \rangle| \leq \frac{\mu}{\sqrt{n}}$ for all $i \neq j$, and $||A_i^*|| = 1$ for all i. This means that for $i \neq j$,*

$$\begin{aligned} |\langle W_i, A_j^* \rangle| &= |\langle W_i - A_i^*, A_j^* \rangle| + |\langle A_i^*, A_j^* \rangle| \\ &\leq ||W_i - A_i^*||_2 ||A_j^*||_2 + \frac{\mu}{\sqrt{n}} \leq (\delta + \frac{\mu}{\sqrt{n}}) \end{aligned} \tag{3}$$

Otherwise for $i = j$,

$$\langle W_i, A_i^* \rangle = \langle W_i - A_i^*, A_i^* \rangle + \langle A_i^*, A_i^* \rangle = \langle W_i - A_i^*, A_i^* \rangle + 1,$$

and thus,

$$1 - \delta \leq \langle W_i, A_i^* \rangle \leq 1 + \delta, \tag{4}$$

where we use the fact that $|\langle W_i - A_i^*, A_i^* \rangle| \leq \delta$.

Let $y = A^* x^*$ and let S be the support of x^*. Then we define the input to the ReLU activation $Q - \epsilon = W\mathbf{y} - \epsilon$ as

$$\begin{aligned} Q_i &= \sum_{j \in S} \langle W_i, A_j^* \rangle x_j^* = \langle W_i, A_i^* \rangle x_i^* \mathbf{1}_{i \in S} + \sum_{j \in S \setminus i} \langle W_i, A_j^* \rangle x_j^* \\ &= \langle W_i, A_i^* \rangle x_i^* \mathbf{1}_{i \in S} + Z_i \end{aligned}$$

First we try to get bounds on Q_i when $i \in supp(x^*)$. From our assumptions on the distribution of x_i^* we have, $0 \leq a \leq x_i^* \leq b$ and $\mathbb{E}[x_i^*] = m_1$ for all i in the support of x^*. For $i \in supp(x^*)$,

$$\begin{aligned} & Q_i = \langle W_i, A_i^* \rangle x_i^* + Z_i \\ \implies & Q_i \geq (1 - \delta) a + Z_i \end{aligned}$$

where we use (4)*. Using* (3)*, Z_i has the following bounds:*

$$-bk \left(\delta + \frac{\mu}{\sqrt{n}} \right) \leq Z_i \leq bk \left(\delta + \frac{\mu}{\sqrt{n}} \right)$$

Plugging in the lower bound for Z_i and the proposed value for the bias, we get

$$Q_i - \epsilon \geq (1 - \delta) a - bk \left(\delta + \frac{\mu}{\sqrt{n}} \right) - 2 m_1 k \left(\delta + \frac{\mu}{\sqrt{n}} \right)$$

For $Q_i - \epsilon \geq 0$, we need:

$$a \geq \frac{(b + 2m_1) \left(\delta + \frac{\mu}{\sqrt{n}} \right) k}{1 - \delta}$$

Now plugging in the values for the various quantities, $\frac{\mu}{\sqrt{n}} = h^{-\xi}$ and $k = h^p$ and $\delta = O\left(h^{-p-\nu^2} \right)$, if we have $a = \Omega\left(b h^{-\nu^2} \right)$, then $Q_i - \epsilon \geq 0$.

Now, for $i \notin supp(x^*)$ we would like to analyze the following probability:

$$Pr[Q_i - \epsilon \geq 0 | i \notin supp(x^*)]$$

We first simplify the quantity $Pr[Q_i - \epsilon \geq 0 | i \notin supp(x^*)]$ as follows

$$\begin{aligned} Pr[Q_i \geq \epsilon | i \notin supp(x^*)] &= Pr[Z_i \geq \epsilon] \\ &= Pr\left[\sum_{j \in S \setminus i} \langle W_i, A_j^* \rangle x_j^* \geq \epsilon\right] \end{aligned}$$

Using the Chernoff's bound, we can obtain

$$\begin{aligned} Pr[Z_i \geq \epsilon] &\leq \inf_{t \geq 0} e^{-t\epsilon} \mathbb{E}\left[\prod_{j \in S \setminus i} \left[e^{t\langle W_i, A_j^* \rangle x_j^*}\right]\right] \\ &= \inf_{t \geq 0} e^{-t\epsilon} \prod_{j \in S \setminus i} \mathbb{E}\left[e^{t\langle W_i, A_j^* \rangle x_j^*}\right] \\ &\leq \inf_{t \geq 0} e^{-t\epsilon} \mathbb{E}^k \left[e^{t\left(\delta + \frac{\mu}{\sqrt{n}}\right) x_j^*}\right] \\ &\leq \inf_{t \geq 0} e^{-t\epsilon} \left(e^{t\left(\delta + \frac{\mu}{\sqrt{n}}\right) m_1} e^{\frac{t^2 \left(\delta + \frac{\mu}{\sqrt{n}}\right)^2 (b-a)^2}{8}}\right)^k \end{aligned}$$

where the second inequality follows from (3) *and the fact that t and x_i^* are both nonnegative, and the third inequality follows from Hoeffding's Lemma. Next, we also have*

$$\begin{aligned} Pr[Z_i \geq \epsilon] &\leq \inf_{t \geq 0} e^{-t\left(\epsilon - k\left(\delta + \frac{\mu}{\sqrt{n}}\right) m_1\right) + t^2 \frac{k}{8} \left(\delta + \frac{\mu}{\sqrt{n}}\right)^2 (b-a)^2} \\ &= e^{-\frac{(\epsilon - k(\delta + \frac{\mu}{\sqrt{n}}) m_1)^2}{\frac{k}{2}(\delta + \frac{\mu}{\sqrt{n}})^2 (b-a)^2}}. \end{aligned}$$

Finally, since $k = h^p$ and $\epsilon = 2m_1 k \left(\delta + \frac{\mu}{\sqrt{n}}\right)$, we have

$$\exp\left(-\frac{2(\epsilon - km_1(\delta + \frac{\mu}{\sqrt{n}}))^2}{h^p(\delta + \frac{\mu}{\sqrt{n}})^2 (b-a)^2}\right) = \exp\left(-\frac{2h^p m_1^2}{(b-a)^2}\right)$$

5 Criticality of a neighborhood of A^* (Proof of Theorem 3.2)

It turns out that the expectation of the full gradient of the loss function (2) is difficult to analyze directly. Hence corresponding to the true gradient with respect to the $i^{\text{th}}-$column of $W^\top$ we create a proxy, denoted by $\widehat{\nabla_i L}$, by replacing in the expression for the true expectation $\nabla_i L = \mathbb{E}\left[\frac{\partial L}{\partial W_i}\right]$ every occurrence of the random variable $\text{Th}(W_i^\top y - \epsilon_i) = \text{Th}(W_i^\top A^* x^* - \epsilon_i)$ by the indicator random variable $\mathbf{1}_{i \in \text{supp}(x^*)}$. This proxy is shown to be a good approximant of the expected gradient in the following lemma.

Lemma 5.1 *Assume that the hypotheses of Theorem 3.1 hold and additionally let b be bounded by a polynomial in h. Then we have for each i (indexing the columns of $W^\top$),*

$$\left|\left|\widehat{\nabla_i L} - \mathbb{E}\left[\frac{\partial L}{\partial W_i}\right]\right|\right|_2 \leq poly(h) exp\left(-\frac{h^p m_1^2}{2(b-a)^2}\right)$$

Lemma 5.2 *Assume that the hypotheses of Theorem 3.1 hold, and $p < \min\{\frac{1}{2}, \nu^2\}$ (and hence $\xi > 2p$). Then for each i indexing the columns of $W^\top$, there exist real valued functions α_i and β_i, and a vector e_i such that $\widehat{\nabla_i L} = \alpha_i W_i - \beta_i A_i^* + e_i$, and*

$$\begin{aligned}
\alpha_i &= \Theta(m_2 h^{p-1}) + o(m_1^2 h^{p-1}) \\
\beta_i &= \Theta(m_2 h^{p-1}) + o(m_1^2 h^{p-1}) \\
\alpha_i - \beta_i &= o(\max\{m_1^2, m_2\} h^{p-1}) \\
||e_i||_2 &= o(\max\{m_1^2, m_2\} h^{p-1})
\end{aligned}$$

The proofs for the above two lemmas in this section can be found in the full paper [18] Given the above results, we are now in a position to assemble the proof of Theorem 3.2.

Proof 5.3 (Proof of Theorem 3.2) *Consider any i indexing the columns of $W^\top$. Recall the definition of the proxy gradient $\widehat{\nabla_i L}$ at the beginning of this section. Let us define $\gamma_i = \widehat{\nabla_i L} - \mathbb{E}\left[\frac{\partial L}{\partial W_i}\right]$. Using α_i, β_i and e_i as defined in Lemma 5.2, we can write the expectation of the true gradient as, $\mathbb{E}\left[\frac{\partial L}{\partial W_i}\right] = \alpha_i W_i - \beta_i A_i^* + e_i - \gamma_i$. Further, by Lemma 5.1,*

$$\|\gamma_i\| \leq \textit{poly}(h) \textit{exp}\left(-\frac{h^p m_1^2}{2(b-a)^2}\right).$$

Since exp $\left(\frac{h^p m_1^2}{2(b-a)^2}\right)$ is superpolynomial in h, we obtain

$$\begin{aligned}
\left\| \mathbb{E}\left[\frac{\partial L}{\partial W_i}\right] \right\|_2 &= ||\alpha_i W_i - \beta_i A_i^* + e_i - \gamma_i||_2 \\
&= ||\alpha_i (W_i - A_i^*) + (\alpha_i - \beta_i) A_i^* + e_i - \gamma_i||_2 \\
&\leq |\alpha_i| \|W_i - A_i^*\|_2 + |\alpha_i - \beta_i| + ||e_i - \gamma_i||_2 \\
&\leq \frac{\Theta(m_2 h^{p-1})}{h^{2p+\theta^2}} + o(\max\{m_1^2, m_2\} h^{p-1}) \\
&+ o(\max\{m_1^2, m_2\} h^{p-1}) \\
&= o(\max\{m_1^2, m_2\} h^{p-1})
\end{aligned}$$

6 Simulations

We conduct some experiments on synthetic data in order to check whether the gradient norm is indeed small within the columnwise δ-ball of A^*. We also make some observations about the landscape of the squared loss function, which has implications for being able to recover the ground-truth dictionary A^*.

Data Generation Model We generate random gaussian dictionaries (A^*) of size $n \times h$ where $n = 50$, and $h = 256, 512, 1024, 2048$ and 4096. For each h, we generate a dataset containing $N = 5000$ sparse vectors with h^p non-zero entries, for various $p \in [0.05, 0.5]$. In our experiments, the coherence parameter ξ was approximately 0.1. The support of each sparse vector x^* is drawn uniformly from all sets of indices of size h^p, and the non-zero entries in the sparse vectors are drawn from a uniform distribution between $a = 1$ and $b = 10$. Once we have generated the sparse vectors, we collect them in a matrix $X^* \in \mathbb{R}^{h \times N}$ and then compute the signals $Y = A^* X^*$. We set up the autoencoder as defined through equation 1. We analyze the squared loss function in (2) and its gradient with respect to a column of W through their empirical averages over the signals in Y.

Results Once we have generated the data, we compute the empirical average of the gradient of the loss function in (2) at 200 random points which are columnwise $\frac{\delta}{2} = \frac{1}{2h^{2p}}$ away from A^*. We average the gradient over the 200 points which are all at the same distance from A^*, and compare the average column norm of the gradient to h^{p-1}. Our experimental results shown in Table 1 demonstrate that the average column norm of the gradient is of the order of h^{p-1} (and thus falling with h for any fixed p) as expected from Theorem 3.2.

h \ p	0.05	0.1	0.2	0.3	0.5
256	(0.0126, 0.0052)	(0.0095, 0.0068)	(0.0284, 0.0118)	(0.0464, 0.0206)	(0.0343, 0.0625)
512	(0.0054, 0.0027)	(0.0071, 0.0036)	(0.0104, 0.0068)	(0.0214, 0.0127)	(0.0028, 0.0442)
1024	(0.0026, 0.0014)	(0.0079, 0.0020)	(0.0078, 0.0039)	(0.0099, 0.0078)	(0.00, 0.0313)
2048	(0.0025, 0.0007)	(0.0031, 0.0010)	(0.0032, 0.0022)	(0.0036, 0.0048)	(0.00, 0.0221)
4096	(0.0013, 0.0004)	(0.0026, 0.0006)	(0.0020, 0.0013)	(0.0008, 0.0030)	(0.00, 0.0156)

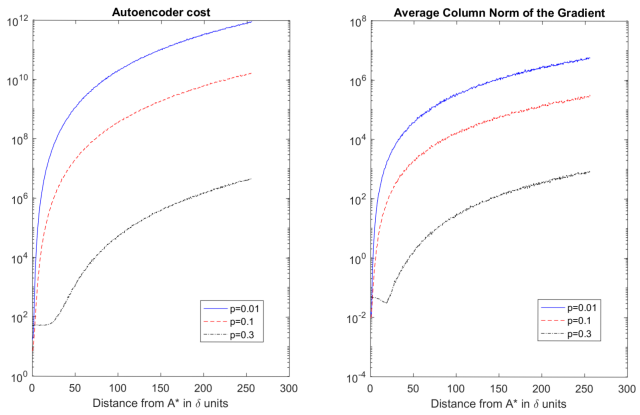
Table 1: Average gradient norm for points that are columnwise $\frac{\delta}{2}$ away from A^*. For each h and p we report $\left(||\mathbb{E}\left[\frac{\partial L}{\partial W_i}\right]||, h^{p-1}\right)$. We note that the gradient norm and h^{p-1} are of the same order, and for any fixed p the gradient norm is decreasing with h as expected from Theorem 3.2

We also plot the squared loss of the autoencoder along a randomly chosen direction to understand the geometry of the landscape of the loss function around A^*. We draw a matrix ΔW from a standard normal distribution, and normalize its columns. We then plot $f(t) = L((A^* + t\Delta W)^\top)$, as well as the gradient norm averaged over all the columns. For purposes of illustration, we show these plots for $p = 0.01, 0.1, 0.3$. The plots for $h = 256$ are in Figure 1a, and those for $h = 4096$ in Figure 1b. From the plots for $p = 0.01$ and 0.1, we can observe that the loss function value, and the gradient norm keeps decreasing as we get close to A^*. Figure 1a and 1b are representative of the shapes obtained for every direction, ΔW that we checked. This suggests that A^* might conveniently lie at the bottom of a well in the landscape of the loss function. For the value of $p = 0.3$, (which is much larger than the coherence parameter ξ), Theorem 3.1 is no longer valid. We see that the value of the loss function decreases a little as we move away from A^*, and then increases. We suspect that A^* is now in a region where $\text{ReLU}(A^{*\top} y - \epsilon) = 0$, which means the function is flat in a small neighborhood of A^*.

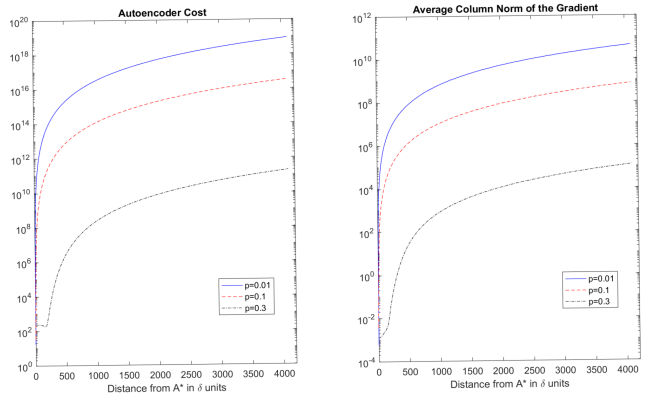
We also tried to minimize the squared loss of the autoencoder using gradient descent. In these experiments, we initialized $W^\top$ far away from A^* (precisely at a columnwise distance of $\frac{h}{5} \times \delta$), and did gradient descent until the gradient norm dropped below a factor of 2×10^{-5} of the initial norm of the gradient. We then computed the average columnwise distance between $W^\top_{\text{final}}$ and A^*, and report the $\%$ decrease in the average columnwise distance from the initial point. These results are reported in Table 2 below. These experiments suggest that there is a neighborhood of A^* (the radius of which is increasing with h), such that gradient descent initialized at the edge of that neighborhood, greatly reduces the average columnwise distance between $W^\top$ and A^*.

h	$p = 0.05$	$p = 0.1$
256	97.7%	96.9%
512	98.6%	98.2%
1024	99%	98.8%
2048	99.2%	99%
4096	99.4%	99.2%

Table 2: Fraction of initial columnwise distance covered by the gradient descent procedure



(a) Loss function plot for $h = 256, n = 50$



(b) Loss function plot for $h = 4096, n = 50$

Figure 1: Plotting the squared error loss function around the ground truth dictionary (A^*) along a randomly chosen direction

7 Conclusion

In this paper we have undertaken a rigorous analysis of the squared loss of an autoencoder when the data is assumed to be generated by sensing of sparse high dimensional vectors by an overcomplete dictionary. **We have proven and have given supporting experiments that the expected gradient of this loss function is very close to zero in a neighborhood of the generating overcomplete dictionary.**

Recent investigations have led to the conjecture/belief that many important unsupervised learning tasks, e.g. recognizing handwritten digits, are sparse coding problems in disguise [13, 14]. Thus, our results could shed some light on the observed phenomenon that gradient descent based algorithms train autoencoders to low reconstruction error for natural data sets, like MNIST.

It remains to rigorously understand the global properties of this loss function and how the use of regularizers can assist sparse coding. It remains an exciting open avenue of research to try to do a similar study as in this work to determine if and how deeper architectures under the same generative model might provide better means of doing sparse coding.

Acknowledgment

Akshay Rangamani and Peter Chin are supported by the AFOSR grant FA9550-12-1-0136. Amitabh Basu and Anirbit Mukherjee gratefully acknowledge support from the NSF grant CMMI1452820.

We would like to thank Raman Arora (JHU), and Siva Theja Maguluri (Georgia Institute of Technology) for illuminating comments and discussion.

References

[1] A. Agarwal, A. Anandkumar, P. Jain, P. Netrapalli, and R. Tandon. Learning sparsely used overcomplete dictionaries. In *COLT*, pages 123–137, 2014.

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

[3] S. Arora, A. Bhaskara, R. Ge, and T. Ma. More algorithms for provable dictionary learning. *arXiv:1401.0579*, 2014.

[4] S. Arora, R. Ge, T. Ma, and A. Moitra. Simple, efficient, and neural algorithms for sparse coding. In *COLT*, pages 113–149, 2015.

[5] S. Arora, R. Ge, and A. Moitra. New algorithms for learning incoherent and overcomplete dictionaries. In *COLT*, pages 779–806, 2014.

[6] B. Barak, J. A. Kelner, and D. Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. *CoRR*, abs/1407.1543, 2014.

[7] J. Błasiok and J. Nelson. An improved analysis of the er-spud dictionary learning algorithm. *arXiv:1602.05719*, 2016.

[8] D. Charte, F. Charte, S. García, M. J. del Jesus, and F. Herrera. A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines. *Information Fusion*, 2017.

[9] D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

[10] A. Gilbert. Cbms conference on sparse approximation and signal recovery algorithms, may 22-26, 2017 and 16th new mexico analysis seminar, may 21. *https://www.math.nmsu.edu/ jlakey/cbms2017/cbms_lecture_notes.html*, 2017.

[11] M. Janzamin, H. Sedghi, and A. Anandkumar. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473*, 2015.

[12] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with relu activation. *arXiv preprint arXiv:1705.09886*, 2017.

[13] A. Makhzani and B. Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

[14] A. Makhzani and B. J. Frey. Winner-take-all autoencoders. In *Advances in Neural Information Processing Systems*, pages 2791–2799, 2015.

[15] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607, 1996.

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

[17] B. A. Olshausen and D. J. Field. How close are we to understanding v1? *Neural computation*, 17(8):1665–1699, 2005.

[18] A. Rangamani, A. Mukherjee, A. Arora, T. Ganapathy, A. Basu, S. Chin, and T. D. Tran. Sparse coding and autoencoders. *arXiv preprint arXiv:1708.03735*, 2017.

[19] D. A. Spielman, H. Wang, and J. Wright. Exact recovery of sparsely-used dictionaries. In *COLT*, pages 37–1, 2012.

[20] J. Sun, Q. Qu, and J. Wright. Complete dictionary recovery over the sphere I: overview and the geometric picture. *CoRR*, abs/1511.03607, 2015.

[21] J. Sun, Q. Qu, and J. Wright. Complete dictionary recovery over the sphere II: recovery by riemannian trust-region method. *CoRR*, abs/1511.04777, 2015.

[22] Y. Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. *arXiv preprint arXiv:1703.00560*, 2017.

[23] A. M. Tillmann. On the computational intractability of exact and approximate dictionary learning. *IEEE Signal Processing Letters*, 22(1):45–49, 2015.

[24] Q. Zhang, R. Panigrahy, S. Sachdeva, and A. Rahimi. Electron-proton dynamics in deep learning. *arXiv preprint arXiv:1702.00458*, 2017.