Fast, Provable Algorithms for Learning Structured Dictionaries and Autoencoders

Chinmay Hegde
Iowa State University

Collaborators:
Thanh Nguyen (ISU)
Raymond Wong (Texas A&M)
Akshay Soni (Yahoo! Research)
Flavors of machine learning

Supervised learning
- Classification
- Regression
- Categorization
- Search
- ...

Unsupervised learning
- Representation learning
- Clustering
- Dimensionality reduction
- Density estimation
- ...

In the landscape of ML research:
- Supervised ML dominates not only practice...
Flavors of machine learning

Supervised learning

- Classification
- Regression
- Categorization
- Search
- ...

Unsupervised learning

- Representation learning
- Clustering
- Dimensionality reduction
- Density estimation
- ...

In the landscape of ML research:

- Supervised ML dominates not only practice ...
Flavors of machine learning

- Supervised learning
  - Classification
  - Regression
  - Categorization
  - Search
  - ...

- Unsupervised learning
  - Representation learning
  - Clustering
  - Dimensionality reduction
  - Density estimation
  - ...

In the landscape of ML research:
  - Supervised ML dominates not only practice ...
  - ... but also theory
Learning data representations

PCA was among the first attempts

PCA on 12 × 12-patches of natural images
Learning data representations

PCA was among the first attempts

PCA on $12 \times 12$-patches of natural images

not localized, visually difficult to interpret
Learning data representations

Sparse coding (Olshausen and Field, ’96)
Learning data representations

Sparse coding (Olshausen and Field, ’96)

local, oriented, interpretable
Sparse coding (a.k.a. dictionary learning):
learn an over-complete, sparse representation for a set of data points
Sparse coding (a.k.a. dictionary learning): learn an **over-complete, sparse** representation for a set of data points

\[ y \in \mathbb{R}^n \text{ (e.g. images)} \quad \approx \quad \text{dictionary } A \in \mathbb{R}^{n \times m} \quad \times \quad \text{code } x \in \mathbb{R}^m \]

- dictionary is overcomplete \((n < m)\)
- representation (code) is sparse
Mathematical formulation

**Input:** $p$ data samples: $Y = [y^{(1)}, y^{(2)}, \ldots, y^{(p)}] \in \mathbb{R}^{n \times p}$

**Goal:** find dictionary $A$ and codes $X = [x^{(1)}, x^{(2)}, \ldots, x^{(p)}] \in \mathbb{R}^{m \times p}$ that sparsely represent $Y$: 

$$\min_{A,X} \frac{1}{2} \|Y - AX\|_2^2, \quad \text{s.t.} \quad \|x^{(j)}\|_0 \leq k$$
Mathematical formulation

Input: $p$ data samples: $Y = [y^{(1)}, y^{(2)}, \ldots, y^{(p)}] \in \mathbb{R}^{n \times p}$

Goal: find dictionary $A$ and codes $X = [x^{(1)}, x^{(2)}, \ldots, x^{(p)}] \in \mathbb{R}^{m \times p}$ that sparsely represent $Y$:

$$\min_{A, X} \mathcal{L}(A, X) = \frac{1}{2} \| Y - AX \|_F^2, \quad \text{s.t.} \quad \| x^{(j)} \|_0 \leq k$$
Challenges

\[
\min_{A, X} \mathcal{L}(A, X) = \frac{1}{2} \|Y - AX\|_F^2, \quad \text{s.t.} \quad \|x^{(j)}\|_0 \leq k
\]

Two major obstacles:
Challenges

\[
\min_{A,X} \mathcal{L}(A, X) = \frac{1}{2} \|Y - AX\|_F^2, \quad \text{s.t.} \quad \|x^{(j)}\|_0 \leq k
\]

Two major obstacles:

1. **Theory**
   - Highly non-convex both in objective and constraints
   - Few provably correct algorithms (barring recent breakthroughs)
Challenges

\[
\min_{A, X} \mathcal{L}(A, X) = \frac{1}{2}\|Y - AX\|_F^2, \quad \text{s.t.} \quad \|x^{(j)}\|_0 \leq k
\]

Two major obstacles:

1. **Theory**
   - Highly non-convex both in objective and constraints
   - Few provably correct algorithms (barring recent breakthroughs)

2. **Practice**
   - Even heuristics face *memory* and *running-time* issues
   - Merely storing an estimate of $A$ requires $mn = \Omega(n^2)$ memory
Overview of our recent algorithmic work on **sparse coding**

- Autoencoder training
- Dealing with missing data
- Computational challenges
Structured dictionaries

\[ Y \approx AX \]

Key idea: impose \textbf{additional structure} on \( A \)
Structured dictionaries

\[ Y \approx AX \]

Key idea: impose **additional structure** on \( A \)

One type of structure is **double-sparsity**

- Dictionary is *itself* sparse in some fixed basis \( \Phi \)

\[ y \in \mathbb{R}^n \]

sparse comp. \( A \in \mathbb{R}^{n \times m} \)

sparse code \( x \in \mathbb{R}^m \)
Double-sparse coding$^1$

Regular sparse coding

Double-sparse coding w/ sym8 wavelets

$^1$figures reproduced using Trainlets [Sulam et al. ’16]
$Y \approx AX + \text{noise}$

<table>
<thead>
<tr>
<th>Setting</th>
<th>Approach</th>
<th>S.C (w/o noise)</th>
<th>S.C (w/ noise)</th>
<th>Run. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>K-SVD (Aharon et al '06)</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
</tr>
<tr>
<td></td>
<td>Er-SPuD (Spielman '12)</td>
<td>$O(n^2 \log n)$</td>
<td>$\times$</td>
<td>$\tilde{\Omega}(n^4)$</td>
</tr>
<tr>
<td></td>
<td>Arora et al '15</td>
<td>$\tilde{O}(mk)$</td>
<td>$\times$</td>
<td>$\tilde{O}(mn^2p)$</td>
</tr>
</tbody>
</table>
Previous work

\[ Y \approx AX + \text{noise} \]

<table>
<thead>
<tr>
<th>Setting</th>
<th>Approach</th>
<th>S.C (w/o noise)</th>
<th>S.C (w/ noise)</th>
<th>Run. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>K-SVD (Aharon et al ‘06)</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Er-SPuD (Spielman ‘12)</td>
<td>(O(n^2 \log n))</td>
<td>✗</td>
<td>(\tilde{\Omega}(n^4))</td>
</tr>
<tr>
<td></td>
<td>Arora et al ‘15</td>
<td>(\tilde{O}(mk))</td>
<td>✗</td>
<td>(\tilde{O}(mn^2p))</td>
</tr>
<tr>
<td>Double Sparse</td>
<td>Rubinstein et al ‘10</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Gribonval et al ‘15</td>
<td>(\tilde{O}(mr))</td>
<td>(\tilde{O}(mr))</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Trainlets (Sulam et al ‘16)</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

\(r: \) sparsity of columns of \(A, \ k: \) sparsity of columns of \(X\)

But **no provable, tractable** algorithms had been reported to date.
Our contributions (I)

\[ Y \approx AX + \text{noise} \]

<table>
<thead>
<tr>
<th>Setting</th>
<th>Approach</th>
<th>S.C (w/o noise)</th>
<th>S.C (w/ noise)</th>
<th>Run. Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular</td>
<td>K-SVD (Aharon et al ’06)</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Er-SPuD (Spielman ’12)</td>
<td>(O(n^2 \log n))</td>
<td>✗</td>
<td>(\tilde{\Omega}(n^4))</td>
</tr>
<tr>
<td></td>
<td>Arora et al ’15</td>
<td>(\tilde{O}(mk))</td>
<td>✗</td>
<td>(\tilde{O}(mn^2p))</td>
</tr>
<tr>
<td>Double Sparse</td>
<td>Rubinstein et al ’10</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Gribonval et al ’15</td>
<td>(\tilde{O}(mr))</td>
<td>(\tilde{O}(mr))</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td>Sulam et al ’16</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td><strong>Our method</strong></td>
<td>(\tilde{O}(mr))</td>
<td>(\tilde{O}(mr + \sigma^2_{\epsilon \frac{mnr}{k}}))</td>
<td>(\tilde{O}(mnp))</td>
</tr>
</tbody>
</table>

Setup

We assume the following **generative model**

Suppose that $p$ samples are generated$^a$ as

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

- $A^*$ is unknown, true dictionary with $r$-sparse columns
- $x^*$ has uniform $k$-sparse support with independent nonzeros

$^a$For simplicity, assume $\Phi = I$, no noise

Goal: Provably learn $A^*$ with low **sample complexity** and **running time**
Approach overview

1. Spectral initialization to obtain a coarse estimate $A_0$
2. Gradient descent to refine this estimate
Approach overview

1. **Spectral initialization** to obtain a coarse estimate $A^0$
1. **Spectral initialization** to obtain a coarse estimate $A^0$
2. **Gradient descent** to refine this estimate
Approach overview

\[
\min_{A,X} \mathcal{L}(A, X) = \frac{1}{2} \|Y - AX\|_F^2,
\]

s.t. \( \|x^{(j)}\|_0 \leq k, \quad \|A \cdot i\|_0 \leq r \)

1. **Spectral initialization** to obtain a coarse estimate of \( A^0 \)
2. **Gradient descent** to refine the initial estimate
Approach overview

\[
\min_{A,X} \mathcal{L}(A, X) = \frac{1}{2} \| Y - AX \|^2_F,
\]

s.t. \( \| x^{(j)} \|_0 \leq k, \quad \| A \cdot i \|_0 \leq r \)

1. **Spectral initialization** to obtain a coarse estimate of \( A^0 \)
2. **Gradient descent** to refine the initial estimate

Two key elements in our (double-sparse coding) setup:

1. Identity atom **supports** in initialization (a la Sparse PCA)
2. Use **projected** gradient descent onto these supports
Initialization

Intuition:

Fix samples $u, v$ such that $u = A^*\alpha, v = A^*\alpha'$, and consider a third sample $y = A^*x^*$;
Initialization

Intuition:

Fix samples $u, v$ such that $u = A^* \alpha, v = A^* \alpha'$, and consider a third sample $y = A^* x^*$; then

$$\langle y, u \rangle \langle y, v \rangle = \langle x^*, A^T A^* \alpha \rangle \langle x^*, A^T A^* \alpha' \rangle \approx \langle x^*, \alpha \rangle \langle x^*, \alpha' \rangle$$
Initialization

**Intuition:**

Fix samples $u, v$ such that $u = A^*\alpha, v = A^*\alpha'$, and consider a third sample $y = A^*x^*$; then

$$\langle y, u \rangle \langle y, v \rangle = \langle x^*, A^T A^* \alpha \rangle \langle x^*, A^T A^* \alpha' \rangle \approx \langle x^*, \alpha \rangle \langle x^*, \alpha' \rangle$$

The weight $\langle y, u \rangle \langle y, v \rangle$ is big **only** if $y$ shares an atom with **both** $u$ and $v$.
Lemma (1)

Fix samples \( u \) and \( v \). Then,

\[
e_l \triangleq \mathbb{E}[\langle y, u \rangle \langle y, v \rangle y_i^2] = \sum_{i \in U \cap V} q_i c_i \beta_i \beta_i' A_{li}^2 + o(k/m \log n)
\]

where \( q_i = \mathbb{P}[i \in S] \), \( q_{ij} = \mathbb{P}[i, j \in S] \) and \( c_i = \mathbb{E}[x_i^4 | i \in S] \).

When \( U \cap V = \{i\} \), we can **guess** the support \( R \) of \( A_{\bullet i}^* \):

- \( |e_l| > \Omega(k/mr) \) for \( l \in \text{supp}(A_{\bullet i}^*) \)
- \( |e_l| < o(k/m \log n) \) otherwise

This lets us “isolate” samples which share exactly one atom.
**Init: Key lemma (II)**

**Idea:** Similar idea lets us (coarsely) estimate the atoms themselves:

**Lemma (2)**

Define the *truncated* weighted covariance matrix:

\[
M_{u,v} \triangleq \mathbb{E}[\langle y, u \rangle \langle y, v \rangle y_R y_R^T] = \sum_{i \in U \cap V} q_i c_i \beta_i \beta_i' A_{R,i}^* A_{R,i}^T + o(k/m \log n)
\]

\[\text{where } q_i = \mathbb{P}[i \in S], \quad q_{ij} = \mathbb{P}[i, j \in S] \text{ and } c_i = \mathbb{E}[x_i^4 | i \in S].\]

When \( U \cap V = \{i\}, \)

- \( M_{u,v} \) has \( \sigma_1 > \Omega(k/m) \)
- the second \( \sigma_2 < o(k/m \log n) \)
Descent stage

Projected approximate gradient descent

Given $A^0$ from the initialization stage

1) Encode: $x^{(i)} = \text{threshold}(A^T y^{(i)})$

2) Update: $A \leftarrow A - \eta P_k(\underbrace{(AX - Y)\text{sgn}(X)^T}_g)$

Note: $g$ is a (biased) approximation of the true gradient:

$$\nabla_A \mathcal{L} = - \sum_{i=1}^{p} (y^{(i)} - Ax^{(i)})(x^{(i)})^T = -(Y - AX)X^T$$
Convergence analysis

**Intuition:** If initialized well, then gradient approximation “points” in the right direction.

**Lemma (Descent)**

Suppose that $A$ is column-wise $\delta$-close to $A^*$ and $R = \text{supp}(A_{\bullet i}^*)$, then:

$$
\langle 2g_{R,i}, A_{R,i} - A_{R,i}^* \rangle \geq \alpha \| A_{R,i} - A_{R,i}^* \|^2 + 1/(2\alpha)\|g_{R,i}\|^2 - \epsilon^2/\alpha
$$

for $\alpha = O(k/m)$ and $\epsilon^2 = O(\alpha k^2/n^2)$. 


**Convergence analysis**

**Intuition:** If initialized well, then gradient approximation “points” in the right direction.

**Lemma (Descent)**

Suppose that $A$ is column-wise $\delta$-close to $A^*$ and $R = \text{supp}(A^*_i)$, then:

$$\langle 2g_{R,i}, A_{R,i} - A^*_{R,i} \rangle \geq \alpha \|A_{R,i} - A^*_{R,i}\|^2 + 1/(2\alpha)\|g_{R,i}\|^2 - \epsilon^2/\alpha$$

for $\alpha = O(k/m)$ and $\epsilon^2 = O(\alpha k^2/n^2)$.
Empirical results

Setup setup: $\Phi = I$, $A$: 32-block diagonal with $r = 2$, $x^*$: Uniform support, Rademacher coefficients, $k = 6$
This talk

Describe our recent algorithmic work on **sparse coding**

- Computational challenges
- Dealing with missing data
- Training autoencoders
Generative model:

\[ Y \approx AX \]

What if only a random fraction \((\rho)\) of the data entries are observed?
Missing data

Generative model:

\[ Y \approx AX \]

What if only a random fraction (\( \rho \)) of the data entries are observed?

Structural assumption: Democracy

Definition (Democratic dictionaries)

\( A \) is democratic if the following holds for all columns \( i \neq j \), and for any subset \( \Gamma \) with \( \sqrt{n} \leq |\Gamma| \leq n \):

\[ \frac{\|A_{\Gamma,i} \cdot A_{\Gamma,j}\|}{\|A_{\Gamma,i}\| \cdot \|A_{\Gamma,j}\|} \leq \frac{\mu}{\sqrt{n}}. \]
Our contributions (II)

Generative model:

\[ Y \approx AX \]

Observe: only a \( \rho \)-fraction of the entries of each sample (column of \( Y \))

Theorem (Informal)

When given a sufficiently-close initial estimate \( A^0 \), there exists a gradient descent-type algorithm that linearly converges to the true dictionary with \( \tilde{O}_\rho(mk) \) incomplete samples.

Our contributions (II)

Generative model:

\[ Y \approx AX \]

Observe: only a \( \rho \)-fraction of the entries of each sample (column of \( Y \))

**Theorem (Informal)**

*When given a sufficiently-close initial estimate \( A^0 \), there exists a gradient descent-type algorithm that linearly converges to the true dictionary with \( \tilde{O}_\rho(mk) \) incomplete samples.*

Matches the sample complexity of [Arora et al, '15], but uses only incomplete samples.

Autoencoders are popular building blocks of deep networks.
Autoencoders are popular building blocks of deep networks.

Architecture of a shallow autoencoder (w/ weight sharing)

Does training such architectures with gradient descent work?
Our contributions (III)

Generative model:

\[ Y \approx AX + \text{noise} \]
Our contributions (III)

Generative model:

\[ Y \approx AX + \text{noise} \]

- \( X \): indicator vectors; noise: gaussian → mixture of gaussians
- \( X \): \( k \)-sparse → dictionary models
- \( X \): non-negative sparse → topic models
Our contributions (III)

Generative model:

\[ Y \approx AX + \text{noise} \]

- \( X \): indicator vectors; noise: gaussian → mixture of gaussians
- \( X \): \( k \)-sparse → dictionary models
- \( X \): non-negative sparse → topic models

Theorem (Autoencoder training)

Autoencoders, trained with gradient descent over the squared-error loss (with column-wise normalization), provably learn the parameters of the above generative models.

Summary

New family of sparse coding algorithms that enjoy **provable statistical and algorithmic guarantees**
New family of sparse coding algorithms that enjoy **provable statistical and algorithmic guarantees**

- *time- and memory-efficient*
- *robust to missing data*
- *connections with autoencoder learning*
Summary

New family of sparse coding algorithms that enjoy **provable statistical and algorithmic guarantees**

- *time- and memory-efficient*
- *robust to missing data*
- *connections with autoencoder learning*

Open questions:

- Other dictionary structures? (convolutional, Kronecker)
- Independent components analysis
- Analyzing deeper autoencoder architectures