Sparse Coding and Dictionary Learning

Vahid Tarokh

ECE 685D, Fall 2025

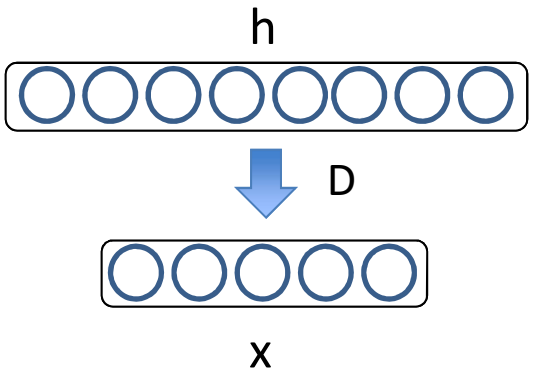
Unsupervised Learning

- Unsupervised learning: we only use the inputs $\mathbf{x}^{(t)}$ for learning
  - automatically extract meaningful features for your data
  - leverage the availability of unlabeled data
  - add a data-dependent regularizer to training (in some cases)

- We will consider 3 models for unsupervised learning that will form the basic building blocks for deeper models:
  - Autoencoders
  - Sparse Coding
  - Restricted Boltzmann Machines

Sparse Coding

- Sparse coding was originally developed in AI literature to explain early visual processing in the brain (edge detection) [Olshausen & Field, 1996].
- For each input $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - The representation is sparse: the vector $\mathbf{h}^{(t)}$ has many zeros
  - It can be used to well reconstruct the original input $\mathbf{x}^{(t)}$



Sparse Coding

- For each $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - The representation is sparse: the vector $\mathbf{h}^{(t)}$ has many zeroes
  - It can be used to well reconstruct the original input $\mathbf{x}^{(t)}$

- In other words:

Reconstruction: $\widehat{\mathbf{x}}^{(t)}$

Sparsity vs. reconstruction control

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\,\mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

Reconstruction error

Sparsity penalty

Sparse Coding

- For each $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - The representation is sparse: the vector $\mathbf{h}^{(t)}$ has many zeroes
  - It can be used to well reconstruct the original input $\mathbf{x}^{(t)}$
- In other words:

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

  - we also constrain the columns of D to be of norm 1
  - otherwise, D could grow big while h becomes small to satisfy the L_1 constraint

Sparse Coding

- For each $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - The representation is sparse: the vector $\mathbf{h}^{(t)}$ has many zeroes
  - It can be used to well reconstruct the original input $\mathbf{x}^{(t)}$

- In other words:

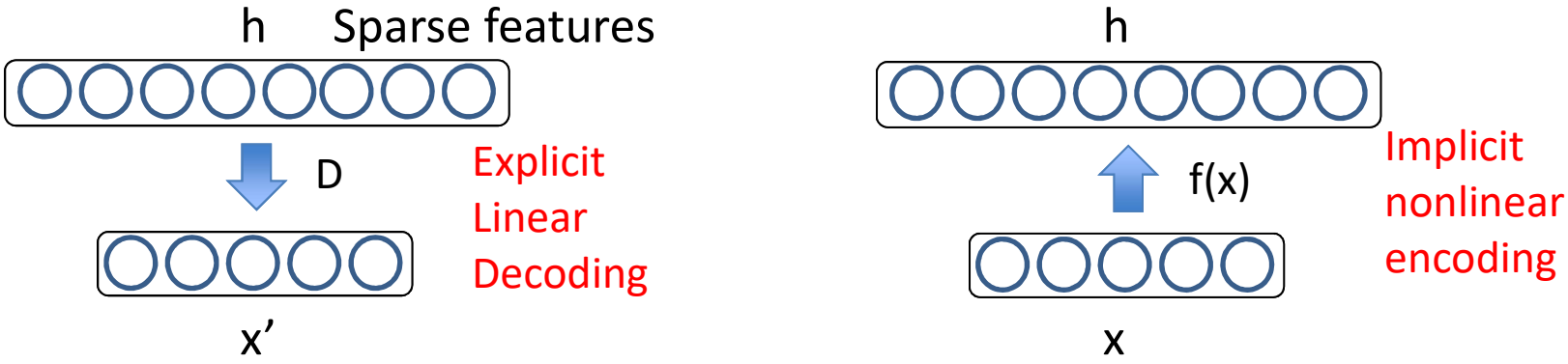
$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

  - D is equivalent to the autoencoder output weight matrix
  - However, $\mathbf{h}(\mathbf{x}^{(t)})$ is now a complicated function of $\mathbf{x}^{(t)}$
  - Encoder is the minimization problem:

$$\mathbf{h}(\mathbf{x}^{(t)}) = \arg\min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

Interpreting Sparse Coding

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\,\mathbf{h}^{(t)}||_2^2 + \lambda ||\mathbf{h}^{(t)}||_1$$

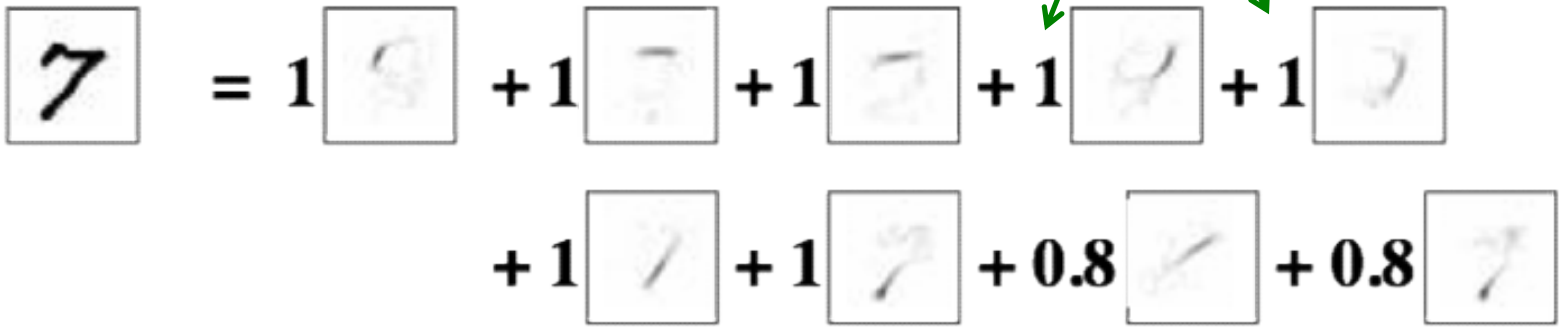


- Sparse, over-complete representation **h.**
- Encoding **h** = f(**x**) is implicit and nonlinear function of **x**.
- Reconstruction (or decoding) **x'** = Dh is linear and explicit.

Sparse Coding

- We can also write:

$$\hat{\mathbf{x}}^{(t)} = \mathbf{D}\,\mathbf{h}(\mathbf{x}^{(t)}) = \sum_{k \text{ s.t. } h(\mathbf{x}^{(t)})_k \neq 0} \mathbf{D}_{.,k}\, h(\mathbf{x}^{(t)})_k$$

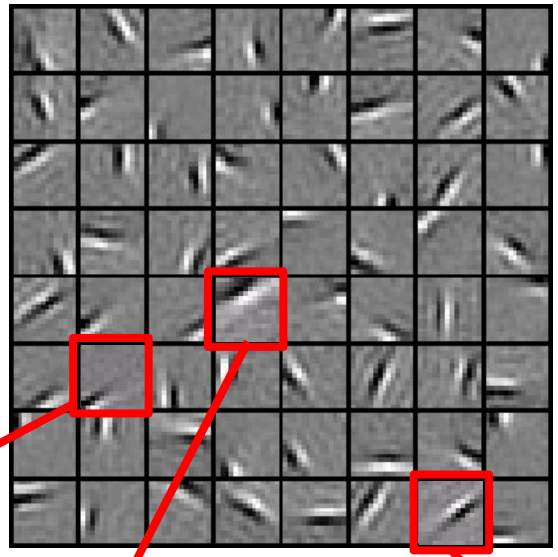


- D is often referred to as Dictionary
- In certain applications, we know what dictionary matrix to use
- In many cases, we have to learn it

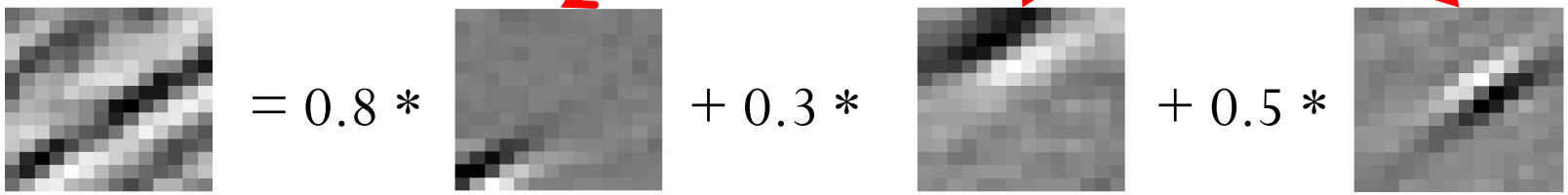
Sparse Coding

Natural Images

Learned bases: "Edges"



New example



$$x = 0.8 * \phi_{36} + 0.3 * \phi_{42} + 0.5 * \phi_{65}$$

[0, 0, … **0.8**, …, **0.3**, …, **0.5**, …] = coefficients (feature representation)

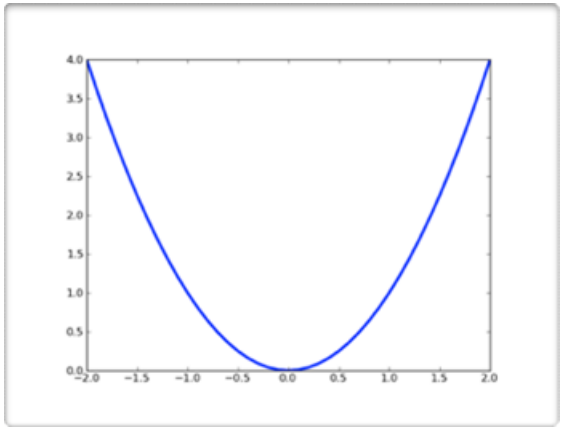
Slide Credit: Honglak Lee

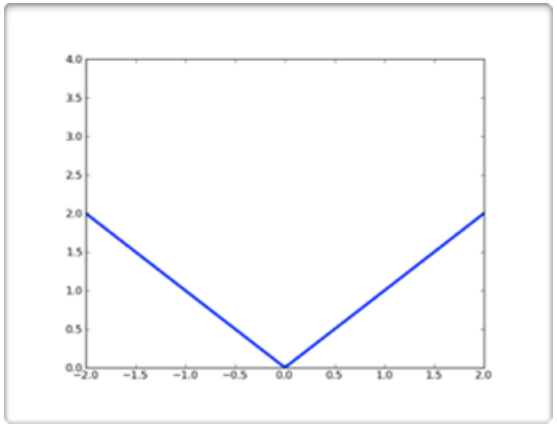
Inference

- Given dictionary D , how do we compute $\mathbf{h}(\mathbf{x}^{(t)})$?
  - We need to optimize:

$$l(\mathbf{x}^{(t)}) = \frac{1}{2}||\mathbf{x}^{(t)} - \mathbf{D}\,\mathbf{h}^{(t)}||_2^2 + \lambda||\mathbf{h}^{(t)}||_1$$

  - This is Lasso.





  - We could use a gradient descent method:

$$\nabla_{\mathbf{h}^{(t)}} l(\mathbf{x}^{(t)}) = \mathbf{D}^\top(\mathbf{D}\,\mathbf{h}^{(t)} - \mathbf{x}^{(t)}) + \lambda\,\mathrm{sign}(\mathbf{h}^{(t)})$$

Inference

- For a single hidden unit:

$$\frac{\partial}{\partial h_k^{(t)}} l(\mathbf{x}^{(t)}) = (\mathbf{D}_{\cdot,k})^\top (\mathbf{D}\,\mathbf{h}^{(t)} - \mathbf{x}^{(t)}) + \lambda\,\mathrm{sign}(h_k^{(t)})$$

  - issue: L_1 norm not differentiable at 0
  - very unlikely for gradient descent to "land" on $h_k^{(t)} = 0$ (even if it's the solution)

- Solution: if $h_k^{(t)}$ changes sign because of L_1 norm gradient, clamp to 0.

Inference

- For a single hidden unit:

$$\frac{\partial}{\partial h_k^{(t)}} l(\mathbf{x}^{(t)}) = (\mathbf{D}_{\cdot,k})^\top (\mathbf{D}\ \mathbf{h}^{(t)} - \mathbf{x}^{(t)}) + \lambda\, \mathrm{sign}(h_k^{(t)})$$

- Solution: if $h_k^{(t)}$ changes sign because of L1 norm gradient, clamp to 0.

- Each hidden unit update would be performed as follows:

  - $h_k^{(t)} \Longleftarrow h_k^{(t)} - \alpha (\mathbf{D}_{\cdot,k})^\top (\mathbf{D}\ \mathbf{h}^{(t)} - \mathbf{x}^{(t)})$ — Update from reconstruction
  - If $\mathrm{sign}(h_k^{(t)}) \neq \mathrm{sign}(h_k^{(t)} - \alpha\ \lambda\ \mathrm{sign}(h_k^{(t)}))$ then $h_k^{(t)} \Longleftarrow 0$
  - Else $h_k^{(t)} \Longleftarrow h_k^{(t)} - \alpha\ \lambda\ \mathrm{sign}(h_k^{(t)})$ — Update sparsity term

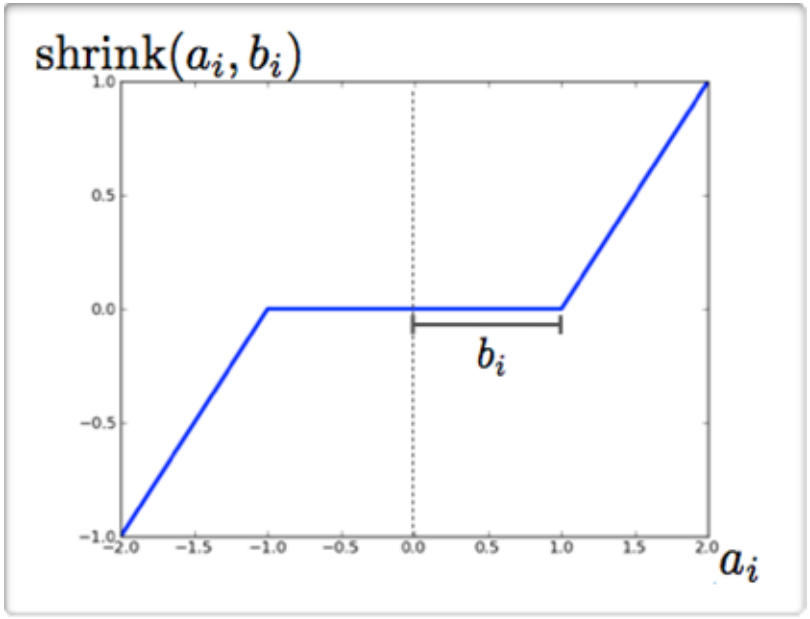
ISTA Algorithm

• This process corresponds to the ISTA (Iterative Shrinkage and Thresholding) Algorithm:

- Initialize $\mathbf{h}^{(t)}$ (for example to 0)
- While $\mathbf{h}^{(t)}$ has not converged

$$\mathbf{h}^{(t)} \Longleftarrow \mathbf{h}^{(t)} - \alpha\, \mathbf{D}^\top(\mathbf{D}\, \mathbf{h}^{(t)} - \mathbf{x}^{(t)})$$

$$\mathbf{h}^{(t)} \Longleftarrow \text{shrink}(\mathbf{h}^{(t)}, \alpha\, \lambda)$$



where

$$\text{shrink}(\mathbf{a}, \mathbf{b}) = [\ldots, \text{sign}(a_i)\ \max(|a_i| - b_i, 0), \ldots]$$

• Will converge if $\frac{1}{\alpha}$ is bigger than the largest eigenvalue of $\mathbf{D}^\top\mathbf{D}$

Dictionary Learning I

- Remember our optimization problem:

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} l(\mathbf{x}^{(t)}) = \min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}(\mathbf{x}^{(t)})||_2^2 + \lambda ||\mathbf{h}(\mathbf{x}^{(t)})||_1$$

- Let us first assume that $\mathbf{h}(\mathbf{x}^{(t)})$ does not depend on D
  - We then minimize:

$$\min_{\mathbf{D}} \frac{1}{T} \sum_{t=1}^{T} \frac{1}{2} ||\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}(\mathbf{x}^{(t)})||_2^2$$

  - we must also constrain the columns of D to be of unit norm

column norm constraints: ‖D:,j‖2=1, ∀j

Dictionary Learning I

- We can use projected gradient descent algorithm.

  - While D has not converged:
    - Perform gradient update of D

$$\mathbf{D} \Longleftarrow \mathbf{D} + \alpha \frac{1}{T} \sum_{t=1}^{T} (\mathbf{x}^{(t)} - \mathbf{D}\, \mathbf{h}(\mathbf{x}^{(t)}))\, \mathbf{h}(\mathbf{x}^{(t)})^{\top}$$

    - Renormalize the columns of D
    - For each column of D:

$$\mathbf{D}_{\cdot,j} \Longleftarrow \frac{\mathbf{D}_{\cdot,j}}{||\mathbf{D}_{\cdot,j}||_2}$$

Dictionary Learning II

- An alternative method is to solve for each column $\mathbf{D}_{\cdot,j}$ in cycle.
  - setting the gradient for $\mathbf{D}_{\cdot,j}$ to zero, we have

$$
\begin{aligned}
0 &= \frac{1}{T}\sum_{t=1}^{T}(\mathbf{x}^{(t)} - \mathbf{D}\,\mathbf{h}(\mathbf{x}^{(t)}))\,h(\mathbf{x}^{(t)})_j \\
0 &= \sum_{t=1}^{T}\left(\mathbf{x}^{(t)} - \left(\sum_{i\neq j}\mathbf{D}_{\cdot,i}\,h(\mathbf{x}^{(t)})_i\right) - \mathbf{D}_{\cdot,j}\,h(\mathbf{x}^{(t)})_j\right)\,h(\mathbf{x}^{(t)})_j \\
\sum_{t=1}^{T}\mathbf{D}_{\cdot,j}h(\mathbf{x}^{(t)})_j^2 &= \sum_{t=1}^{T}\left(\mathbf{x}^{(t)} - \left(\sum_{i\neq j}\mathbf{D}_{\cdot,i}\,h(\mathbf{x}^{(t)})_i\right)\right)\,h(\mathbf{x}^{(t)})_j \\
\mathbf{D}_{\cdot,j} &= \frac{1}{\sum_{t=1}^{T}h(\mathbf{x}^{(t)})_j^2}\sum_{t=1}^{T}\left(\mathbf{x}^{(t)} - \left(\sum_{i\neq j}\mathbf{D}_{\cdot,i}\,h(\mathbf{x}^{(t)})_i\right)\right)\,h(\mathbf{x}^{(t)})_j
\end{aligned}
$$

  - Note that we don't need to specify a learning rate to update D.

Dictionary Learning II

- An alternative method is to solve for each column $\mathbf{D}_{\cdot,j}$ in cycle.
  - We can rewrite

$$\mathbf{D}_{\cdot,j} = \frac{1}{\sum_{t=1}^{T} h(\mathbf{x}^{(t)})_j^2} \sum_{t=1}^{T} \left(\mathbf{x}^{(t)} - \left(\sum_{i \neq j} \mathbf{D}_{\cdot,i}\, h(\mathbf{x}^{(t)})_i \right) \right) h(\mathbf{x}^{(t)})_j$$

$$= \frac{1}{\sum_{t=1}^{T} h(\mathbf{x}^{(t)})_j^2} \left(\left(\sum_{t=1}^{T} \mathbf{x}^{(t)} h(\mathbf{x}^{(t)})_j \right) - \sum_{i \neq j} \mathbf{D}_{\cdot,i} \left(\sum_{t=1}^{T} h(\mathbf{x}^{(t)})_i h(\mathbf{x}^{(t)})_j \right) \right)$$

$$= \frac{1}{A_{j,j}} (\mathbf{B}_{\cdot,j} - \mathbf{D}\ \mathbf{A}_{\cdot,j} + \mathbf{D}_{\cdot,j} A_{j,j})$$

  - this way, we only need to store:

$$\mathbf{A} \Longleftarrow \sum_{t=1}^{T} \mathbf{h}(\mathbf{x}^{(t)})\, \mathbf{h}(\mathbf{x}^{(t)})^\top$$

$$\mathbf{B} \Longleftarrow \sum_{t=1}^{T} \mathbf{x}^{(t)}\, \mathbf{h}(\mathbf{x}^{(t)})^\top$$

Dictionary Learning II

- This leads to the following algorithm

  - While D has not converged:
    - for each column $\mathbf{D}_{.,j}$ perform updates

$$\mathbf{D}_{.,j} \Longleftarrow \frac{1}{A_{j,j}} \left(\mathbf{B}_{.,j} - \mathbf{D}\ \mathbf{A}_{.,j} + \mathbf{D}_{.,j}\ A_{j,j}\right)$$

$$\mathbf{D}_{.,j} \Longleftarrow \frac{\mathbf{D}_{.,j}}{||\mathbf{D}_{.,j}||_2}$$

- This is referred to as a block-coordinate descent algorithm
  - a different block of variables are updated at each step
  - the "blocks" are the columns $\mathbf{D}_{.,j}$

Learning Sparse Coding Model

• Putting it all together, we have the following algorithm, where learning alternates between inference and dictionary learning.

• While D has not converged:

- find the sparse codes $\mathbf{h}(\mathbf{x}^{(t)})$ for all $\mathbf{x}^{(t)}$ in the training set with ISTA
- Update the dictionary:

$$\mathbf{A} \Longleftarrow \sum_{t=1}^{T} \mathbf{x}^{(t)}\ \mathbf{h}(\mathbf{x}^{(t)})^{\top}$$

$$\mathbf{B} \Longleftarrow \sum_{t=1}^{T} \mathbf{h}(\mathbf{x}^{(t)})\ \mathbf{h}(\mathbf{x}^{(t)})^{\top}$$

$$\mathbf{D}_{\cdot,j} \Longleftarrow \frac{1}{A_{j,j}} (\mathbf{B}_{\cdot,j} - \mathbf{D}\ \mathbf{A}_{\cdot,j} + \mathbf{D}_{\cdot,j}\ A_{j,j})$$

$$\mathbf{D}_{\cdot,j} \Longleftarrow \frac{\mathbf{D}_{\cdot,j}}{||\mathbf{D}_{\cdot,j}||_2}$$

Instead of computing A and B from scratch over all data, we maintain running averages that update with each new sample.

Online Learning

- This algorithm is "batch" (i.e. not online)
  - single update of the dictionary per pass on the training set
  - for large datasets, we'd like to update D after visiting each $\mathbf{x}^{(t)}$

- Solution: for each input $\mathbf{x}^{(t)}$
  - perform inference of $\mathbf{h}(\mathbf{x}^{(t)})$ for the current input
  - update running averages of the quantities required to update D:

$$\mathbf{B} \Longleftarrow \beta\, \mathbf{B} + (1-\beta)\, \mathbf{x}^{(t)}\, \mathbf{h}(\mathbf{x}^{(t)})^\top$$

$$\mathbf{A} \Longleftarrow \beta\, \mathbf{A} + (1-\beta)\, \mathbf{h}(\mathbf{x}^{(t)})\, \mathbf{h}(\mathbf{x}^{(t)})^\top$$

Online Learning

- While D has not converged:
  - For each $\mathbf{x}^{(t)}$
    - Infer code $\mathbf{h}(\mathbf{x}^{(t)})$
    - Update the dictionary:

$$\mathbf{A} \Longleftarrow \beta\ \mathbf{A} + (1-\beta)\ \mathbf{h}(\mathbf{x}^{(T+1)})\ \mathbf{h}(\mathbf{x}^{(T+1)})^\top$$

$$\mathbf{B} \Longleftarrow \beta\ \mathbf{B} + (1-\beta)\ \mathbf{x}^{(t)}\ \mathbf{h}(\mathbf{x}^{(t)})^\top$$

    - while D hasn't converged
      - for each column of D perform gradient update

$$\mathbf{D}_{\cdot,j} \Longleftarrow \frac{1}{A_{j,j}} (\mathbf{B}_{\cdot,j} - \mathbf{D}\ \mathbf{A}_{\cdot,j} + \mathbf{D}_{\cdot,j}\ A_{j,j})$$

$$\mathbf{D}_{\cdot,j} \Longleftarrow \frac{\mathbf{D}_{\cdot,j}}{||\mathbf{D}_{\cdot,j}||_2}$$

Online Dictionary Learning for Sparse Coding. Mairal, Bach, Ponce and Sapiro, 2009.

Whitening

- Before running a sparse coding algorithm, it is beneficial to remove "obvious" structure from the data
  - normalize such that mean is 0 and covariance is the identity (whitening)
  - this will remove 1st and 2nd order statistical structure

- preprocessing
  - let the empirical mean be $\widehat{\boldsymbol{\mu}}$ and the empirical covariance matrix be $\widehat{\Sigma} = \mathbf{U}\Lambda\mathbf{U}^\top$(in its eigenvalue/eigenvector representation)
  - ZCA transforms each input as follows:

$$\mathbf{x} \Longleftarrow \mathbf{U}\ \Lambda^{-\frac{1}{2}}\ \mathbf{U}^\top(\mathbf{x} - \widehat{\boldsymbol{\mu}})$$

Whitening

- After this transformation
  - the empirical mean is 0

$$
\begin{aligned}
& \frac{1}{T}\sum_t \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top(\mathbf{x}^{(t)}-\widehat{\boldsymbol{\mu}}) \\
= \quad & \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\left(\left(\frac{1}{T}\sum_t \mathbf{x}^{(t)}\right)-\widehat{\boldsymbol{\mu}}\right) \\
= \quad & \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top(\widehat{\boldsymbol{\mu}}-\widehat{\boldsymbol{\mu}}) \\
= \quad & 0
\end{aligned}
$$

Whitening

- After this transformation
  - the empirical covariance matrix is the identity

$$\frac{1}{T-1}\sum_t \left(\mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top(\mathbf{x}^{(t)}-\widehat{\boldsymbol{\mu}})\right)\left(\mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top(\mathbf{x}^{(t)}-\widehat{\boldsymbol{\mu}})\right)^\top$$

$$\begin{aligned}
&= \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\left(\frac{1}{T-1}\sum_t(\mathbf{x}^{(t)}-\widehat{\boldsymbol{\mu}})(\mathbf{x}^{(t)}-\widehat{\boldsymbol{\mu}})\right)^\top\mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\\
&= \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\,\widehat{\Sigma}\,\mathbf{U}\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\\
&= \mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\,\mathbf{U}\,\Lambda\,\mathbf{U}^\top\,\mathbf{U}\,\Lambda^{-\frac{1}{2}}\,\mathbf{U}^\top\\
&= \mathbf{I}
\end{aligned}$$

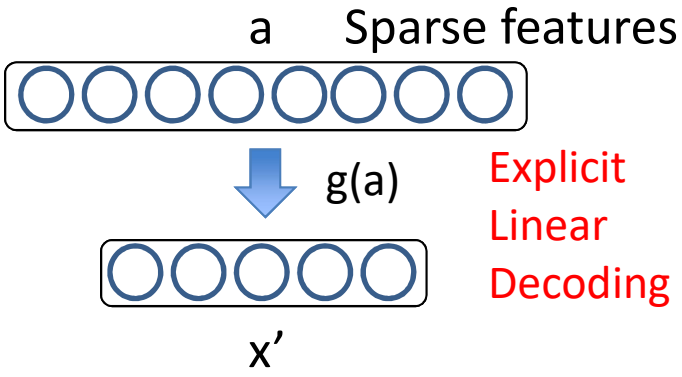
Feature Learning

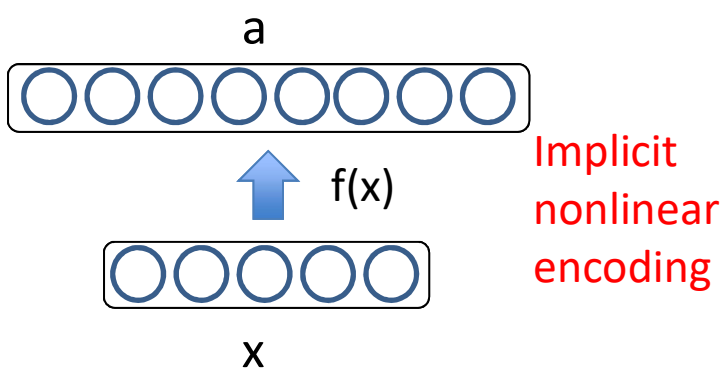
- A sparse coding model can be used to extract features
  - given a labeled training set $\{(\mathbf{x}^{(t)}, y^{(t)})\}$
  - train sparse coding dictionary only on training inputs $\{\mathbf{x}^{(t)}\}$
  - this yields a dictionary from which to infer sparse codes
  - train your favorite classifier on transformed training set $\{(\mathbf{h}(\mathbf{x}^{(t)}), y^{(t)})\}$

- When classifying test input x
  - infer its sparse representation: $\mathbf{h}(\mathbf{x})$
  - feed it to the classifier

Interpreting Sparse Coding

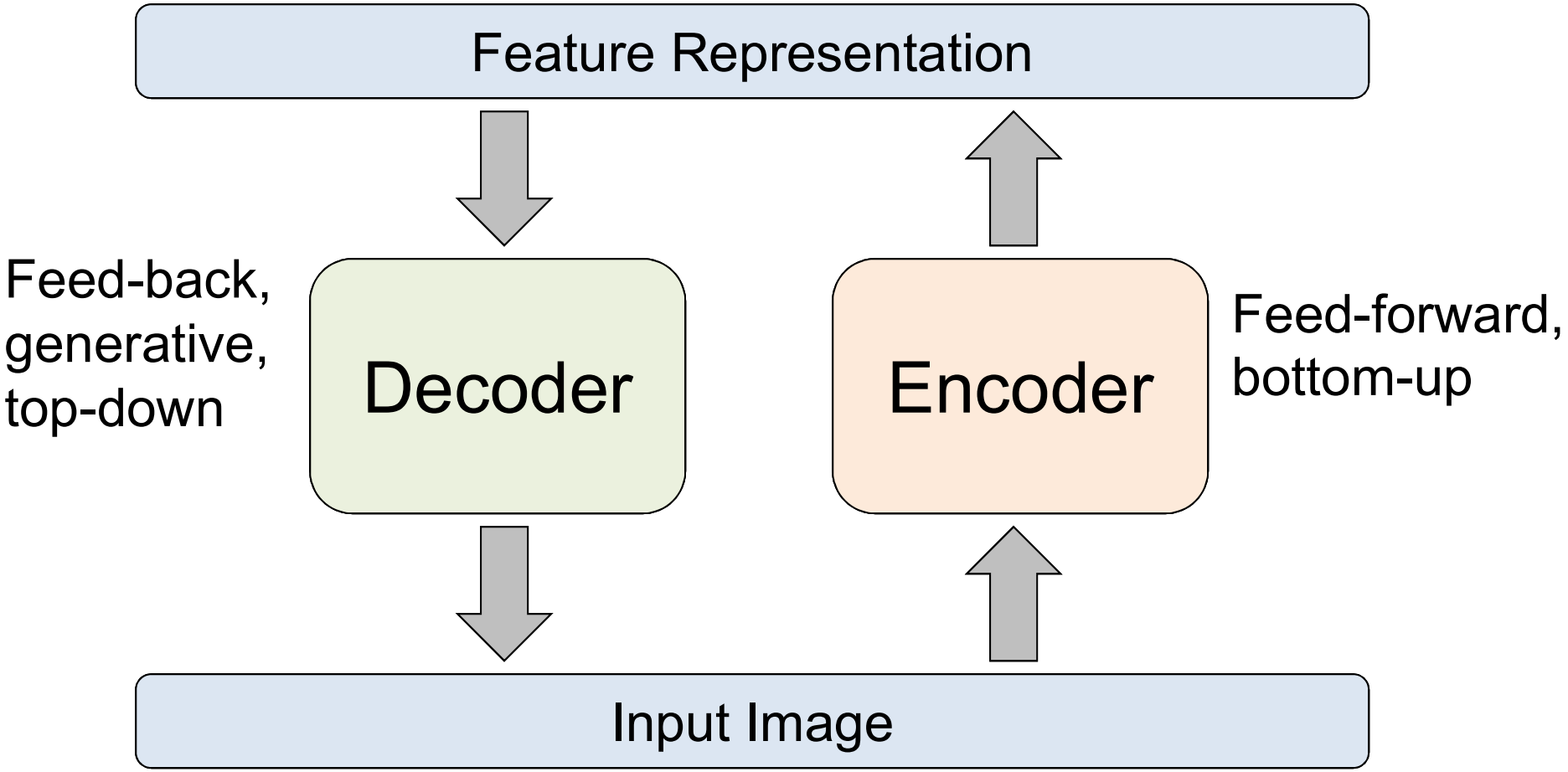
$$\min_{\mathbf{a},\phi} \sum_{n=1}^{N} \left|\left| \mathbf{x}_n - \sum_{k=1}^{K} a_{nk}\phi_k \right|\right|_2^2 + \lambda \sum_{n=1}^{N}\sum_{k=1}^{K} |a_{nk}|$$





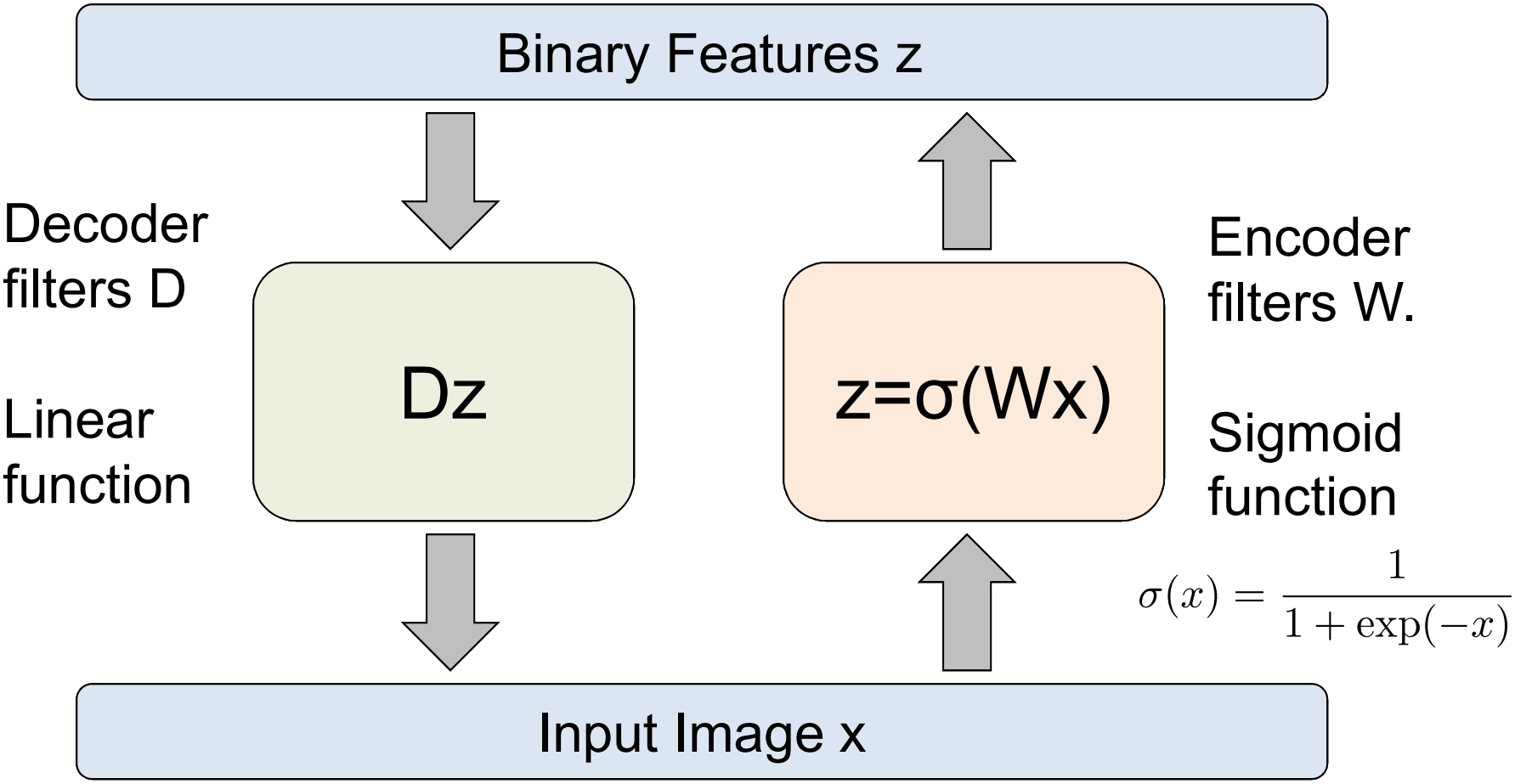
- Sparse, over-complete representation **a.**
- Encoding **a** = f(**x**) is implicit and nonlinear function of **x**.
- Reconstruction (or decoding) **x'** = g(**a**) is linear and explicit.

Autoencoder

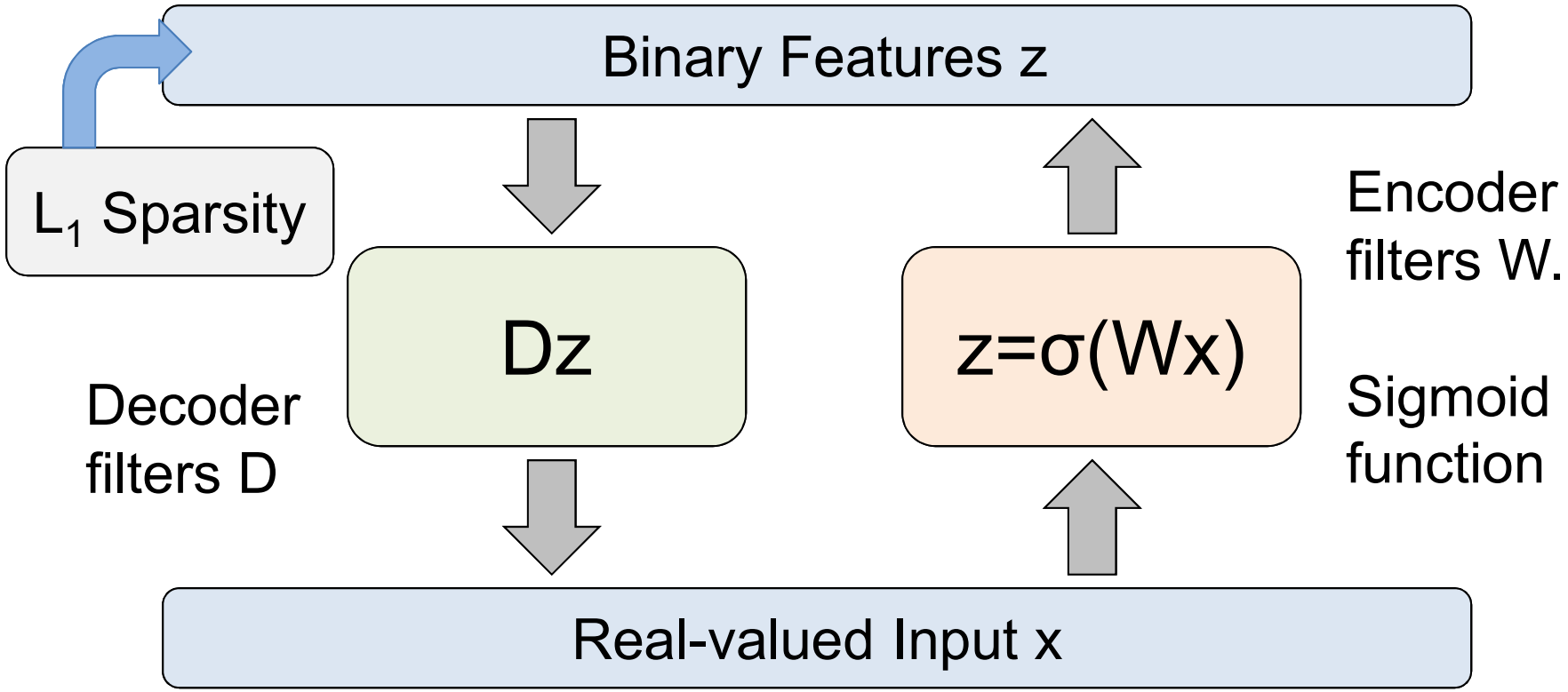


- Details of what goes insider the encoder and decoder matter!
- Need constraints to avoid learning an identity.

Autoencoder



Predictive Sparse Decomposition



At training time

$$\min_{D,W,\mathbf{z}} \underbrace{||D\mathbf{z}-\mathbf{x}||_2^2+\lambda|\mathbf{z}|_1}_{\text{Decoder}} + \underbrace{||\sigma(W\mathbf{x})-\mathbf{z}||_2^2}_{\text{Encoder}}$$

Kavukcuoglu et al., '09

Stacked Sparse Coding?