Sparse Coding

Machine Learning: A probabilistic perspective-Kevin P Murphy (Chapter 13.8)

Sparse coding

- Use of sparse priors for unsupervised learning
- If the problem utilizes a non-Gaussian prior which is sparsity promoting (such as a Laplace distribution), we will be observing each observed vector as a sparse combination of basis vectors (columns of $\boldsymbol{W}$);
- In such a case, when there is no constraint on the orthogonality of $\boldsymbol{W}$, we get a method called **sparse coding.**
- In this context, we call the factor loading matrix $\boldsymbol{W}$ a **dictionary;** each column is referred to as an **atom.**
- In sparse coding, the dictionary can be fixed or learned. If it is fixed, it is common to use a wavelet or DCT basis, since many natural signals can be well approximated by a small number of such basis functions.

Sparse coding

- However, it is also possible to learn the dictionary $\mathcal{D}$, by maximizing the likelihood

$$\log p(\mathcal{D}|\boldsymbol{W}) = \sum_{i=1}^{N} \log \int_{\boldsymbol{z}_i} \mathcal{N}(\mathbf{x}_i|\boldsymbol{W}\boldsymbol{z}_i, \sigma^2\boldsymbol{I}) p(\boldsymbol{z}_i) d\boldsymbol{z}_i$$

- Here, $\boldsymbol{z}_i$ are the latent factors. In sparse coding, a sparsity promoting prior is put on the latent factors, $\boldsymbol{z}_i$.

Learning a sparse coding dictionary

- The above equation is a hard objective to minimize, hence the following approximation is made

$$\log p(\mathcal{D}|\boldsymbol{W}) \approx \sum_{i=1}^{N} \max_{\boldsymbol{z}_i} [\log \mathcal{N}(\mathbf{x}_i|\boldsymbol{W}\boldsymbol{z}_i, \sigma^2\boldsymbol{I}) + \log p(\boldsymbol{z}_i)]$$

- If $p($ A random variable has a $\mathrm{Laplace}(\mu, b)$ distribution if its probability density function is ve log l

$$f(x \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

$$= \frac{1}{2b} \begin{cases} \exp\left(-\frac{\mu - x}{b}\right) & \text{if } x < \mu \\ \exp\left(-\frac{x-\mu}{b}\right) & \text{if } x \geq \mu \end{cases}$$

Learning a sparse coding dictionary

- To prevent W from becoming arbitrarily large, it is common to constrain the norm of its columns to be less than or equal to 1. Denote this constraint as
$$C = \{\boldsymbol{W} \in \mathbb{R}^{D \times L} \;\; s.t \; \boldsymbol{w}_j^T \boldsymbol{w}_j \leq 1\}$$
- Then we want to solve $\min_{\boldsymbol{W} \in C, \boldsymbol{Z} \in \mathbb{R}^{\boldsymbol{L} \times \boldsymbol{N}}} NLL(\boldsymbol{W}, \boldsymbol{Z})$
- For a fixed $\boldsymbol{z}_i$, the optimization over $\boldsymbol{W}$ is a simple least squares problem. And for a fixed dictionary $\boldsymbol{W}$, the optimization problem over $\boldsymbol{Z}$ is identical to the lasso problem, for which many fast algorithms exist.
- This suggests an obvious iterative optimization scheme, in which we alternate between optimizing $\boldsymbol{W}$ and $\boldsymbol{Z}$.

- A variety of other models result in an optimization problem that is similar to sparse coding.
- For example, **non-negative matrix factorization or NMF** requires solving an objective of the form:

$$\min_{\boldsymbol{W}\in C,\boldsymbol{Z}\in\mathbb{R}^{L\times N}} \frac{1}{2}\sum_{i=1}^{N}\left|\left|\mathbf{x}_i-\boldsymbol{W}\boldsymbol{z}_i\right|\right|_2^2 \quad \text{s. t } \boldsymbol{W}\geq 0, \boldsymbol{z}_i\geq 0$$

- The intuition behind this constraint is that the learned dictionary may be more interpretable if it is a positive sum of positive "parts", rather than a sparse sum of atoms that may be positive or negative.
- When NMF is combined with a sparsity promoting prior on the latent factors, it is called **non-negative sparse coding**

Sparse coding

- Following is a summary of the various latent models
- Summary of abbreviations: PCA = principal components analysis; FA = factor analysis; ICA = independent components analysis; MF matrix factorization.

Method	$p(\mathbf{z}_i)$	$p(\mathbf{W})$	$\mathbf{W}$ orthogonal
PCA	Gauss	-	yes
FA	Gauss	-	no
ICA	Non-Gauss	-	yes
Sparse coding	Laplace	-	no
Sparse PCA	Gauss	Laplace	maybe
Sparse MF	Laplace	Laplace	no

Table 13.3 Summary of various latent factor models. A dash "-" in the p(W) column means ML parameter estimation is performed rather than MAP parameter estimation.

Sparse matrix factorization

- Alternatively, we can drop the positivity constraint, but impose a sparsity constraint on both the factors $\boldsymbol{z}_i$ and the dictionary **W**. We call this **sparse matrix factorization.**
- To ensure strict convexity, we can use an elastic net type penalty on the weights resulting in

$$min_{\boldsymbol{W},\boldsymbol{Z}} \frac{1}{2} \sum_{i=1}^{N} \frac{1}{2} \left|\left|\mathbf{x}_i - \boldsymbol{W}\boldsymbol{z}_i\right|\right|_2^2 + \lambda \left|\left|\boldsymbol{z}_i\right|\right|_1$$

$$s.t \ \left|\left|w_j\right|\right|_2^2 + \gamma \left|\left|w_j\right|\right|_1 \leq 1$$

Results of dictionary learning from image patches

- The dictionary that is learned by applying sparse coding to patches of natural images consists of basis vectors that look like the filters that are found in simple cells in the primary visual cortex of the mammalian brain.
- In particular, the filters look like bar and edge detectors, as shown in Figure 13.21(b).
- Interestingly, using ICA gives visually similar results, as shown in Figure 13.21(a). By contrast, applying PCA to the same data results in sinusoidal gratings, as shown in Figure 13.21(c); these do not look like cortical cell response patterns.
- It has therefore been conjectured that parts of the cortex may be performing sparse coding of the sensory input; the resulting latent representation is then further processed by higher levels of the brain.

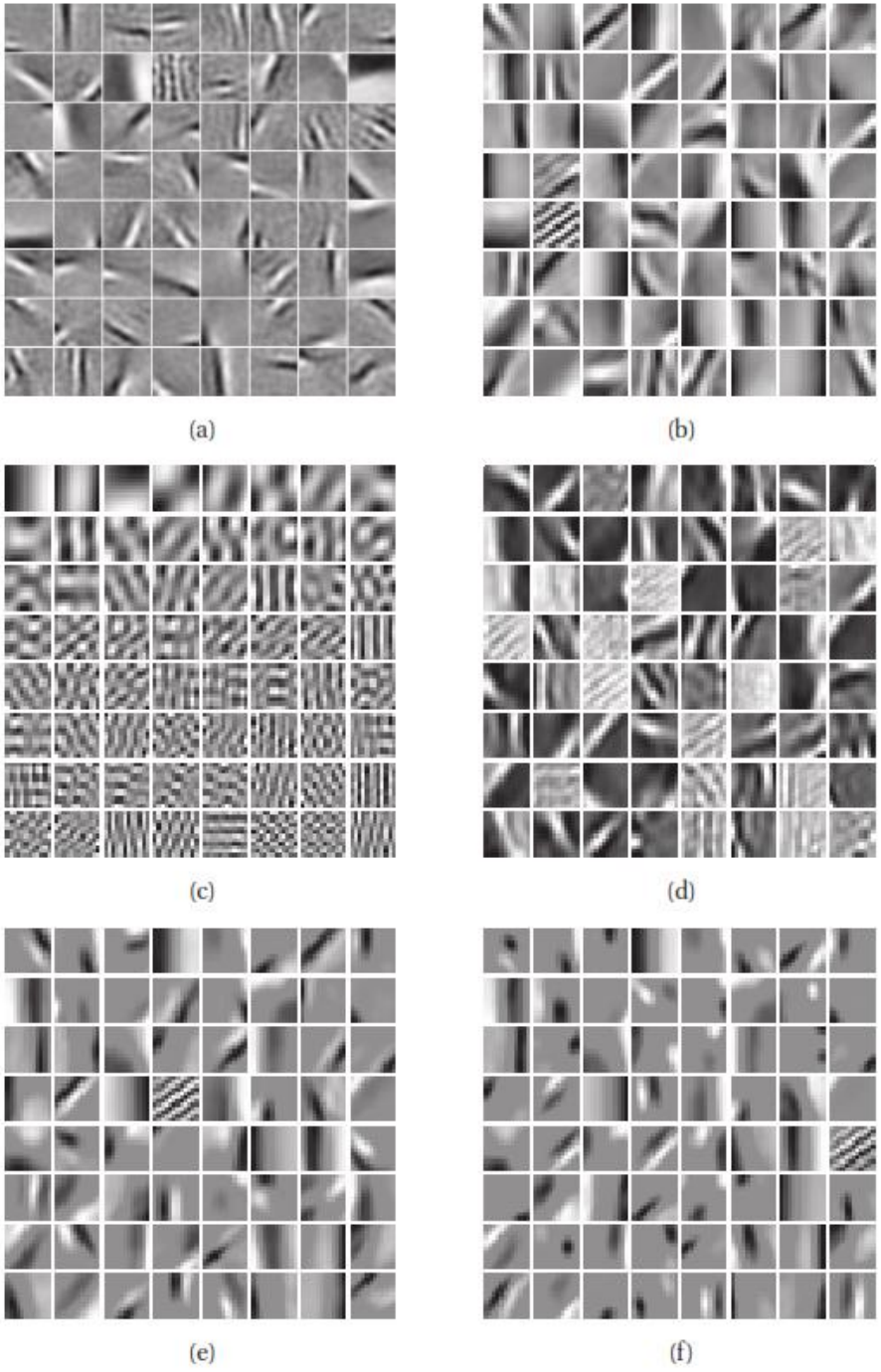

Figure 13.21 Illustration of the filters learned by various methods when applied to natural image patches:

(Each patch is first centered and normalized to unit norm.) (a) ICA. (b) sparse coding. (c) PCA. (d) non-negative matrix factorization. (e) sparse PCA with low sparsity on weight matrix. (f) sparse PCA with high sparsity on weight matrix.

Compressed sensing

- Imagine that, instead of observing the data $\mathbf{x} \in \mathbb{R}^D$, we observe a low-dimensional projection of it, $\mathbf{y} = \boldsymbol{R}\mathbf{x} + \epsilon$ where $\mathbf{y} \in \mathbb{R}^M$, $\boldsymbol{R}$ is a $M \times D$ matrix, $M \ll D$, and ϵ is a noise term(usually Gaussian).
- We assume $\boldsymbol{R}$ is a known sensing matrix, corresponding to different linear projections of $\mathbf{x}$.

Compressed Sensing

- Our goal is to infer $p(\mathbf{x}|\boldsymbol{y}, \boldsymbol{R})$. How can we hope to recover all of **x** if we do not measure all of **x**? The answer is: we can use Bayesian inference with an appropriate prior, that exploits the fact that natural signals can be expressed as a weighted combination of a small number of suitably chosen basis functions.
- That is, we assume $\mathbf{x} = \boldsymbol{W}\boldsymbol{z}$,where $\boldsymbol{z}$ has a sparse prior, and $\boldsymbol{W}$ is suitable dictionary. This is called **compressed sensing or compressive sensing.**
- For CS to work, it is important to represent the signal in the right basis, otherwise it will not be sparse. In traditional CS applications, the dictionary is fixed to be a standard form, such as wavelets.

Image inpainting and denoising

- Suppose we have an image which is corrupted in some way, e.g., by having text or scratches sparsely superimposed on top of it, as in Figure 13.23.
- We might want to estimate the underlying "clean" image. This is called **image inpainting.**

Figure 13.23 An example of image inpainting using sparse coding. Left: original image. Right: reconstruction.

Image inpainting and denoising

- We can model this as a special kind of compressed sensing problem.
- The basic idea is as follows. We partition the image into overlapping patches, $\boldsymbol{y_i}$, and concatenate them to form $\boldsymbol{y}$.
- We define $\boldsymbol{R}$ so that the i'th row selects out patch i.
- Now define $\boldsymbol{V}$ to be the visible (uncorrupted) components of $\boldsymbol{y}$, and $\mathcal{H}$ to be the hidden components.
- To perform image inpainting, we just compute $p(\boldsymbol{y}_{\mathcal{H}}|\boldsymbol{y}_{\boldsymbol{V}}, \theta)$,where θ are the model parameters, which specify the dictionary **W** and the sparsity level λ of $\boldsymbol{z}$.
- We can either learn a dictionary offline from a database of images, or we can learn a dictionary just for this image, based on the non-corrupted patches.