CEE 616: Probabilistic Machine Learning

M5 Unsupervised Learning:
5B: Factor Analysis and Autoencoders

Jimi Oke

UMass Amherst
College of Engineering

Dec 4, 2025

Outline

1. Factor analysis
2. FA Estimation
3. Autoencoders
4. AE variants
5. Outlook

Factor analysis model

- Basic idea: there are latent (hidden) **common factors** $\boldsymbol{z}$ underlying some multivariate observations $\boldsymbol{x}_n \in \mathbb{R}^D$

Factor analysis (FA) is a latent variable generative model specified as:

$$
\begin{aligned}
p(\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) &(1)\\
p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi}) &(2)
\end{aligned}
$$

where:

- $\boldsymbol{z}$: latent vector
- $\boldsymbol{W}$: factor loading matrix, $D \times L$
- $\boldsymbol{\Psi}$: diagonal covariance matrix, $D \times D$

Induced marginal distribution

$$
\begin{aligned}
p(\boldsymbol{x}|\boldsymbol{\theta}) &= \int \mathcal{N}(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi})\mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)d\boldsymbol{z} & (3)\\
&= \mathcal{N}(\boldsymbol{x}|\underbrace{\boldsymbol{W}\boldsymbol{\mu}_0 + \boldsymbol{\mu}}_{\text{mean}}, \underbrace{\boldsymbol{\Psi} + \boldsymbol{W}\boldsymbol{\Sigma}_0\boldsymbol{W}^\top}_{\text{variance}}) & (4)
\end{aligned}
$$

Simplifications:

- $\boldsymbol{\mu}_0 \to \boldsymbol{0}$
- $\boldsymbol{\Sigma}_0 \to \boldsymbol{I}$

The simplified marginal distribution then becomes:

$$
p(\boldsymbol{x}|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^\top + \boldsymbol{\Psi}) \tag{5}
$$

Generative model simplified

$$
\begin{aligned}
\text{[Prior]} \quad p(\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{z}|\mathbf{0}, \boldsymbol{I}) && (6) \\
\text{[Likelihood]} \quad p(\boldsymbol{x}|\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi}) && (7) \\
\text{[Evidence/marginal]} \quad p(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^\top + \boldsymbol{\Psi}) && (8)
\end{aligned}
$$

What FA does

It approximates the covariance matrix of the visible/observed vector $\boldsymbol{x}$ using a low-rank decomposition:

$$\boldsymbol{C} = \mathrm{Cov}[\boldsymbol{x}] = \underbrace{\boldsymbol{W}\boldsymbol{W}^\top + \boldsymbol{\Psi}}_{\text{low-rank decomp}} \tag{9}$$

- $\boldsymbol{W}\boldsymbol{W}^\top$ is $D \times D$ (recall: $\boldsymbol{W} \in \mathbb{R}^{D\times L}$)
- $\boldsymbol{\Psi}$ is $D \times D$ (restricted to be diagonal)
- For each variable x_d, the **marginal variance** (each diagonal term in $\boldsymbol{C}$) is given by:

$$\mathbb{V}[x_d] = \sum_{k=1}^{L} \underbrace{w_{dk}^2}_{\text{common}} + \underbrace{\psi_d}_{\text{unique}} \tag{10}$$

Parameters to be estimated

The unknown parameters in FA are:

- $\boldsymbol{W}$: factor loading matrix
- $\boldsymbol{\Psi}$: covariance matrix

These can be estimated via:

- Maximum likelihood estimation (MLE)
- Expectation-maximization (EM) algorithm
- Bayesian methods (e.g., variational inference, MCMC)

Once estimated, the **posterior** of latent embeddings is given by:

$$p(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{W}^\top \boldsymbol{C}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}), \boldsymbol{I} - \boldsymbol{W}^\top \boldsymbol{C}^{-1}\boldsymbol{W}) \tag{11}$$

- Posterior has closed-form solution under Gaussian distribution

FA model summary

$$
\begin{aligned}
\text{[Prior]} \quad p(\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I}) && (12)\\
\text{[Likelihood]} \quad p(\boldsymbol{x}|\boldsymbol{z}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi}) && (13)\\
\text{[Evidence/marginal]} \quad p(\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{C}) && (14)\\
\text{[Posterior]} \quad p(\boldsymbol{z}|\boldsymbol{x}) &= \mathcal{N}(\boldsymbol{z}|\boldsymbol{W}^\top \boldsymbol{C}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}), \boldsymbol{I} - \boldsymbol{W}^\top \boldsymbol{C}^{-1}\boldsymbol{W}) && (15)
\end{aligned}
$$

- $\boldsymbol{z}$: latent vector, length L (assumed to be zero-mean, unit variance)
- $\boldsymbol{x}$: observed vector
- $\boldsymbol{W}$: $D \times L$ factor loading matrix
- $\boldsymbol{\Psi}$: $D \times D$ diagonal covariance matrix or **matrix of unique variances**
- $\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^\top + \boldsymbol{\Psi}$

Unidentifiability of FA parameters

The parameters $\boldsymbol{W}$ and $\boldsymbol{\Psi}$ are unidentifiable. This can be addressed via:

- Constraining $\boldsymbol{W}$ to have orthonormal columns [PCA]
- Constraining $\boldsymbol{W}$ to be lower triangular
- Informative rotation: $\tilde{\boldsymbol{W}} = \boldsymbol{W}\boldsymbol{R}$, where $\boldsymbol{R}$ is the rotation matrix
  - Commonly used rotations: Varimax, Promax, Oblimin, Geomin, Thurstone, Equamax
- Sparsity-promoting priors on $\boldsymbol{W}$
- Non-Gaussian priors for latent factors

PCA as a special case of FA

Principal components analysis (PCA) is a special case of FA with:

$$\mathbf{\Psi} = \sigma^2 \boldsymbol{I} \tag{16}$$

where σ^2 is the isotropic noise variance. In PCA, the covariance of the observed vector is given by:

$$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^\top + \sigma^2 \boldsymbol{I} \tag{17}$$

Autoencoders as nonlinear PCA/FA

In PCA/FA, we learn a **linear mapping** from a high-dimensional observed space $\boldsymbol{x} \in \mathbb{R}^D$ to a low-dimensional latent space $\boldsymbol{z} \in \mathbb{R}^L$ and vice-versa.

- **Encoder** f_e: mapping from $\boldsymbol{x} \to \boldsymbol{z}$
- **Decoder** f_d: mapping from $\boldsymbol{z} \to \boldsymbol{x}$

In PCA, for example, f_e is given by:

$$\boldsymbol{z} = \boldsymbol{W}^\top \boldsymbol{x} \equiv f_e(\boldsymbol{x}) \tag{18}$$

And f_d is given by:

$$\hat{\boldsymbol{x}} = \boldsymbol{W}\boldsymbol{z} \equiv f_d(\boldsymbol{z}) \tag{19}$$

To introduce flexibility, we can specify f_e and f_e are nonlinear/more complex functions. This is best accomplished via neural network, resulting in an **autoencoder**.

Reconstruction loss

The reconstruction function is the approximation of the observation from the decoder:

$$\hat{\boldsymbol{x}} \equiv r(\boldsymbol{x}) = f_d(f_e(\boldsymbol{x})) \tag{20}$$

An autoencoder is thus trained to minimize the reconstruction loss

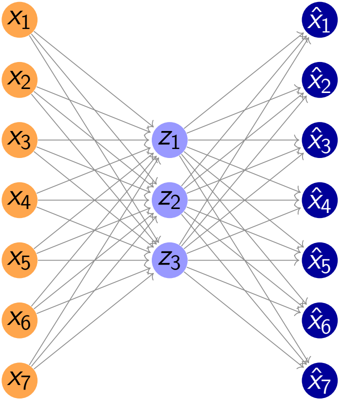
$$\mathcal{L}(\boldsymbol{\theta}) = ||r(\boldsymbol{x}) - \boldsymbol{x}||_2^2 \tag{21}$$

or equivalently, the negative log-likelihood:

$$\mathcal{L}(\boldsymbol{\theta}) = -\log p(\boldsymbol{x}|r(\boldsymbol{x})) \tag{22}$$

Basic autoencoder (AE) architecture

Autoencoder with 2 single-layer MLPS: input layer, hidden layer (latent representation) and output layer (reconstruction)

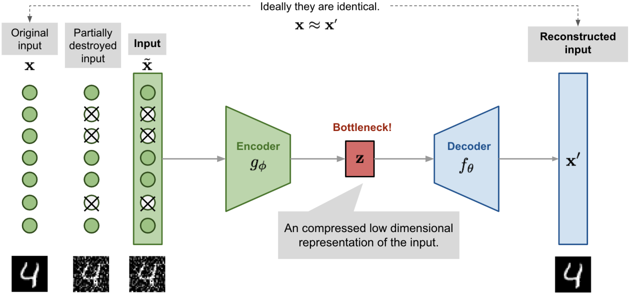


- Hidden layer (size L) is a low-dimensional **bottleneck** between input and reconstruction
- $L \ll D$: undercomplete representation
- $L \gg D$: overcomplete representation (regularize to prevent identity learning)

Denoising autoencoders

In denoising autoencoders (DAEs), the input is corrupted ($\tilde{\boldsymbol{x}}$) by:

- Gaussian noise: $p_c(\tilde{\boldsymbol{x}}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{x}}|\boldsymbol{x}, \sigma^2 \boldsymbol{I})$
- Bernoulli dropout: randomly setting a proportion of input nodes to zero



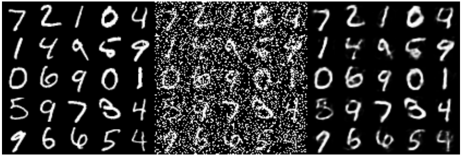
Schematic of a DAE.

Source: https://lilianweng.github.io/posts/2018-08-12-vae/

The model is then trained to minimize the loss between the reconstructed input $r(\tilde{\boldsymbol{x}})$ and its uncorrupted version $\boldsymbol{x}$

Uses of DAE

- DAEs are used for denoising images



Original, corrupted and reconstructed images from MNIST dataset.

Source: http://www.opendeep.org/v0.0.5/docs/tutorial-your-first-model

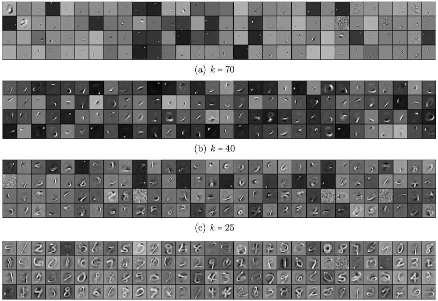
- They can also learn vector fields of input data

Sparse autoencoder (SAE)

Sparse autoencoder (SAE): sparsity penalty on latent activations

$$\Omega(\boldsymbol{z}) = \lambda ||\boldsymbol{z}||_1 \tag{23}$$

- *k*-Sparse autoencoder: use only k largest activations in training



(a) $k = 70$

(b) $k = 40$

(c) $k = 25$

(d) $k = 10$

Filters of the k -sparse autoencoder for different sparsity levels k, learnt from MNIST with 1000 hidden units.

Source: https://arxiv.org/pdf/1312.5663.pdf

Other AEs

- **Contractive autoencoder** (CAE): regularizes via penalty on reconstruction loss

$$\Omega(\boldsymbol{z}, \boldsymbol{x}) = \lambda \left\| \frac{\partial f_e(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\|_F^2 \tag{24}$$

- Variational autoencoder (VAE): probablistic version of AE/generative model

Reading

- **PMLI** 20.3
- **DL** 20