From PCA to Autoencoders

Unsupervised Representation Learning

Florent Forest

forest@lipn.univ-paris13.fr
http://florentfo.rest
FlorentF9

17 décembre 2019

Institut Supérieur de l'Aéronautique et de l'Espace
SUPAERO

Table of contents

Introduction to autoencoders

Motivations

Do better than linear dimensionality techniques (PCA) Using neural networks. Handle very-high-dimensional data (i.e. d = 1k-1M) Scale linearly with the size of data. [4]

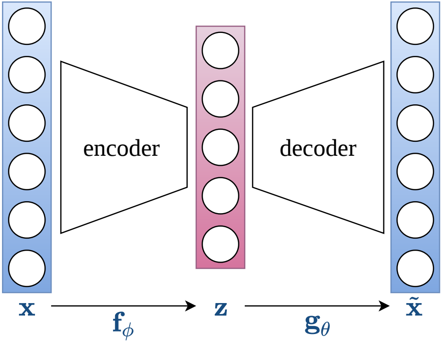
Introduction to autoencoders

Definition

Definition

An autoencoder is a neural network trained to reconstruct its inputs. It is composed of two parts:

1. an encoder, mapping the input to a latent representation ("code") $\mathbf{z} = \mathbf{f}_{\boldsymbol{\phi}}(\mathbf{x})$
2. a decoder, mapping the code back to the input space $\tilde{\mathbf{x}} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z})$



Definition

Challenge

We do not want the encoder to learn the identity function, but to learn a *good representation* of our data.

Regularization?

Reducing the size of the *hypothesis set* $\mathcal{H}$ by constraining the space of possible solutions to the optimization problem.

- L2 weight decay
- Sparsity, L1 weight decay
- ...

Introduction to autoencoders

Mathematical formulation

What is a good representation?

Let $q_{\boldsymbol{\phi}}(Z|X)$ be a (stochastic) parametric mapping from X to Z. A good representation Z of a random variable X maximizes mutual information between X and Z (*infomax principle*):

$$\begin{aligned}\mathbb{I}(X;Z) &= \mathbb{H}(X) - \mathbb{H}(X|Z)\\ &= C(X) + \mathbb{E}_{q_{\boldsymbol{\phi}}(X,Z)}\left[\log q_{\boldsymbol{\phi}}(X|Z)\right]\end{aligned}$$

For any parametric distribution $p_{\boldsymbol{\theta}}(X|Z)$ we have $\mathbb{E}_{q_{\boldsymbol{\phi}}(X,Z)}\left[\log p_{\boldsymbol{\theta}}(X|Z)\right] \leq \mathbb{E}_{q_{\boldsymbol{\phi}}(X,Z)}\left[\log q_{\boldsymbol{\phi}}(X|Z)\right]$ (using $D_{KL}(q||p) \geq 0$).

Task: maximize a lower bound on $\mathbb{I}(X;Z)$

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{maximize}}\ \mathbb{E}_{q_{\boldsymbol{\phi}}(X,Z)}\left[\log p_{\boldsymbol{\theta}}(X|Z)\right]$$

Loss function for deterministic autoencoders

We consider deterministic mappings $Z = \mathbf{f}_{\boldsymbol{\phi}}(X)$ (or $q_{\boldsymbol{\phi}}(Z|X) = \delta(Z - \mathbf{f}_{\boldsymbol{\phi}}(X))$) and $\tilde{X} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{f}_{\boldsymbol{\phi}}(X))$.

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{maximize}}\ \mathbb{E}_{q_{\boldsymbol{\phi}}(X)}\left[\log p_{\boldsymbol{\theta}}(X|Z = \mathbf{f}_{\boldsymbol{\phi}}(X))\right]$$

Using empirical mean over a set of i.i.d. data samples:

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{maximize}}\ \sum_i \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i)} = \mathbf{f}_{\boldsymbol{\phi}}(\mathbf{x}^{(i)}))$$

equivalent to:

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{maximize}}\ \sum_i \log p(\mathbf{x}^{(i)}|\tilde{\mathbf{x}}^{(i)} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{f}_{\boldsymbol{\phi}}(\mathbf{x}^{(i)})))$$

Let us turn this into a minimization the negative sum of individual loss functions $\mathcal{L}(\mathbf{x}|\tilde{\mathbf{x}}) = -\log p(\mathbf{x}|\tilde{\mathbf{x}})$.

Loss function for deterministic autoencoders

The reconstruction $\tilde{\mathbf{x}}$ is the mean of a distribution that may have generated $\mathbf{x}$.

Continuous variables: $\mathbf{x} \in \mathbb{R}^d$

- Gaussian distribution: $X|\tilde{X} = \tilde{\mathbf{x}} \sim \mathcal{N}(\tilde{\mathbf{x}}, \sigma^2\mathbf{I})$
- Loss function: $\mathcal{L}(\mathbf{x}|\tilde{\mathbf{x}}) \propto ||\mathbf{x} - \tilde{\mathbf{x}}||_2^2$

$\rightarrow$ Mean Squared Error (MSE) loss

Binary variables: $\mathbf{x} \in \{0,1\}^d$, or $\mathbf{x} \in [0,1]^d$

- Bernoulli distribution: $X|\tilde{X} = \tilde{\mathbf{x}} \sim \mathcal{B}(\tilde{\mathbf{x}})$
- Loss function: $-\sum_{j=1}^{d} [\mathbf{x}_j \log \tilde{\mathbf{x}}_j + (1 - \mathbf{x}_j) \log(1 - \tilde{\mathbf{x}}_j)]$

$\rightarrow$ Cross-entropy loss

Neurons can learn principal components

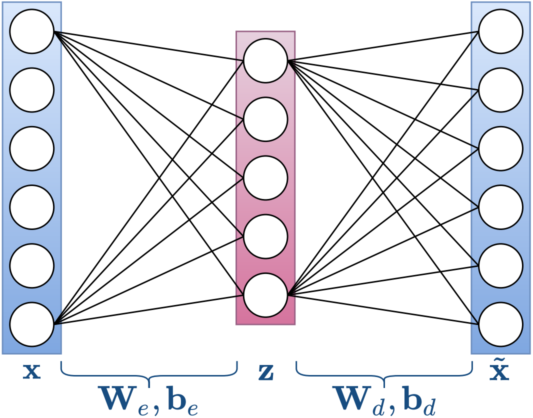
Hebb's Learning Rule

Learning PC with Oja's rule

[1], [7], [8]

Linear autoencoders

Definition



$\mathbf{x}$
$\mathbf{W}_e, \mathbf{b}_e$
$\mathbf{z}$
$\mathbf{W}_d, \mathbf{b}_d$
$\tilde{\mathbf{x}}$

Similarity with PCA

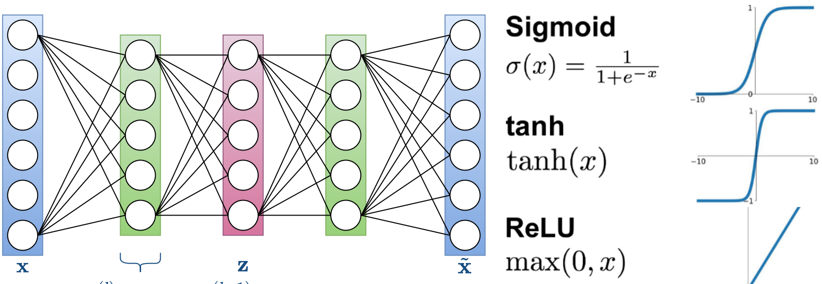
Show equivalence to PCA

[9]

Real-life autoencoders

Real-life autoencoders

Non-linear and deep autoencoders



$$\mathbf{h}^{(l)} = \sigma(\mathbf{W}\mathbf{h}^{(l-1)} + \mathbf{b})$$

Trained end-to-end with backprop. Layer-wise pretraining [4] is in fact not necessary (thanks to ReLU, better optimization and regularization).

- Not equivalent to PCA! (see paper AA-PCA)

Real-life autoencoders

Different types of regularization

Undercomplete or overcomplete

Let's code! (1)

The danger of overfitting

Problem: even a single continuous latent variable can remember the entire training set (one real number per sample).

The latent space is not continuous.

Sparse autoencoders

Denoising autoencoders

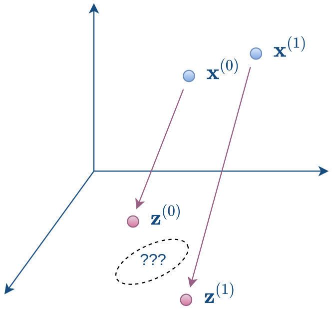
[10]

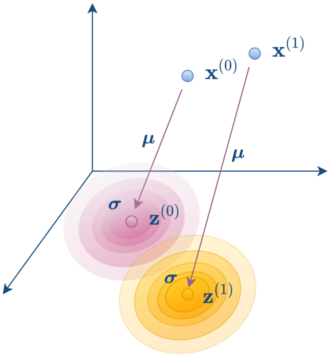
Real-life autoencoders

Variational autoencoders (VAE)

Motivations

Limitation of standard AE: the latent space has *no structure* and may not be continuous; we cannot explore it nor sample from it.





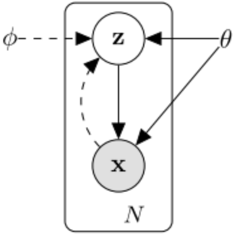
Example latent space figures, examples with MNIST, faces, music styles...

Variational autoencoders (VAE)

Probabilistic setting

Generative model $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$:

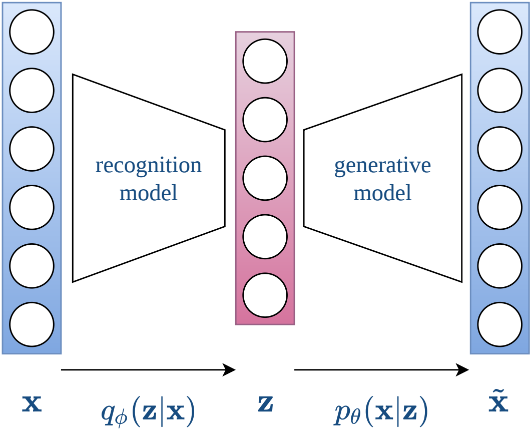
1. $\mathbf{z}$ is sampled from the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$
2. $\mathbf{x}$ is generated with likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ (*probabilistic decoder*)



Problem: $\boldsymbol{\theta}$ is a NN, so $p_{\boldsymbol{\theta}}(\mathbf{x})$ and $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ are intractable.

▶ Stochastic Gradient Variational Bayes (SGVB) [5] is an efficient estimation method in case of intractable marginal likelihood/posterior and large datasets.

Variational autoencoders (VAE)



Recognition model $\rightarrow$ approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ (*probabilistic encoder*)

VAE loss function

VAE ELBO (evidence lower bound)

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{maximize}} \; - D_{KL}\left(q_{\boldsymbol{\phi}}(Z|X)||p_{\boldsymbol{\theta}}(Z)\right) + \mathbb{E}_{q_{\boldsymbol{\phi}}(Z|X)}\left[\log p_{\boldsymbol{\theta}}(X|Z)\right]$$

Key ideas

- The second term is a (negative) reconstruction error (e.g. MSE or cross-entropy) as in a deterministic AE.
- The first term, a Kullback-Leibler divergence between $q_{\boldsymbol{\phi}}(Z|X)$ and $p_{\boldsymbol{\theta}}(Z)$, acts as a regularizer pushing the encoder distribution closer to the prior distribution (typically a gaussian)

VAE loss function

Let's put gaussians everywhere!

- $p_{\boldsymbol{\theta}}(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$
- $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$

The reparameterization trick

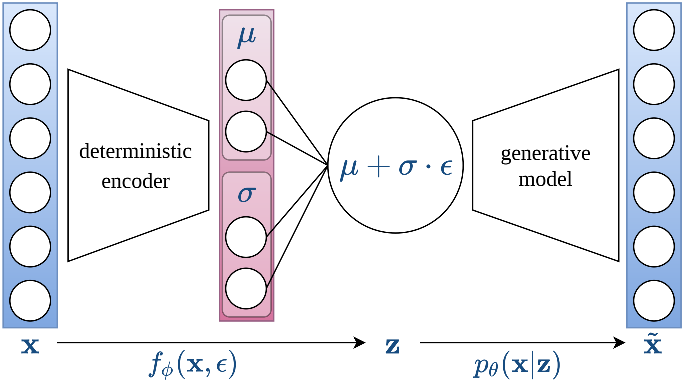
To sample from $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, we use the reparameterization $\mathbf{z} = f_{\boldsymbol{\phi}}(\mathbf{x}, \epsilon) = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

For a given $\mathbf{x}^{(i)}$, and using 1-sample Monte-Carlo estimation, the ELBO becomes:

$$\frac{1}{2}\sum_j \left(1 + \log(\boldsymbol{\sigma}_j^{(i)})^2 - \boldsymbol{\mu}_j^{(i)} - (\boldsymbol{\sigma}_j^{(i)})^2\right) + \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i)})$$

where $\mathbf{z}^{(i)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \cdot \boldsymbol{\epsilon}$

VAE: the reparameterization trick



Let’s code! (2)

Applications

Autoencoders can extract useful low-dimensional representations from high-dimensional data. They can be used as:

- a pre-processing step for any other ML algorithm (clustering, supervised classification or regression)
- an unsupervised pre-training prodecure for supervised deep neural networks (e.g. layer-wise pre-training) $\rightarrow$ See [4], [3], [10].

Data compression

Autoencoders can be used as a compression algorithm, but it is:

- lossy
- data-specific

Not commonly used in practice...

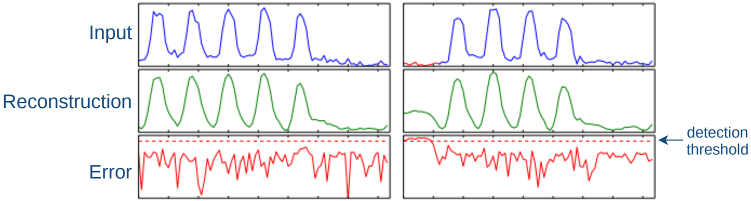
Data augmentation

The decoder model can be used to generate new data samples:

- Deterministic AEs: adding noise, interpolating or extrapolating in latent space [2]
- Generative models (VAE, GAN)

Anomaly detection

When the input differs from the training data distribution (e.g. an outlier), the autoencoder will produce a large reconstruction error. This error can be used to score anomalies [6].



(Malhotra et al, 2016)

S. Becker.
Unsupervised Learning Procedures for Neural Networks.
The International Journal of Neural Systems, 1:17–33, 1991.

T. Devries and G. W. Taylor.
Dataset Augmentation in Feature Space.
In *ICLR 2017 Workshop*, pages 1–12, 2017.

D. Erhan, A. Courville, and P. Vincent.
Why Does Unsupervised Pre-training Help Deep Learning ?
Journal of Machine Learning Research, 11:625–660, 2010.

References ii

G. E. Hinton and R. Salakhutdinov.
Reducing the Dimensionality of Data with Neural Networks.
Science, 313(July):504–507, 2006.

D. P. Kingma and M. Welling.
Stochastic Gradient VB and the Variational Auto-Encoder.
2nd International Conference on Learning Representations (ICLR), pages 1–14, 2014.

P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff.
LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection.
In *ICML 2016 Anomaly Detection Workshop*, 2016.

E. Oja.
A Simplified Neuron Model as a Principal Component Analyzer.
Journal of Mathematical Biology, 15(3):267–273, 1982.

E. Oja.
Principal Components, Minor Components, and Linear Neural Networks, 1992.

E. Plaut.
From Principal Subspaces to Principal Components with Linear Autoencoders.
pages 1–6, 2018.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol.
Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion.
Journal of Machine Learning Research, 11:3371–3408, 2010.