Deep Learning

Chapter 9: Autoencoders

Mina Rezaei

Department of Statistics – LMU Munich

Winter Semester 2020

LECTURE OUTLINE

Autoencoders - Basic Principle

Undercomplete Autoencoders

Principal Component Analysis as Autoencoder

Autoencoders - Basic Principle

AUTOENCODER (AE)-TASK AND STRUCTURE

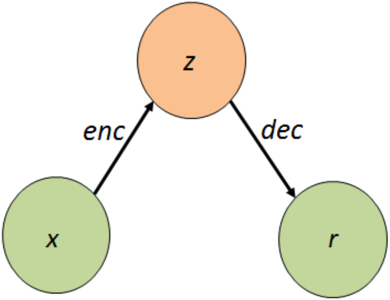
- Autoencoders (AEs) are a special kind of feedforward neural networks.
- Task: Learn a lossy compression of the data
- Autoencoders consist of two parts:
  - **encoder** function $\mathbf{z} = enc(\mathbf{x})$.
  - **decoder** that produces the reconstruction $\hat{\mathbf{x}} = dec(\mathbf{z})$.
- Loss function measures the quality of the reconstruction compared to the input:

$$L\left(\mathbf{x}, dec(enc(\mathbf{x}))\right)$$

- Goal: Learn good **internal representations z** (also called **code**).

AUTOENCODER (AE)- COMPUTATIONAL GRAPH

The general structure of an AE as a computational graph:

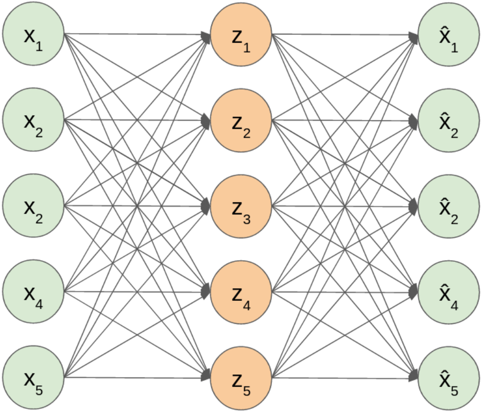


- An AE has two computational steps:
  - the encoder *enc*, mapping **x** to **z**.
  - the decoder *dec*, mapping **z** to **x**.

Undercomplete Autoencoders

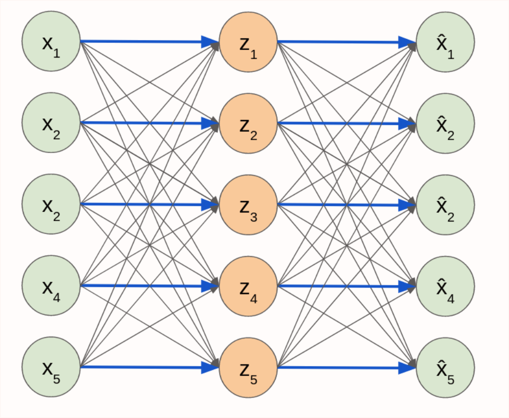
UNDERCOMPLETE AUTOENCODERS

- A naive implementation of an autoencoder would simply learn the identity $dec(enc(\mathbf{x})) = \mathbf{x}$.
- This would not be useful.



UNDERCOMPLETE AUTOENCODERS

- A naive implementation of an autoencoder would simply learn the identity $dec(enc(\mathbf{x})) = \mathbf{x}$.
- This would not be useful.

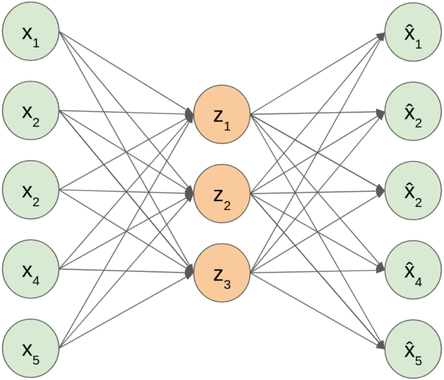


UNDERCOMPLETE AUTOENCODERS

- Therefore we have a “bottleneck” layer: We restrict the architecture, such that

$$\dim(\mathbf{z}) < \dim(\mathbf{x})$$

- Such an AE is called **undercomplete**.

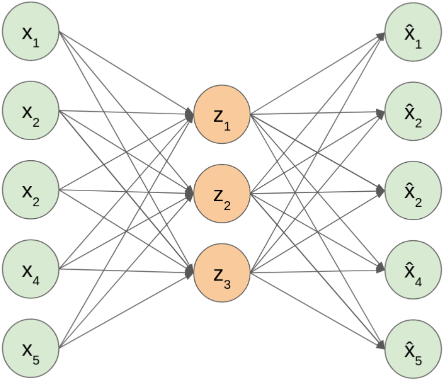


UNDERCOMPLETE AUTOENCODERS

- In other words: In an undercomplete AE, the hidden layer has fewer neurons than the input layer.

$\rightarrow$ That will force the AE to

  - capture only the most salient features of the training data!
  - learn a “compressed” representation of the input.



UNDERCOMPLETE AUTOENCODERS

- Training an AE is done by minimizing the risk, where the loss function penalizes the reconstruction $dec(enc(\mathbf{x}))$ for differing from $\mathbf{x}$.
- The L2-loss

$$\|\mathbf{x} - dec(enc(\mathbf{x}))\|_2^2$$

  is a typical choice, but other loss functions are possible as well.
- For optimization, the very same optimization techniques as for standard feed-forward nets are applied (SGD, RMSProp, ADAM,...).

EXPERIMENT: LEARN TO ENCODE MNIST

- Let us try to compress the MNIST data as good as possible.
- Therefore, we will fit a simple undercomplete autoencoder to learn the best possible representation
- We fit the autoencoder for different dimensions of the internal representation **z** (different “bottleneck” sizes).

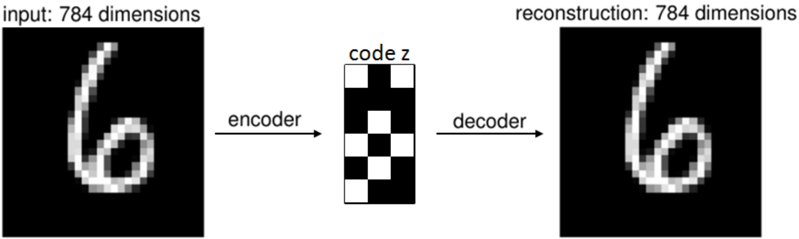


Figure: Flow chart of our our autoencoder: reconstruct the input with fixed dimensions $dim(\mathbf{z}) \ll dim(\mathbf{x})$.

EXPERIMENT: LEARN TO ENCODE MNIST

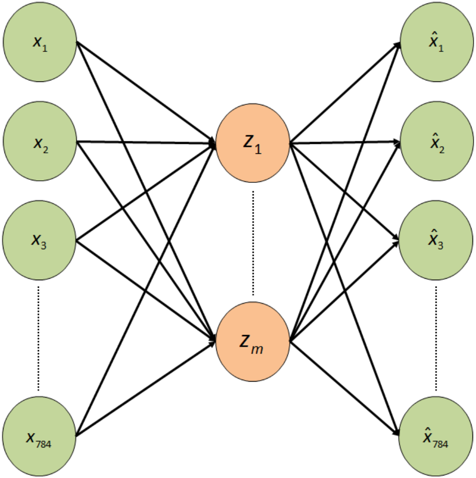


Figure: Architecture of the autoencoder.

EXPERIMENT: LEARN TO ENCODE MNIST

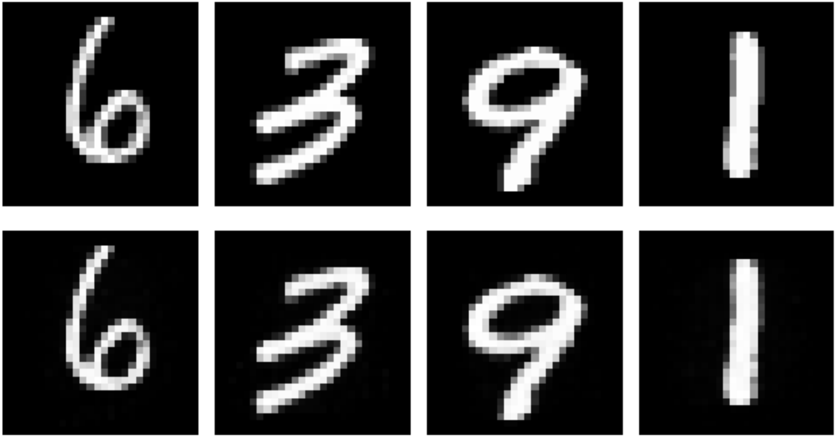


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 784 = dim(\mathbf{x})$.

EXPERIMENT: LEARN TO ENCODE MNIST

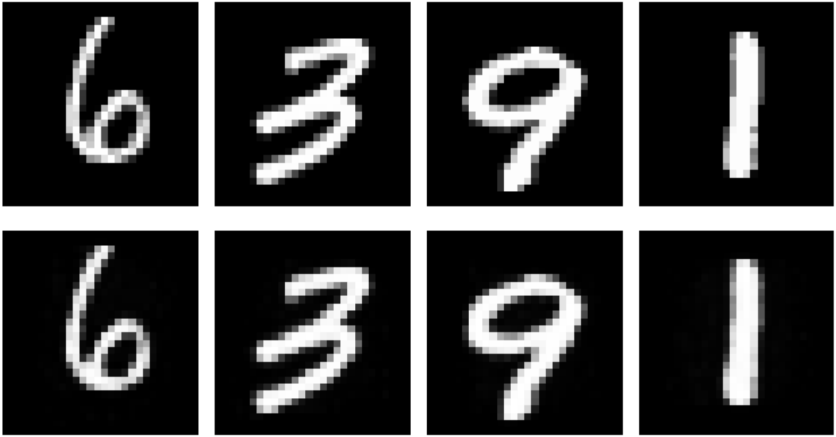


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 256.$

EXPERIMENT: LEARN TO ENCODE MNIST

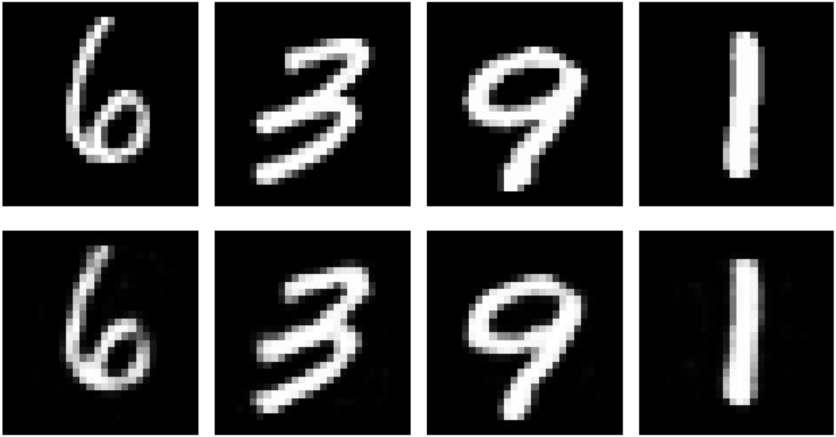


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 64$.

EXPERIMENT: LEARN TO ENCODE MNIST

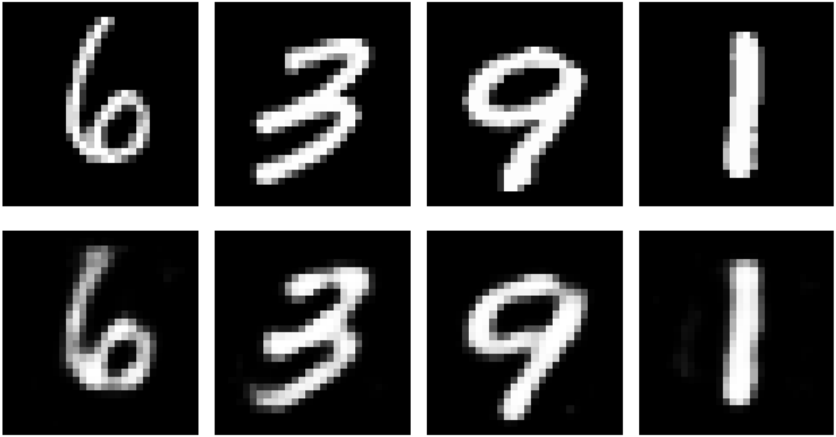


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 32$.

EXPERIMENT: LEARN TO ENCODE MNIST

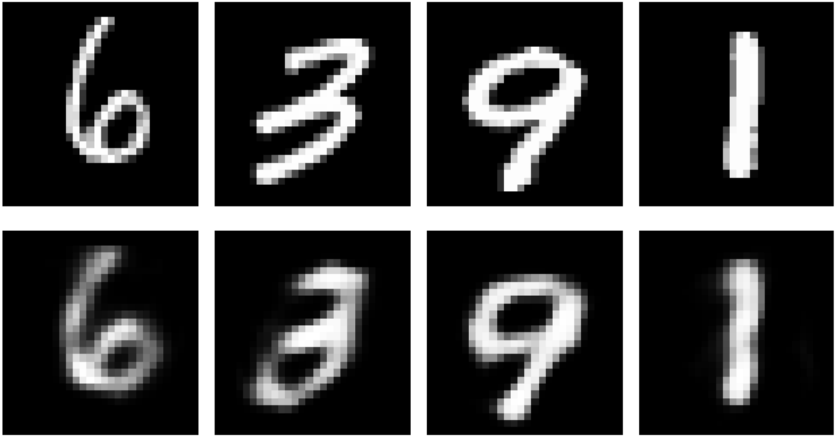


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 16$.

EXPERIMENT: LEARN TO ENCODE MNIST

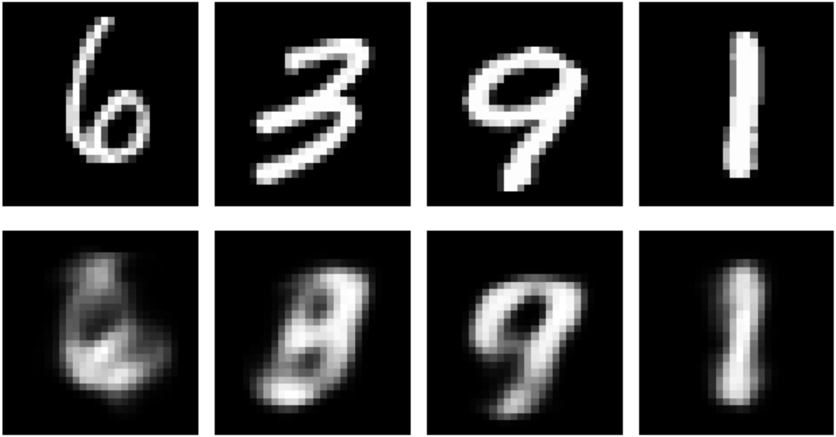


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 8$.

EXPERIMENT: LEARN TO ENCODE MNIST

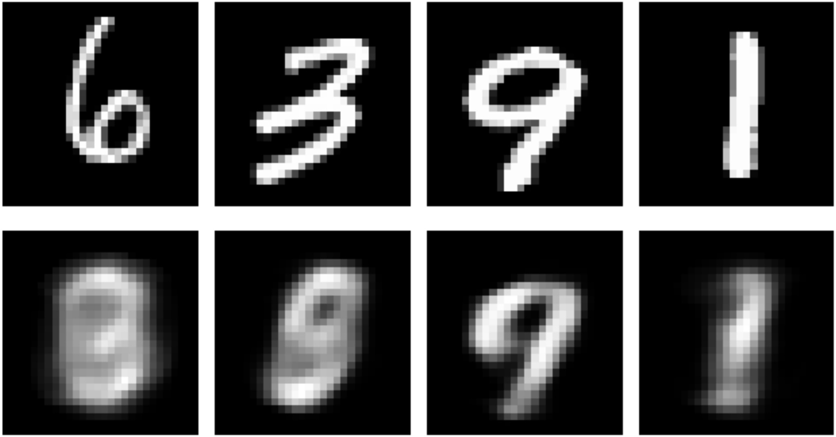


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 4$.

EXPERIMENT: LEARN TO ENCODE MNIST

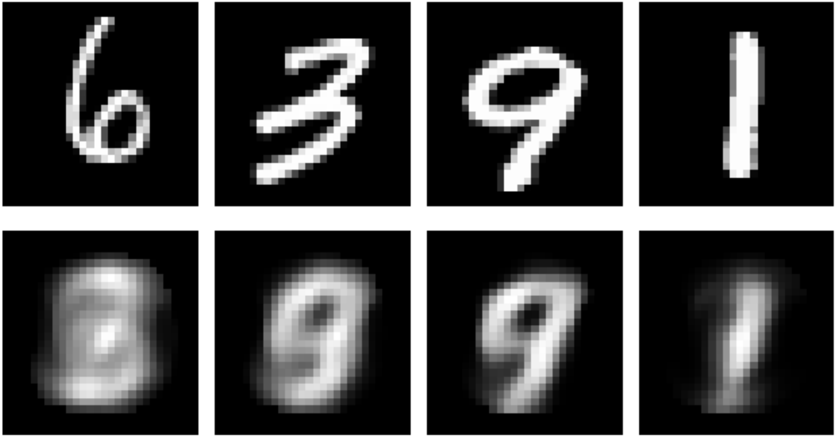


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 2$.

EXPERIMENT: LEARN TO ENCODE MNIST

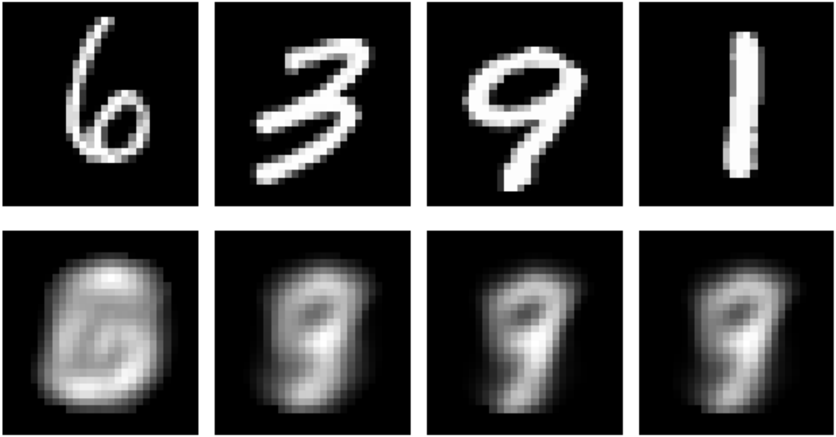
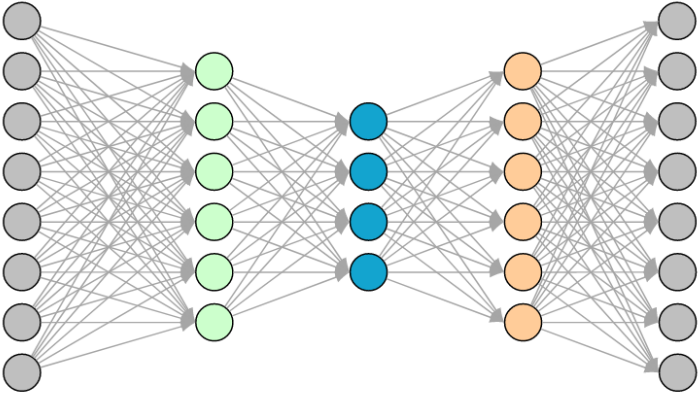


Figure: The top row shows the original digits, the bottom row the reconstructed ones.

- $dim(\mathbf{z}) = 1$.

INCREASING THE CAPACTIY OF AES

Increasing the number of layers adds capacity to autoencoders:

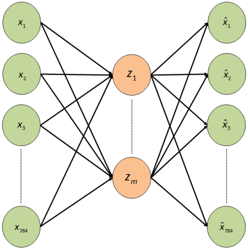


Principal Component Analysis as Autoencoder

PRINCIPAL COMPONENT ANALYSIS

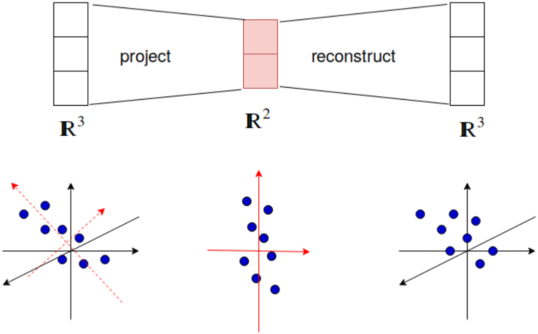
- Consider the same simple undercomplete autoencoder architecture as above, but this time with
  - **linear** encoder function $enc(\mathbf{x})$, and
  - **linear** decoder function $dec(\boldsymbol{z})$.

  Further we use the L2-loss $\|\mathbf{x} - dec(enc(\mathbf{x}))\|_2^2$ and assume that inputs are normalized to zero mean.
- In other words: We want to find the **linear projection** of the data with the minimal L2-reconstruction error.



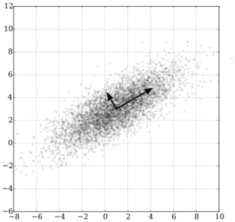
PRINCIPAL COMPONENT ANALYSIS

- It can be shown that, given a $\dim(\boldsymbol{z}) = k$, the optimal solution is an **orthogonal** linear transformation (i.e. a rotation of the coordinate system) given by the k singular vectors with largest singular values.



PRINCIPAL COMPONENT ANALYSIS

- This is an equivalent formulation to **Principal Component Analysis (PCA)**, which uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called **principal components**.
- The transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible)



PRINCIPAL COMPONENT ANALYSIS

- The formulations are equivalent: “Find a linear projection into a *k*-dimensional space that ...”
  - “... minimizes the L2-reconstruction error” (AE-based formulation)
  - “... maximizes the variance of the projected datapoints” (statistical formulation).

PRINCIPAL COMPONENT ANALYSIS

- An AE with a non-linear decoder/encoder can be seen as a non-linear generalization of PCA.

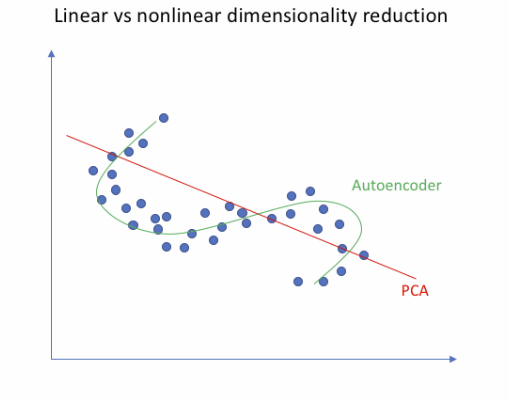


Figure: Credits: Jeremy Jordan “Introduction to autoencoders”

REFERENCES

Ian Goodfellow, Yoshua Bengio and Aaron Courville (2016)
Deep Learning
http://www.deeplearningbook.org/