Neural networks and Deep learning

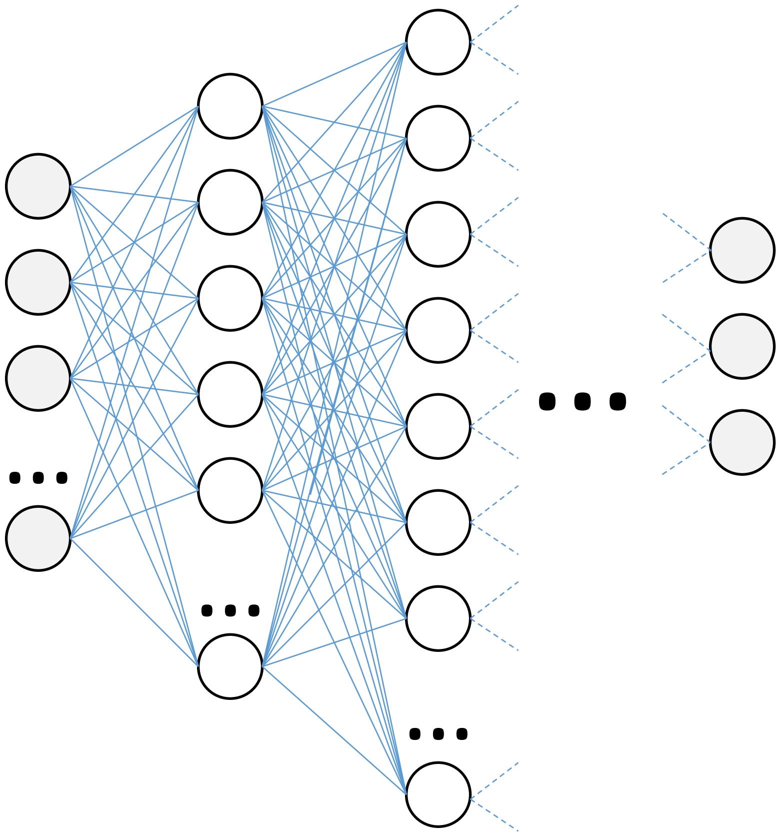
Alberto TONDA

UMR 518 MIA-PS, INRAE, AgroParisTech, Université Paris-Saclay
UAR 3611, Institut des Systèmes Complexes de Paris Île-de-France

Outline

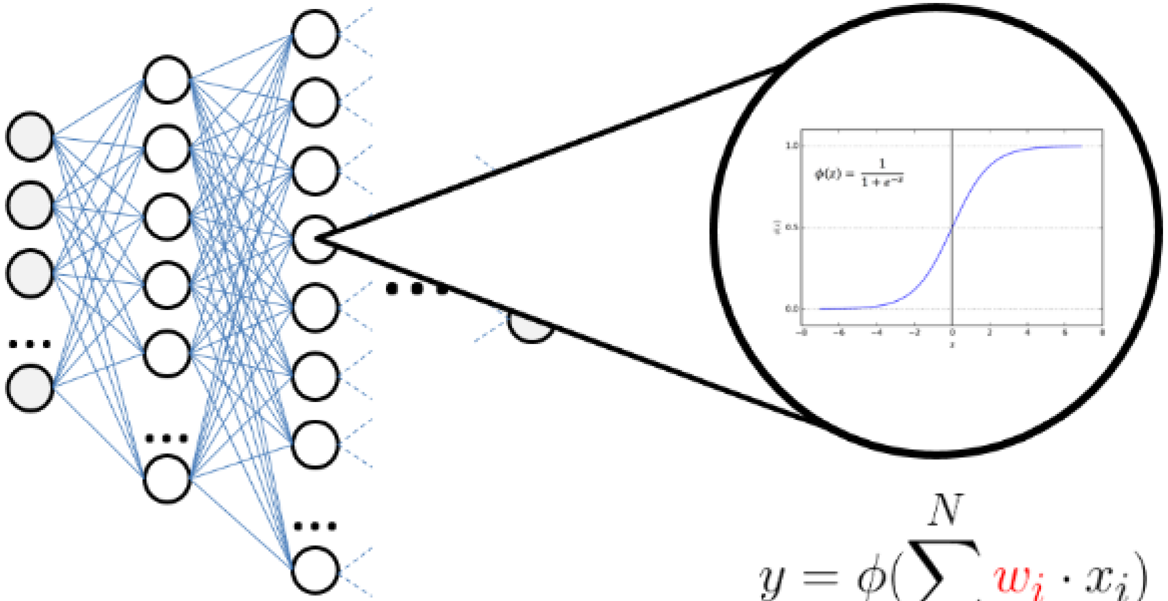
- Artificial neural networks
- Optimizing a neural network
- Overparametrization (“double descent”)
- Convolutional neural networks
- Recurrent neural networks
- Transformers
- Autoencoders
- From the point of view of optimization…

Artificial neural networks



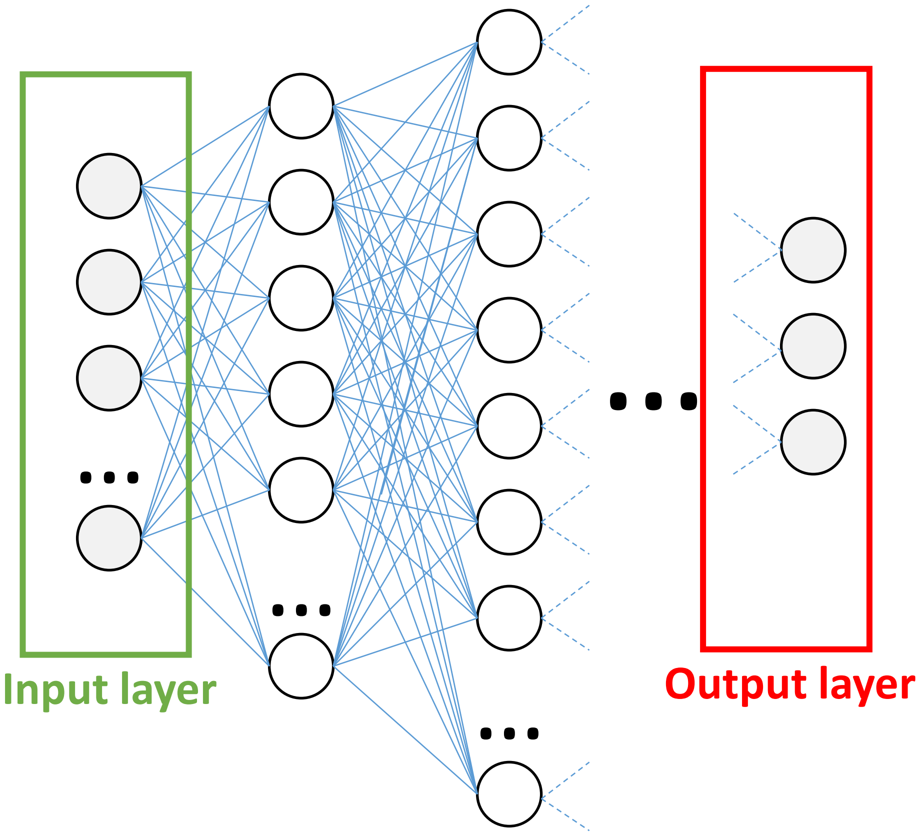
Artificial neural networks

- Based on an old (and wrong) model of a real neuron

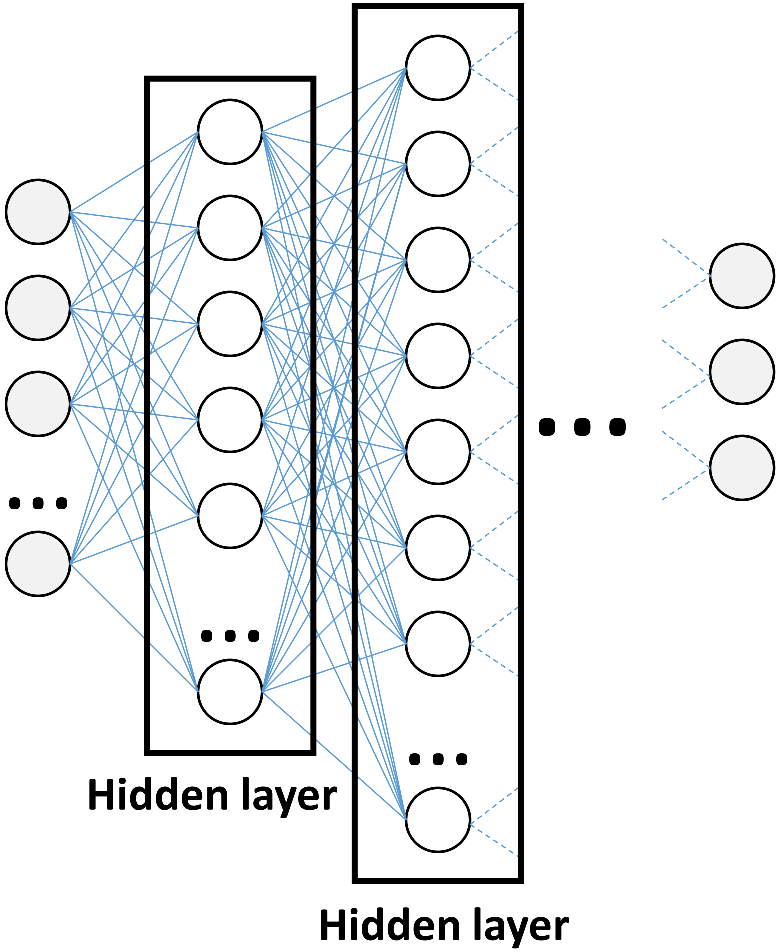


$$y = \phi\left(\sum_{i}^{N} w_i \cdot x_i\right)$$

Artificial neural networks



Artificial neural networks



Artificial neural network

- So, an ANN is just a *very complex function*
- In ML terms, its parameters are the weights
- Modern ANNs have thousands/millions/**billions** of weights!
- How to optimize a function in such high dimension?

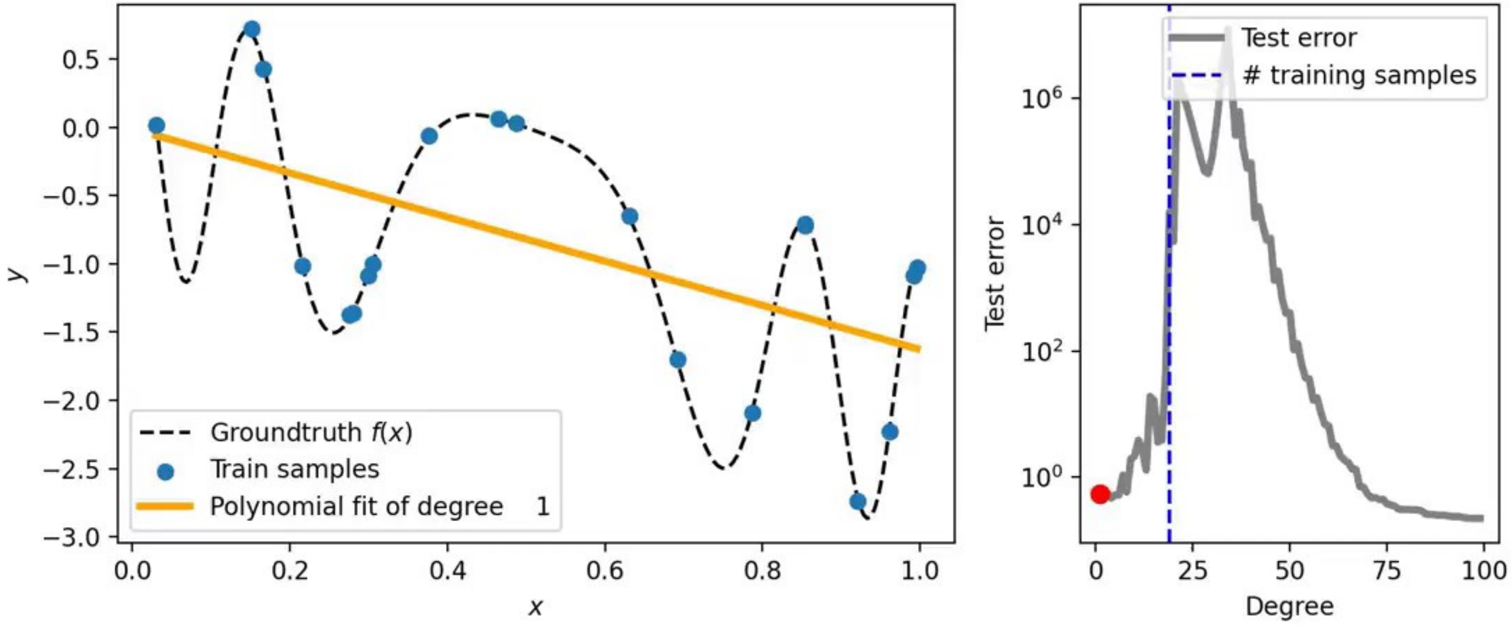
SGD and backpropagation

- Objective function is called **loss function**
  - To be **minimized**; exact content depends on the task
  - MSE for regression, categorical cross-entropy for classification

- It is possible to compute the derivative of the loss
  - Thanks to chain rule and backpropagation
  - Partial derivative of loss with respect to **each weight/parameter**
  - Stochastic Gradient Descent (and successors)
  - “Stochastic” = use only a subset of samples at each iteration

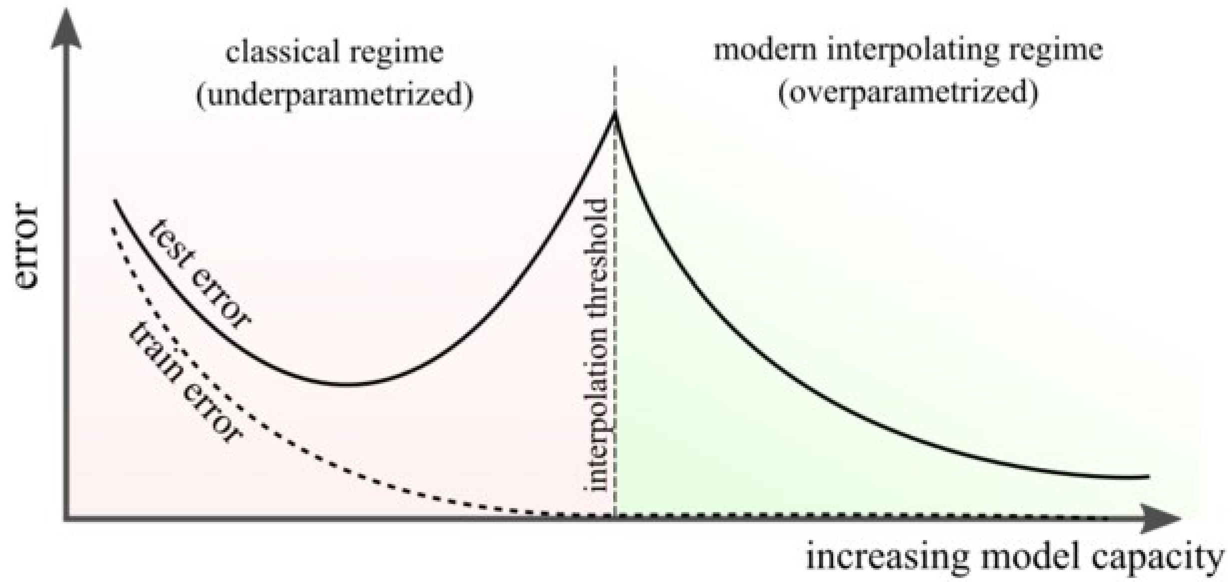
Overparametrization

- Wait a second!
  - From ML, we learned that having too many parameters is **bad**!
  - Models with too many parameters tend to **overfit** terribly…right?



Overparametrization

- What is happening here? Well, we don’t really know
  - Empirical results, **overparametrizing** improves **generalization**
  - “Double descent” or “W figure”
  - Does **regularization** play a role?

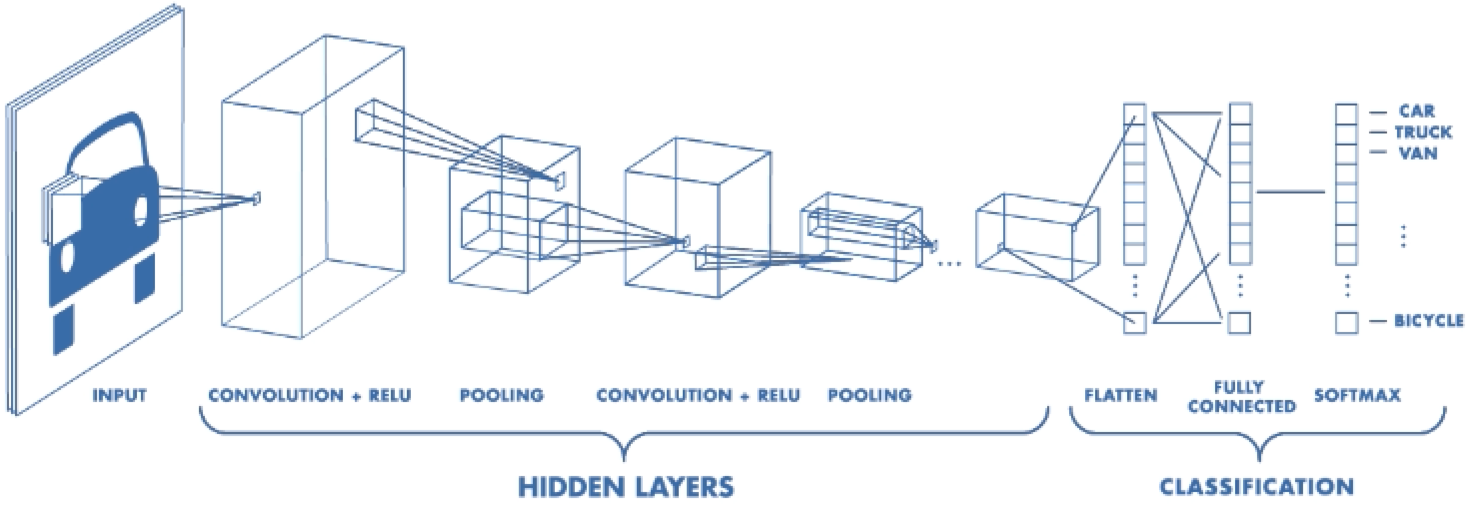


Why are neural networks so successful?

- Extremely performant for **relational data**
  - Thanks to recent architectures (CNNs, RNNs, Transformers, …)
  - Images, text, sound, …
  - Other ML approaches are just not as good
- Graphics Processing Units (GPUs)
  - Are really good at performing parallel computations
  - And in fact, they are excellent to speed up gradient computation
- Multiple outputs can be interpreted as pixels/audio/text…
- More and more data is available!

Convolutional neural networks

- Specialized layers that scan the whole structured input
- They can find patterns in every part of it
- Example: images (square window, slide over pixels)

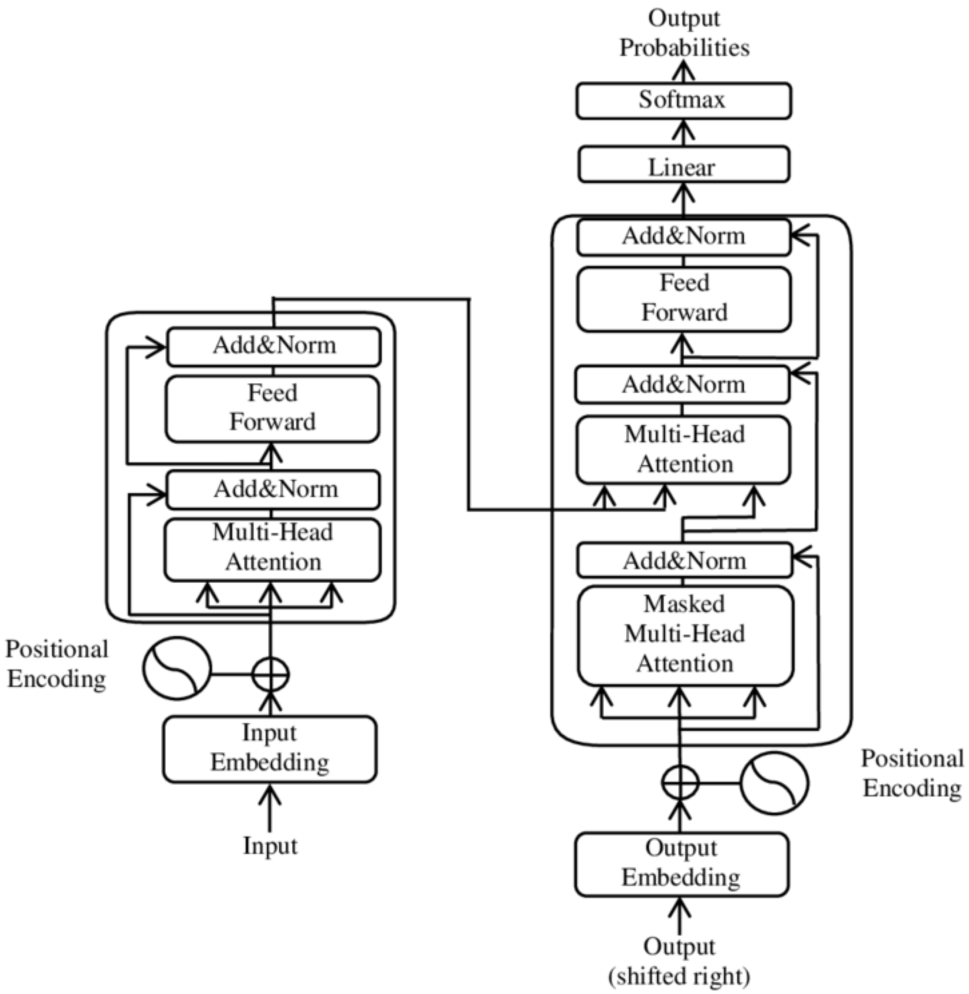


Recurrent neural networks

- Current output is not just $y_n = f(\boldsymbol{x}_n)$
  - Instead, there is a **state** of the system $y_n = f(\boldsymbol{x}_n, S)$
  - $S = g(\boldsymbol{x_o}, \boldsymbol{x_1}, \dots, \boldsymbol{x_{n-1}})$
  - Sequences, time series, dynamic systems…
- RNNs use specialized layers to **keep a memory** of the state
- Modern developments
  - Long-Short Memory Networks (LSTMs)
  - Gated Recurrent Units (GRUs)

Transformers

- De-facto replaced RNNs
- Gets all the data at once
- Specialized layers that try to capture relationships between parts of the input



Autoencoders

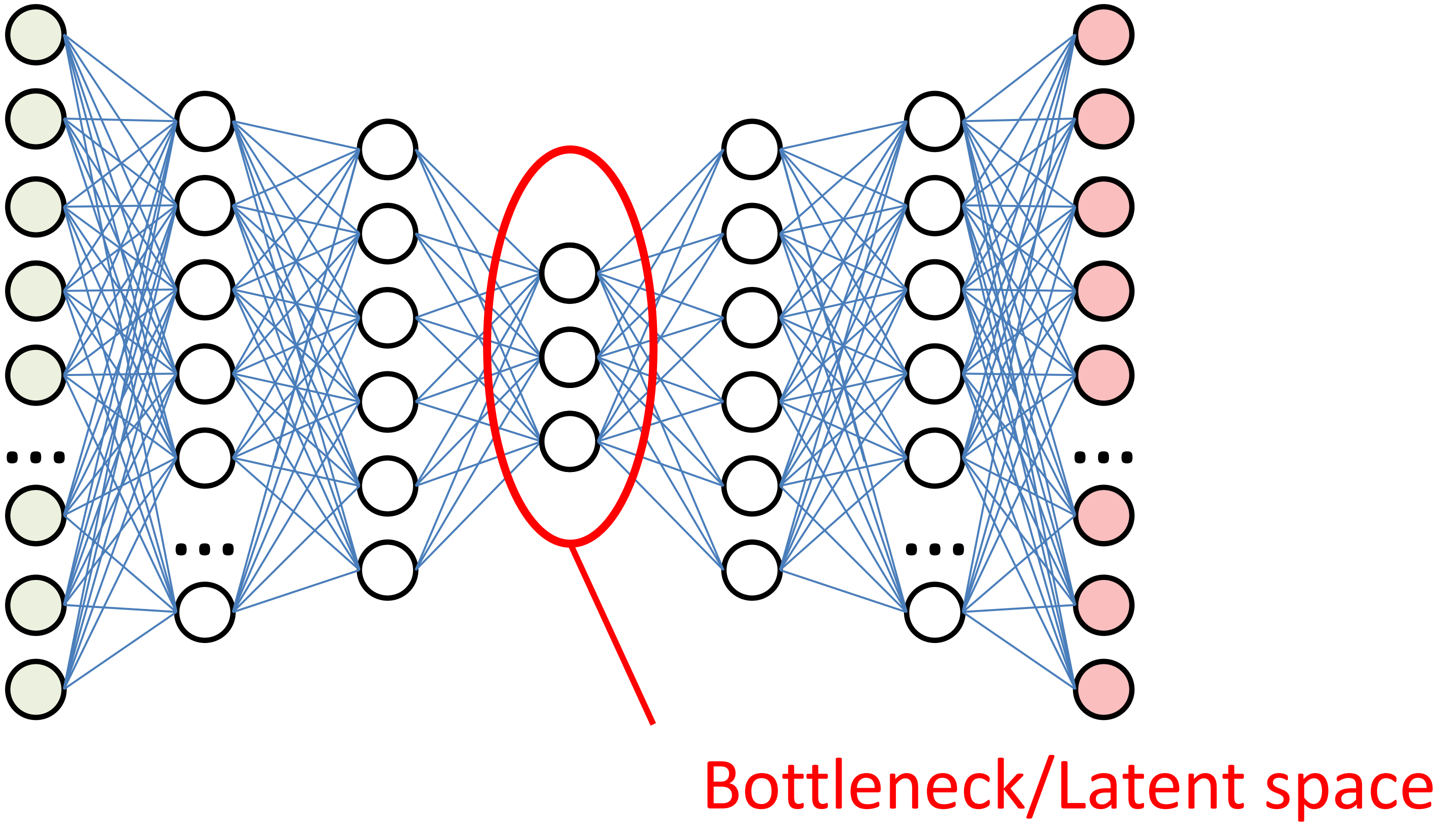
- Train unsupervised to **exactly reproduce the input**

Optimization task: $\operatorname{argmin}\left(\sum_{i=0}^{N} |\, \hat{x}(i) - x(i) \,|\right)$

$$x \longrightarrow \boxed{\text{Autoencoder}} \longrightarrow \hat{x}$$

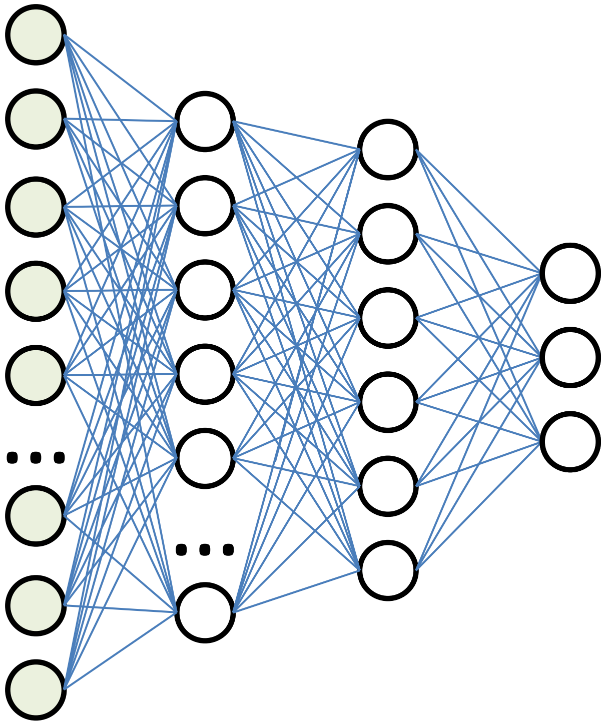
Autoencoders

- Force the computation to go through bottleneck
- Dimension of the bottleneck much smaller than input



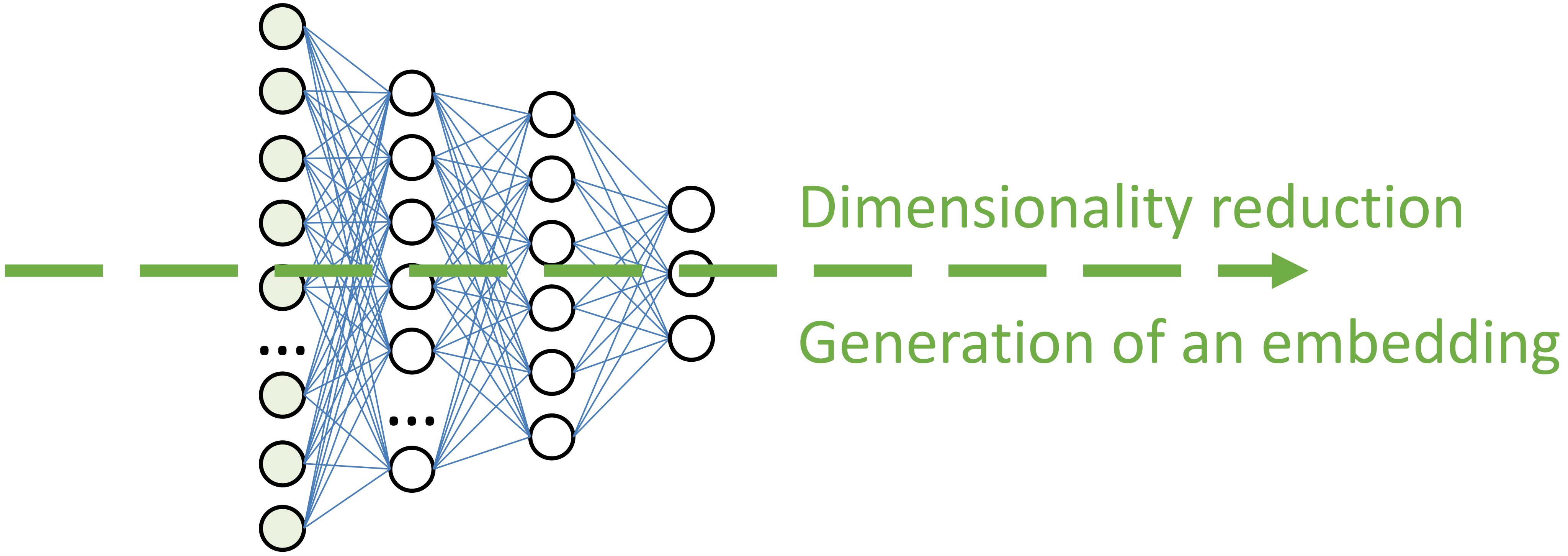
Autoencoders

- Remove the second part of the model



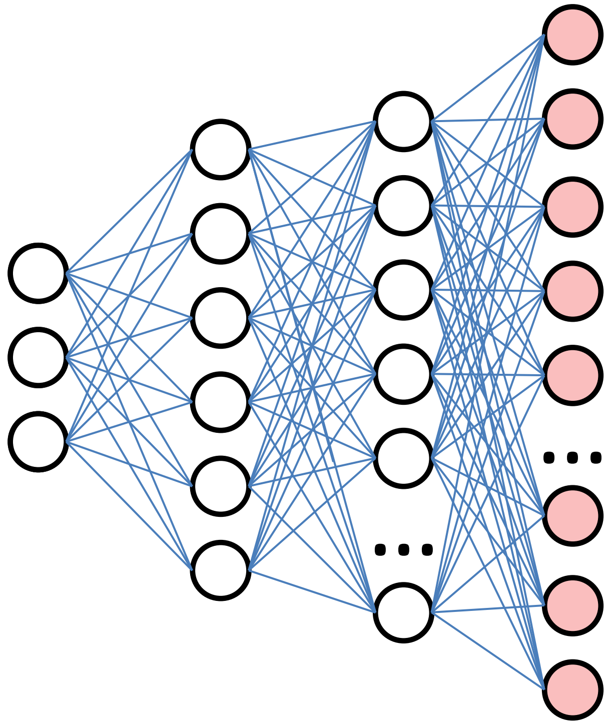
Autoencoders

- Remove the second part of the model
- Going from input to (lower dimensionality) bottleneck



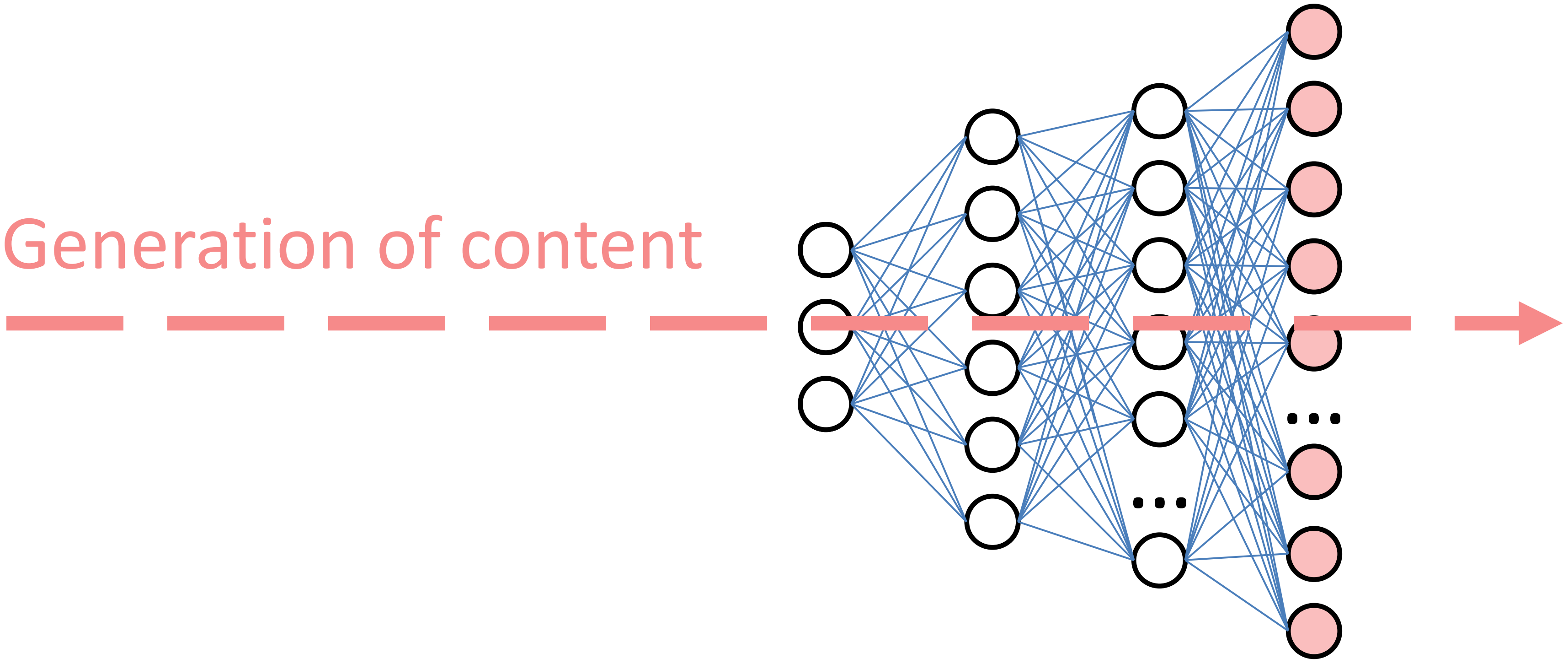
Autoencoders

- However, we can also remove the first part
- From the bottleneck/latent space to the output

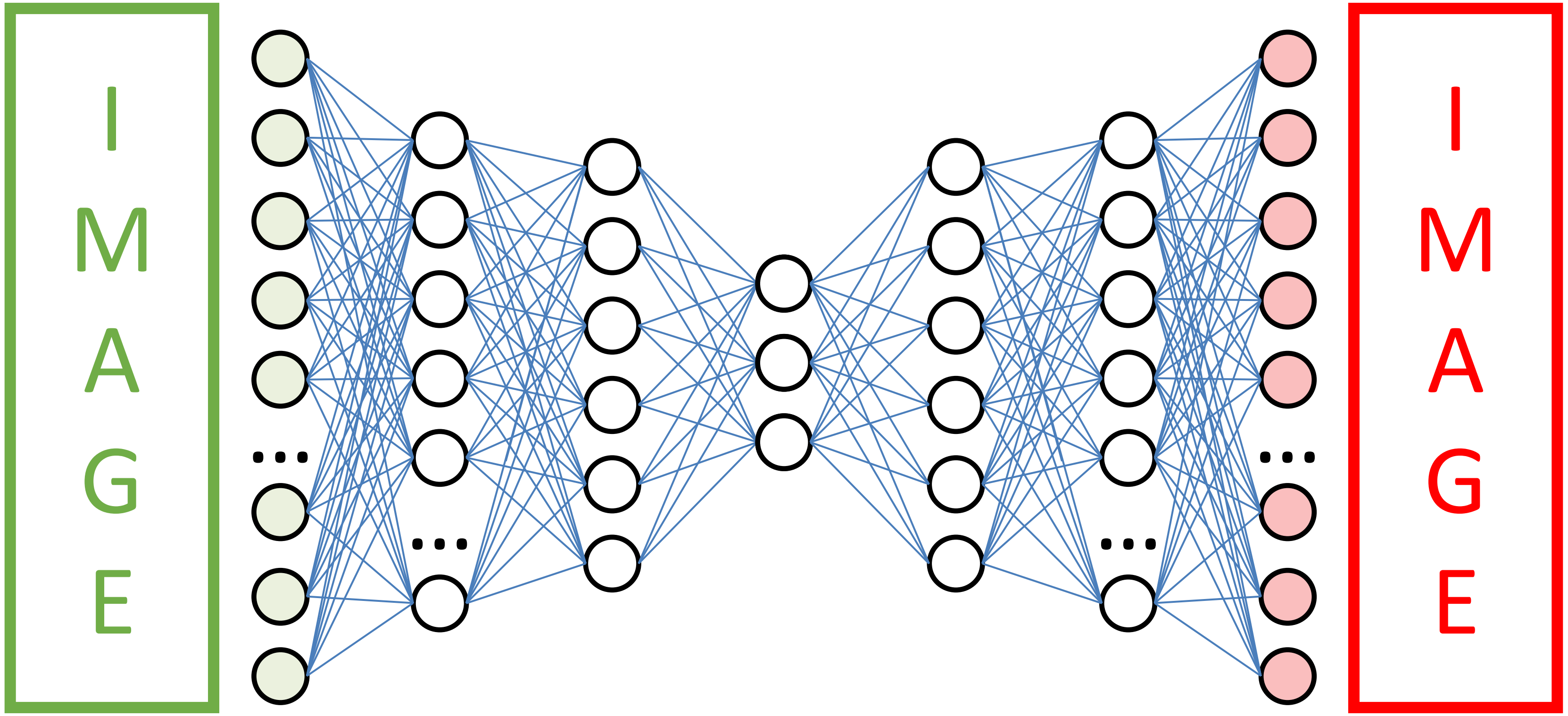


Autoencoders

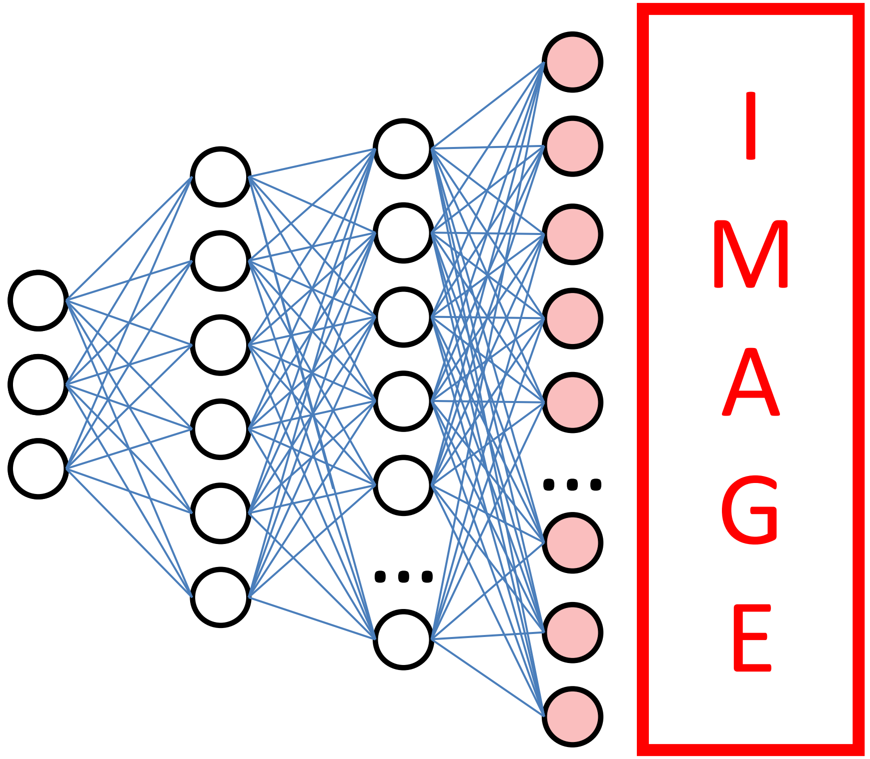
- However, we can also remove the first part
- From the bottleneck/latent space to the output



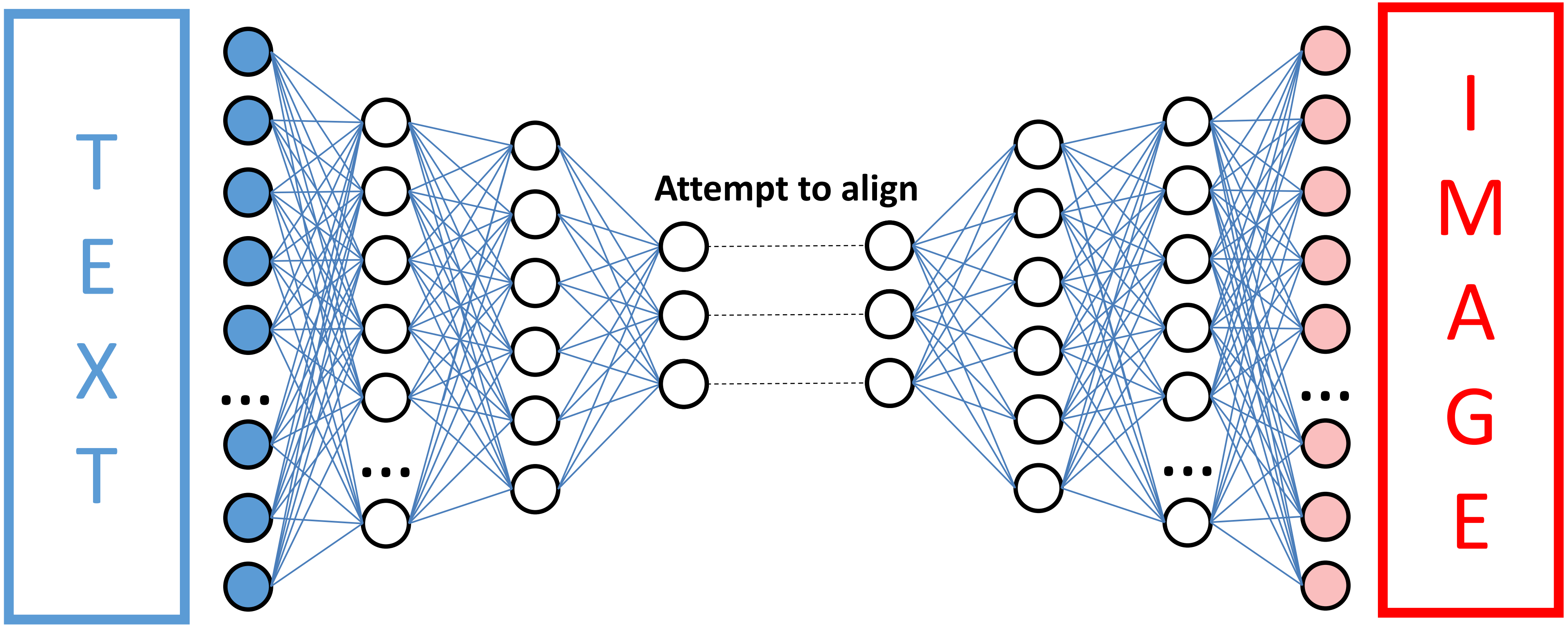
Autoencoders and generative NNs



Autoencoders and generative NNs



Autoencoders and generative NNs



From the point of view of optimization…

- Interesting results from a 2016 paper, "Understanding deep learning requires rethinking generalization"
  - Shows that a neural network can *memorize* a dataset
  - In other words, it has enough **capacity** to overfit completely
  - But when there is an actual relationship $y = f(X)$, **finds it**
- From an optimization point of view
  - There is a **global optimum** of the weights value
  - Global optimum corresponds to **total overfit**
  - SGD finds a *local optimum* that has **better generalization** (!)

Questions?

Bibliography

- Goodfellow et al., *The Deep Learning Book*, 2016
- Zhang et al., *Understanding Deep Learning Requires Rethinking Generalization*, 2016