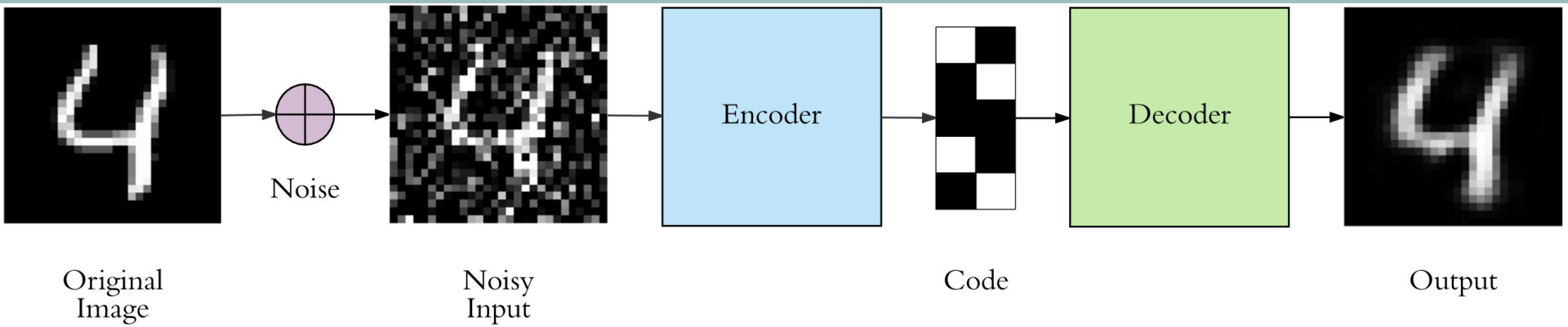


Deep Unsupervised Learning: Autoencoders

AGENDA

- Unsupervised Learning
- Autoencoder (AE)
- Convolutional AE
- Denoising AE
- Deep AE
- Take-home Message

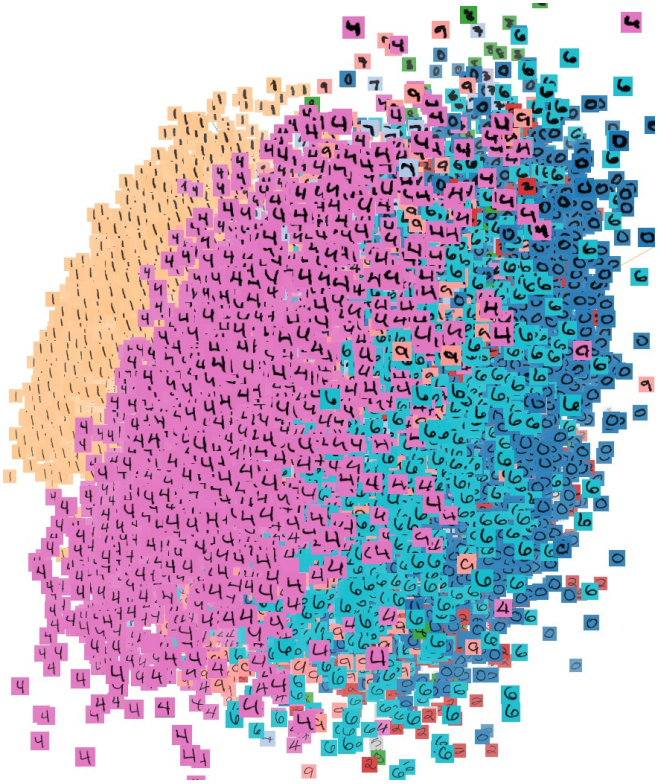
UNSUPERVISED LEARNING

Data: X (no labels! no Y)

Goals:
- Learn the structure of the data
- learn appropriate **representation** of the data

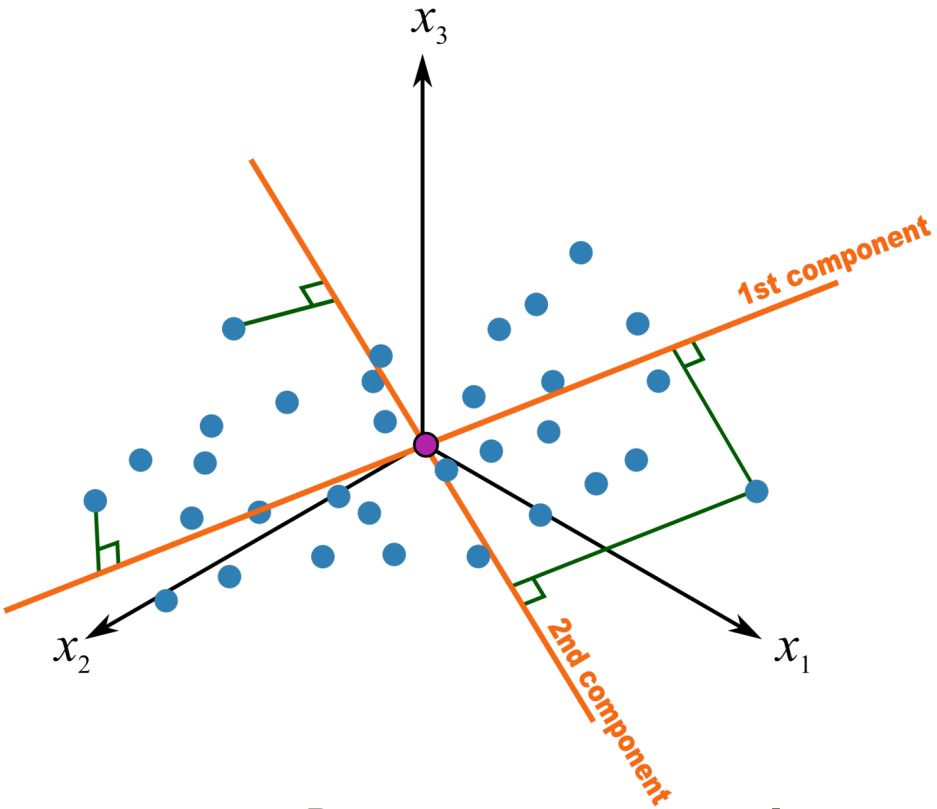
Examples: **Clustering, Dimensionality reduction, Feature & Representation learning**, Generative models ,etc.

You probably learn it in graduate school.



PCA – PRINCIPAL COMPONENT ANALYSIS

- Statistical approach for data compression and visualization
- Invented by Karl Pearson in 1901
- Weakness: linear components only.



PCA Representation! Eigen-vector Representation!

REPRESENTATION LEARNING- AUTOENCODERS (AE)

Used for unsupervised learning

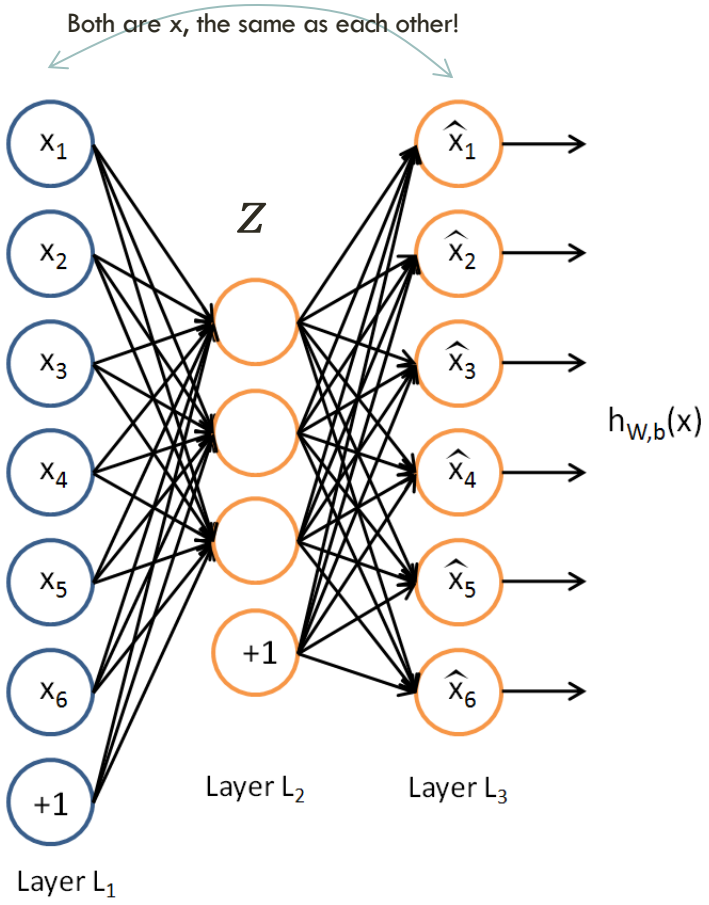
It is a network that learns an efficient coding of its input.

The objective is simply to reconstruct the input, but through the intermediary of a compressed or reduced-dimensional representation.

If the output is formulated using probability, the objective function is to optimize $p(\mathbf{x} = \hat{\mathbf{x}} \mid \tilde{\mathbf{x}})$, that is, the probability that the model gives a random variable x the value $\hat{\mathbf{x}}$ given the observation $\tilde{\mathbf{x}}$, where $\hat{\mathbf{x}} = \tilde{\mathbf{x}}$.

In other words, the model is trained to predict its own input—but it must map it through a representation created by the hidden units of a network.

TRADITIONAL AUTOENCODER



An Autoencoder is a feedforward neural network that learns to predict the input itself in the output.

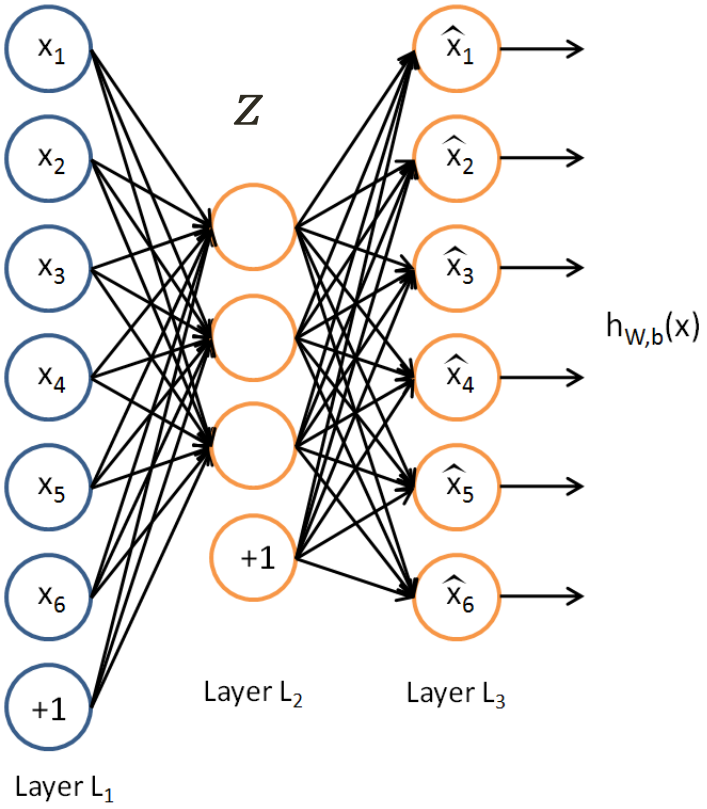
$$y^{(i)} = x^{(i)}$$

The input-to-hidden part corresponds to an encoder

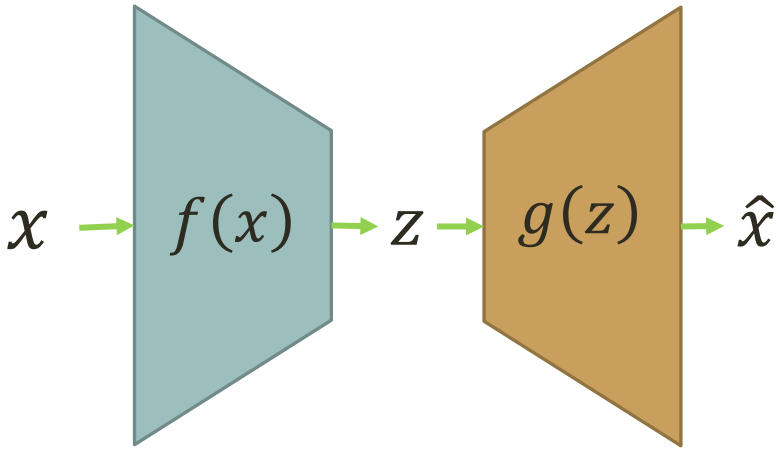
The hidden-to-output part corresponds to a decoder.

TRADITIONAL AUTOENCODER

- Unlike the **PCA** now we can use activation functions to achieve non-linearity.
- It has been shown that an AE without activation functions achieves the **PCA** capacity, i.e. PCA being a special case of AE.



SIMPLE IDEA



- Given data x (no labels) we would like to learn the functions $f(x)$ (encoder) and $g(z)$ (decoder) where:

$$f(x) = s(wx + b) = z$$

$$g(z) = s(w'z + b') = \hat{x}$$

s.t $h(x) = g\big(f(x)\big) = \hat{x}$

where h is an **approximation** of the identity function.

(z is some **latent** representation or **code** and s is a non-linearity such as the sigmoid)

($\hat{x}$ is x's reconstruction)

SIMPLE IDEA

Learning the identity function seems trivial, but with added constraints on the network (such as limiting the number of hidden neurons or regularization or dropout) we can learn information about the structure of the data.

Trying to capture the distribution of the data (data specific!)

TRAINING THE AE

Using **Gradient Descent,** we can simply train the model as any other NN with:

- Traditionally with *squared error loss* function

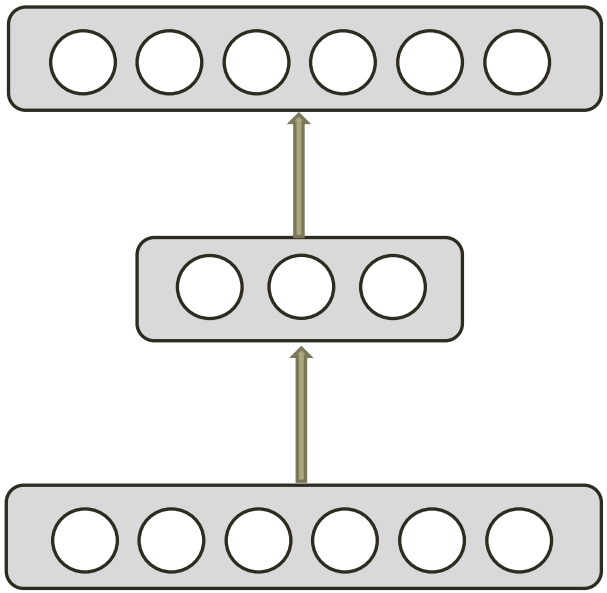
$$L(x, \hat{x}) = \|x - \hat{x}\|^2$$

- If our input is interpreted as bit vectors or vectors of bit probabilities the *cross entropy* can be used

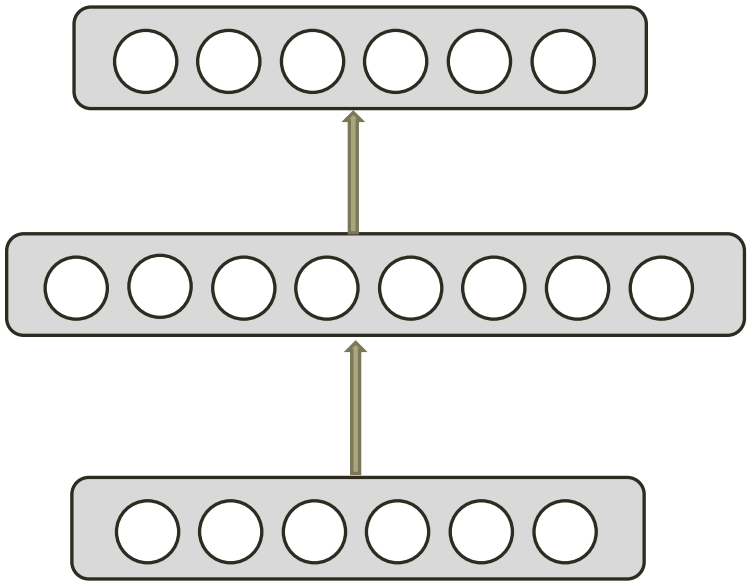
$$H(p, q) = -\sum_x p(x) \log q(x)$$

UNDERCOMPLETE AE VS OVERCOMPLETE AE

We distinguish between two types of AE structures:



Undercomplete AE

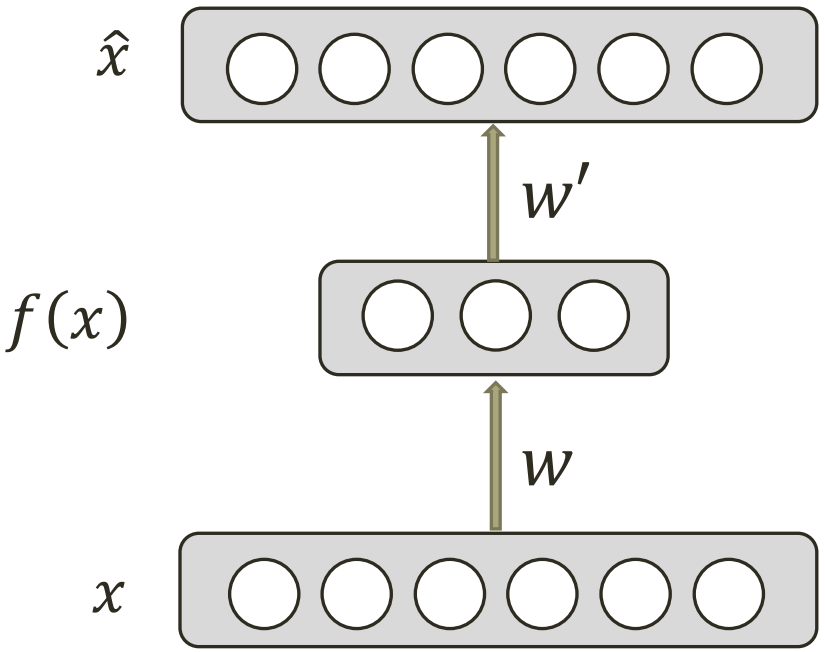


Overcomplete AE

UNDERCOMPLETE AE

- Hidden layer is **Undercomplete** if smaller than the input layer
  - Compresses the input
  - Compresses well only for the training distribution

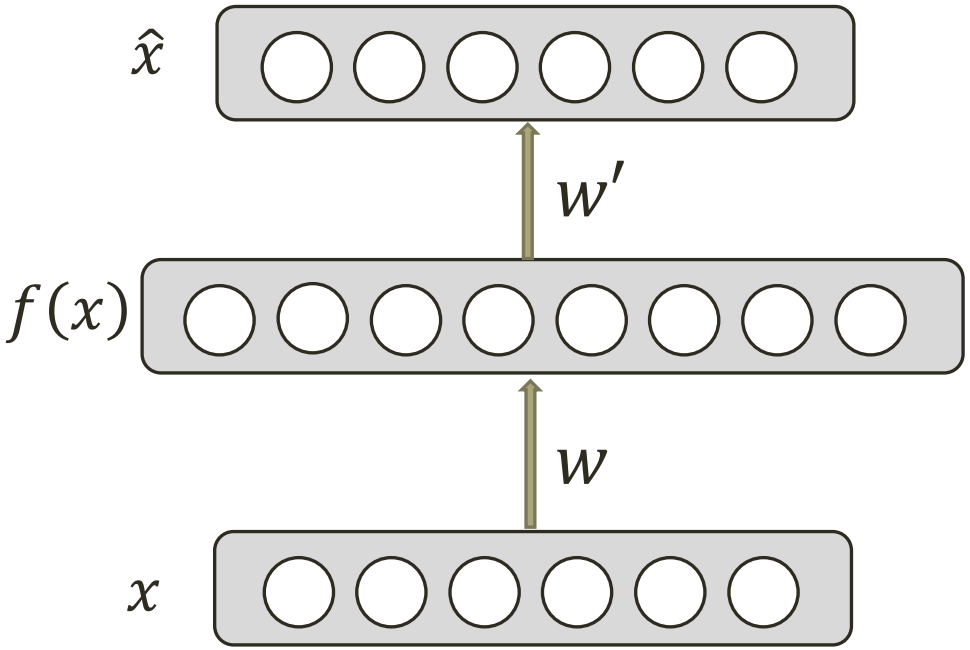
- Hidden nodes will be
  - Good features for the training distribution.
  - Bad for other types on input



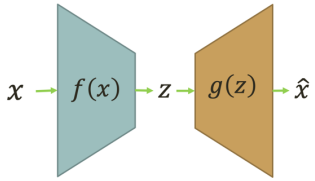
OVERCOMPLETE AE

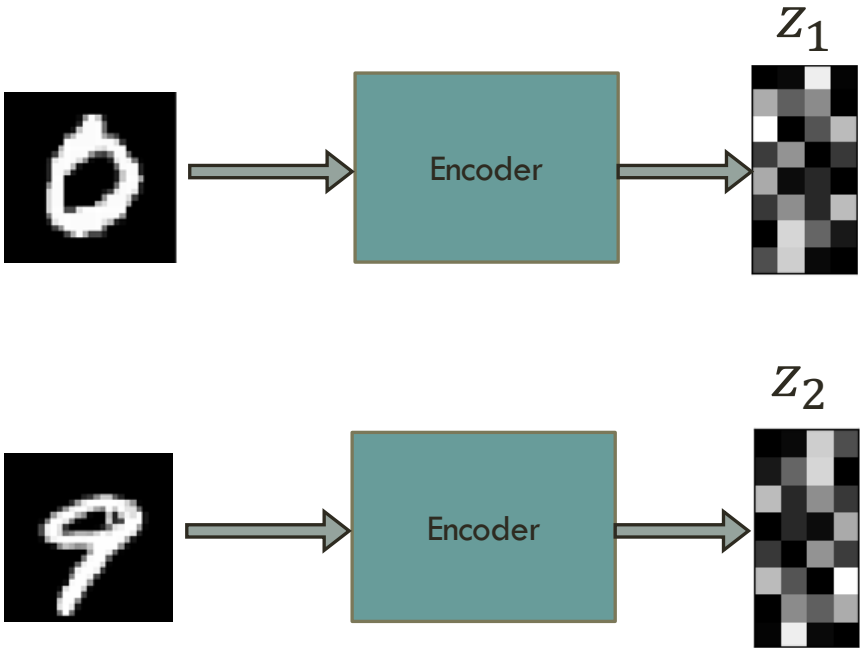
- Hidden layer is **overcomplete** if greater than the input layer
  - No compression in hidden layer.
  - Each hidden unit could copy a different input component.
- No guarantee that the hidden units will extract meaningful structure.

- Adding dimensions is good for training a linear classifier (at the output), i.e., easier to train
- A higher dimensional code helps model a more complex distribution.



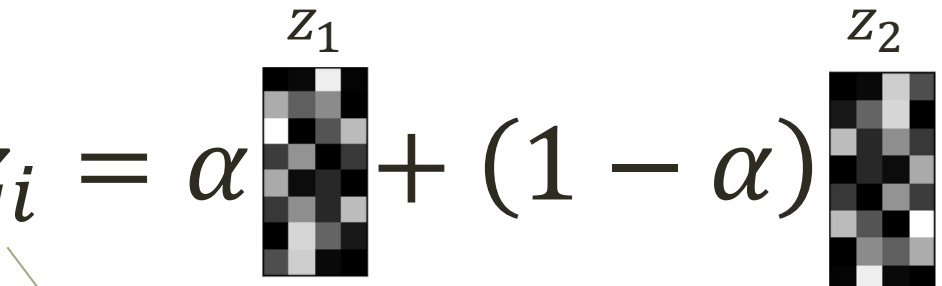
SIMPLE LATENT SPACE INTERPOLATION



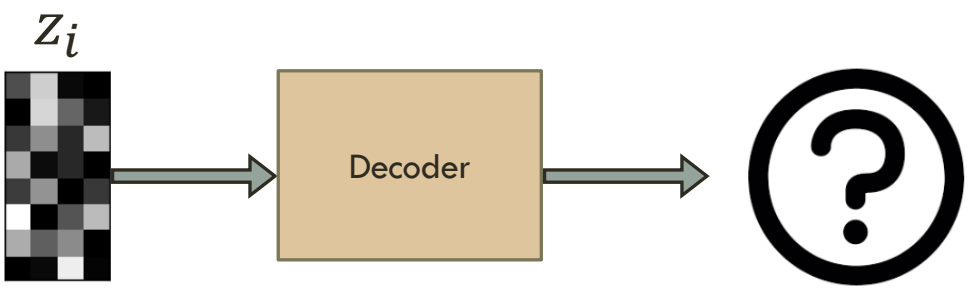


SIMPLE LATENT SPACE INTERPOLATION

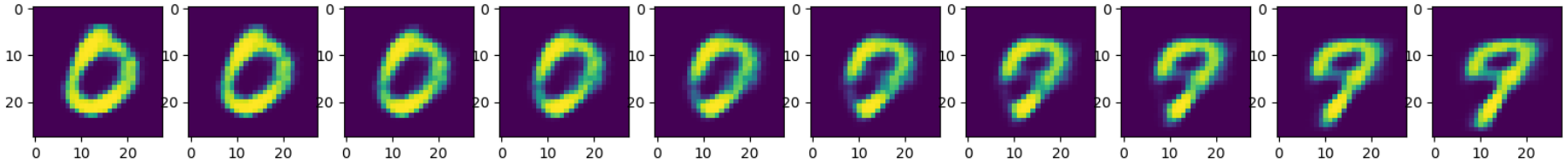
$$z_i = \alpha z_1 + (1-\alpha) z_2$$



Look like word2vec!



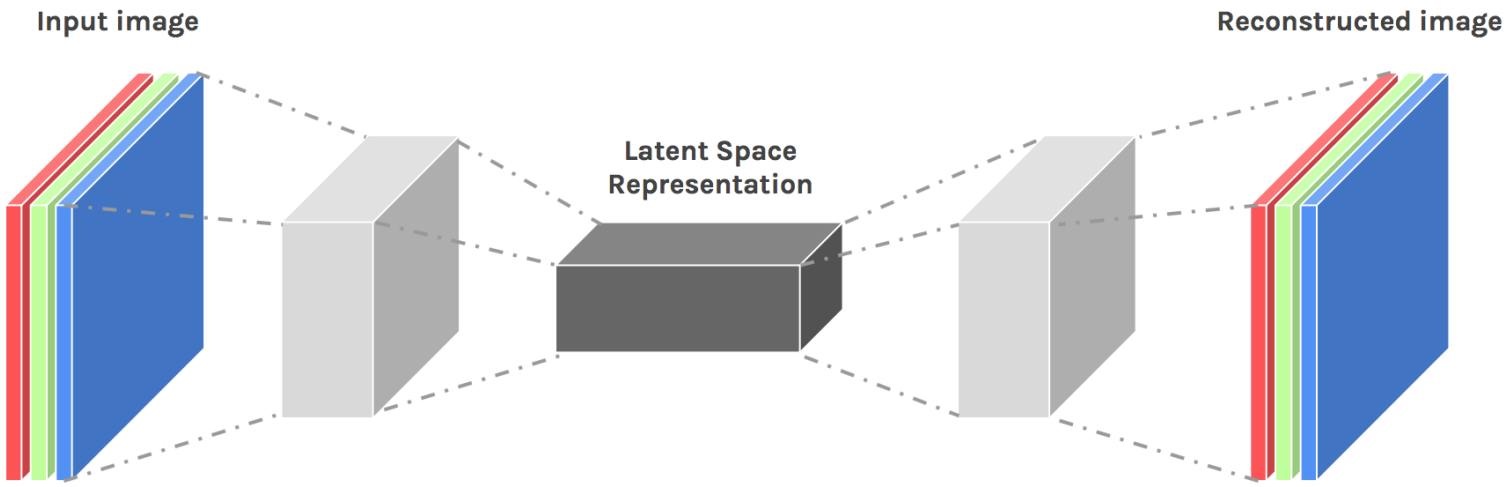
SIMPLE LATENT SPACE – INTERPOLATION



Closer to “0” α Closer to “9”

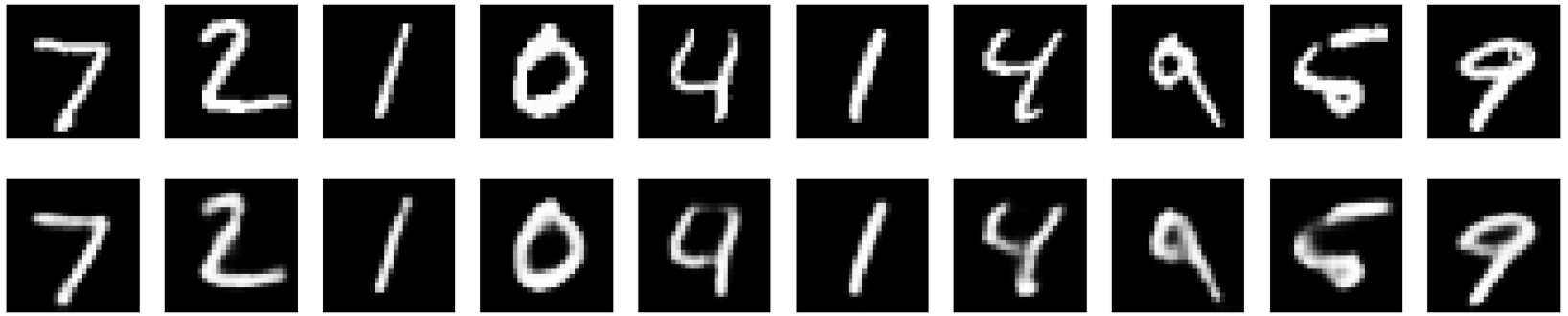
Varying alpha from 1 to 0

CONVOLUTIONAL AE

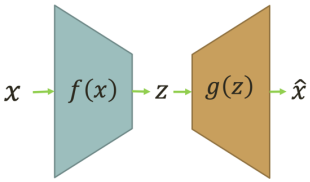


CONVOLUTIONAL AE EXAMPLE RESULTS

Input image



Any kind of AE can be used.



Output image

After the convolutional AE training is completed and based on the latent representation z learnt, we can use any classifier (e.g. kNN, DT, MLP) to classify a given hand-written digit. Experiments done and results obtained as:

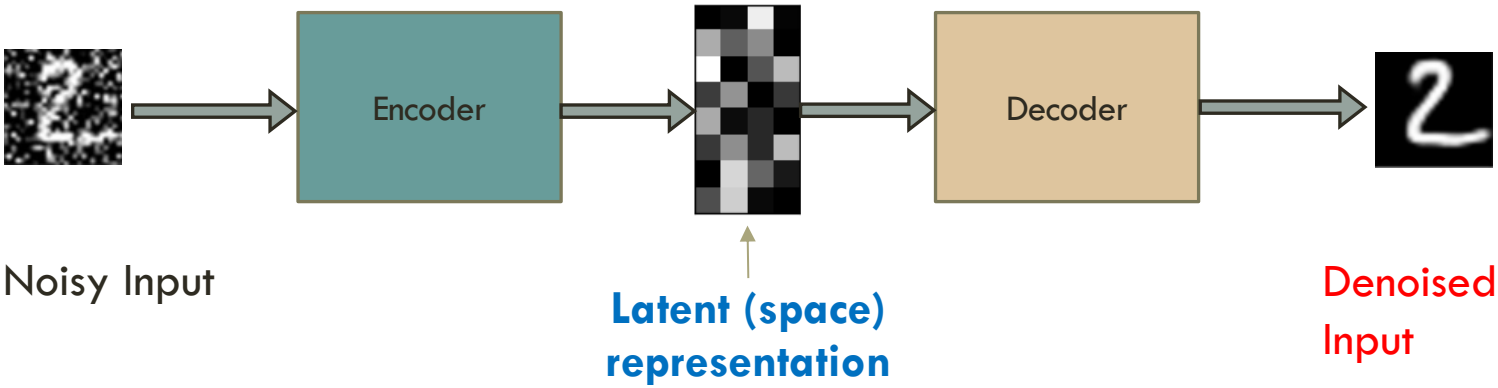
- 50 epochs.
- 88% accuracy on validation set.

DENOISING AUTOENCODERS

Intuition:

- We still aim to encode the input and to NOT mimic the identity function.
- We try to undo the effect of *corruption* process stochastically applied to the input.

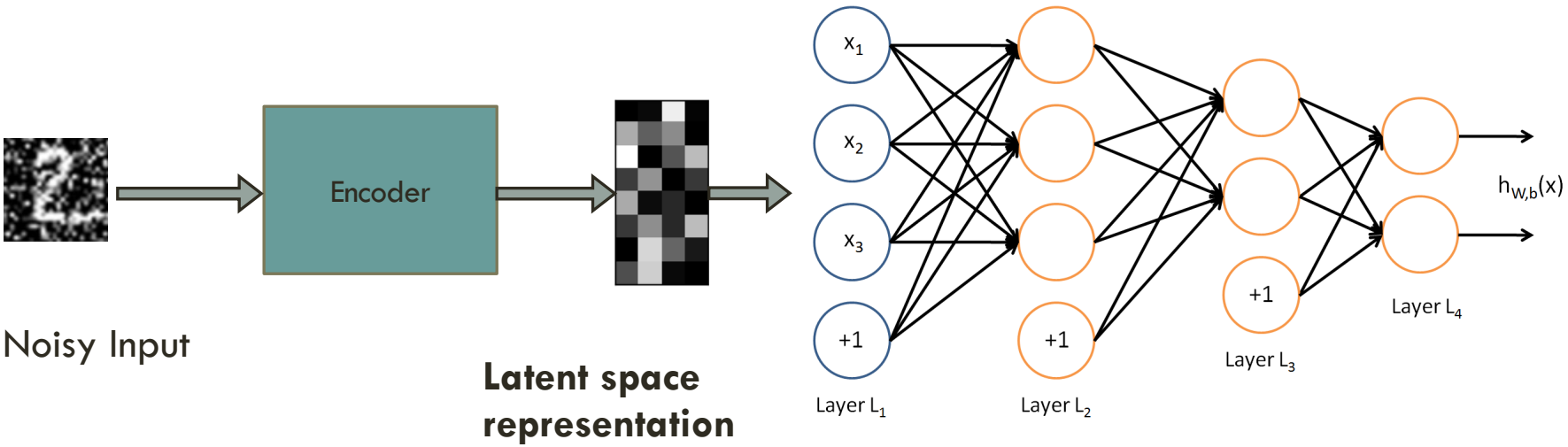
A more robust model



DENOISING AUTOENCODERS

Use Case:

- Extract robust representation (after AE training) for a NN classifier.
- Use representations learned from denoising AE and treat them as features for classifiers (like CNN)



Denoising encoding layers

Fc layers

DENOISING AUTOENCODERS (DAE)

Instead of trying to mimic the identity function by minimizing:

$$L\left(x, g\left(f\left(x\right)\right)\right)$$

where L is some loss function

A **DAE** instead minimizes:

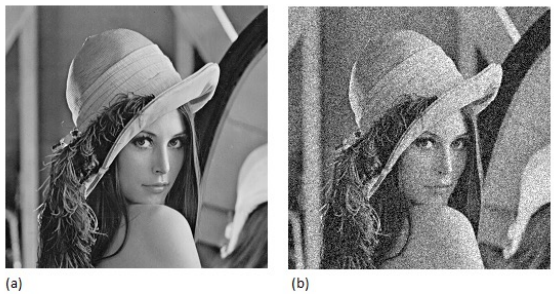
$$L\left(x, g\left(f\left(\tilde{x}\right)\right)\right)$$

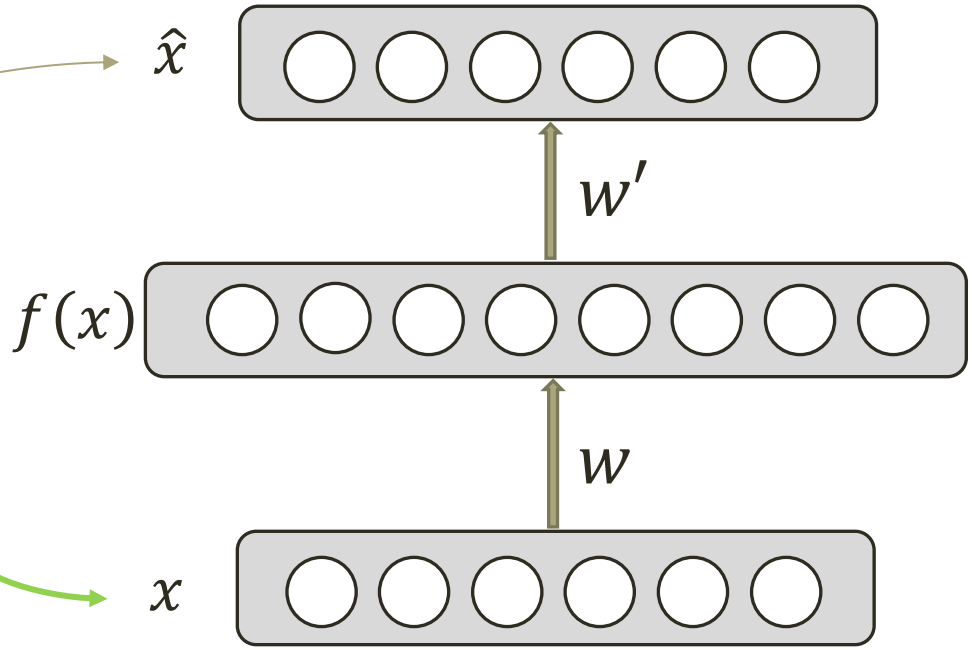
where $\tilde{x}$ is a copy of x that has been corrupted by some form of noise.

DENOISING AUTOENCODERS

Idea: A robust representation against noise:

- Random assignment of subset of inputs to 0, with probability v.
- Gaussian additive noise.





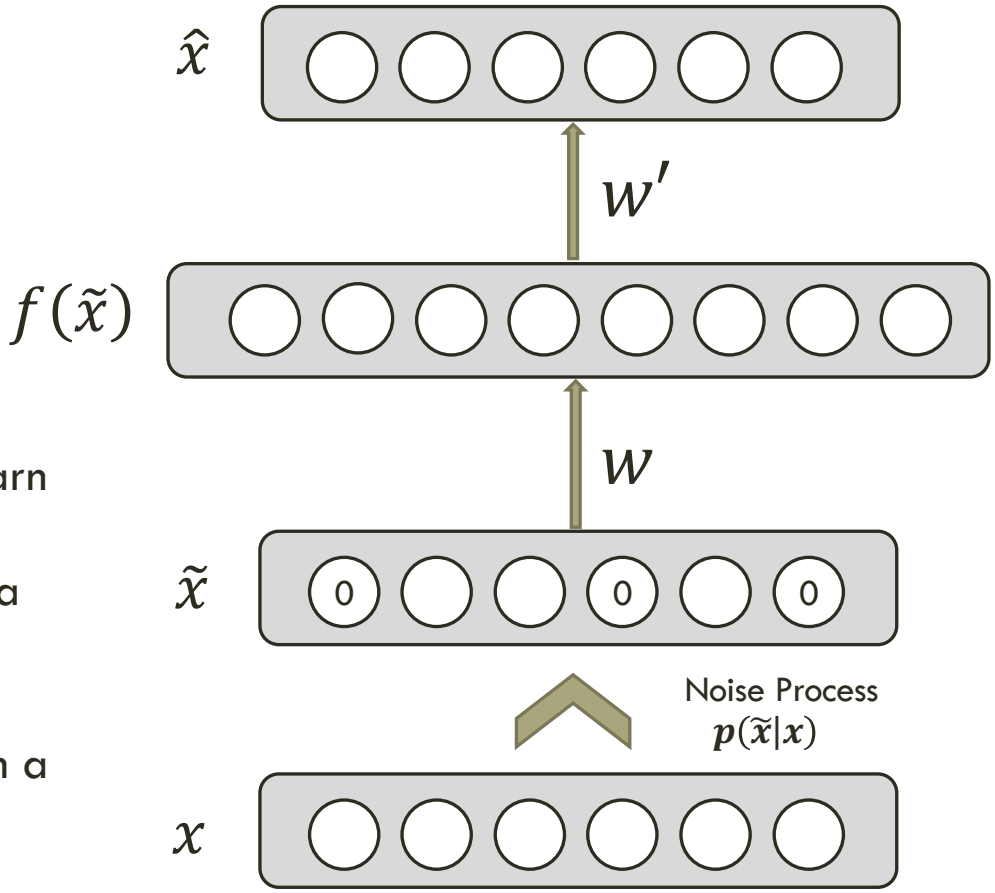
DENOISING AUTOENCODERS

- Reconstruction of $\hat{x}$ computed from the corrupted input $\tilde{x}$.
- Loss function compares $\hat{x}$ reconstruction with the noiseless x.

❖ The autoencoder cannot fully trust each feature of x independently so it must learn the correlations of x's features.

❖ Based on those relations, we can obtain a more 'not prone to changes' model.

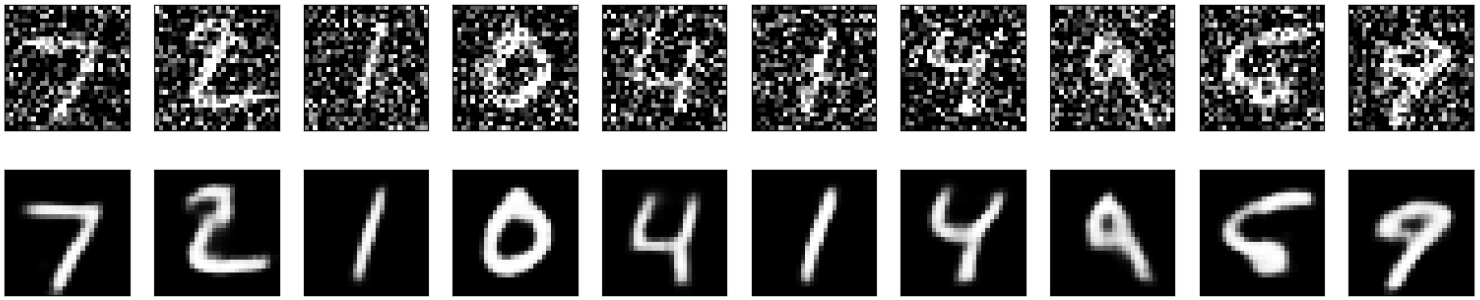
➢ We are forcing the hidden layer to learn a generalized structure of the data.



DENOISING CONVOLUTIONAL AE

A follow-up of the previous experiment
- 50 epochs.
- Noise factor 0.5
- 92% (improved from 88%) accuracy on validation set.

Input image



Output image

DEEP AUTOENCODERS

When building autoencoders from more flexible models, it is common to use a *bottleneck* in the network to produce an under-complete representation, providing a mechanism to obtain an encoding of lower dimension than the input.

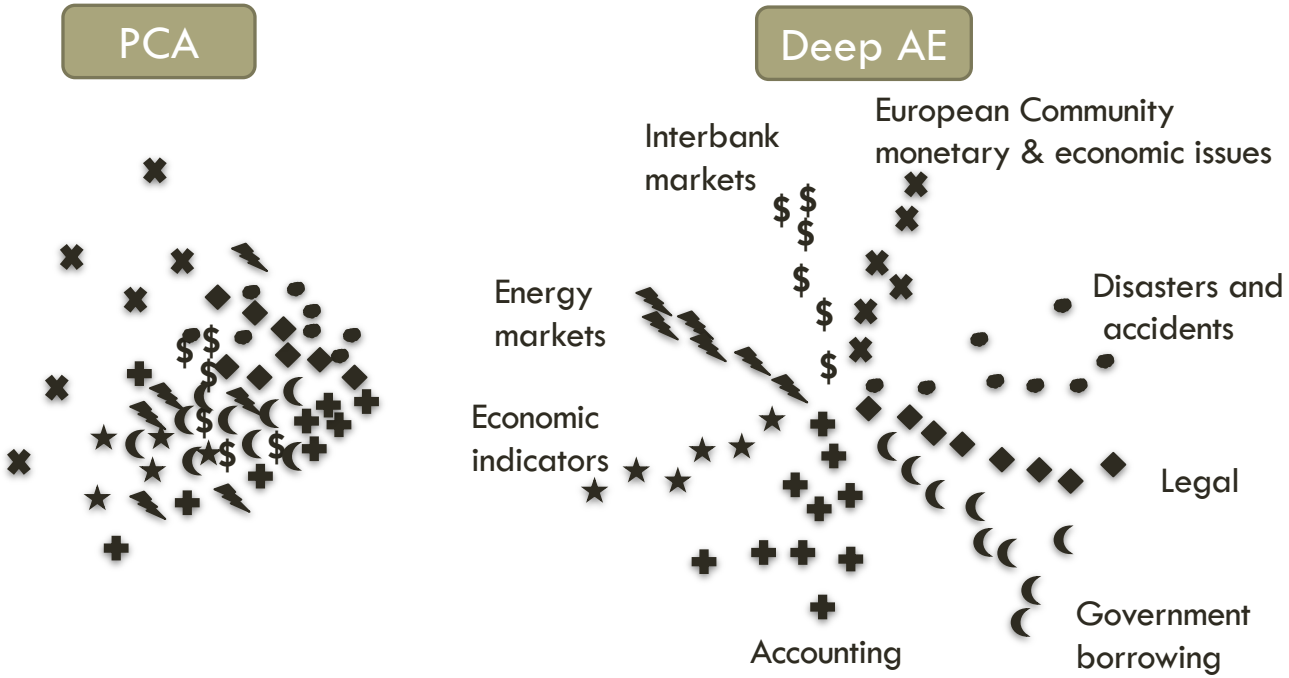
Deep autoencoders are able to learn low-dimensional representations with smaller reconstruction error than PCA using the same number of dimensions.

DEEP AUTOENCODER EXAMPLE

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html - By Andrej Karpathy

DEEP AUTOENCODERS

- A comparison of data projected into a 2D space with PCA (left) vs a deep autoencoder (right) for a text dataset
- The non-linear autoencoder can arrange the learned space in such that it better separates natural groupings of the data



Adapted from Hinton and Salakhutdinov (2006)

DEEP AUTOENCODERS

A deep Autoencoder is constructed by extending the encoder and decoder of autoencoder with multiple hidden layers.

Gradient vanishing problem: the gradient becomes too small as it passes back through many layers

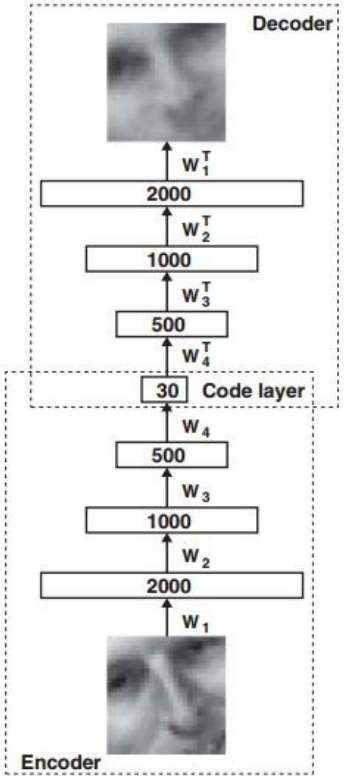


Diagram from (Hinton and Salakhutdinov, 2006)

TAKE-HOME MESSAGE

Autoencoder is simple enough architecturally so that it has been extended in different ways, deep model, denoising model, variational AE (VAE), etc.

It could also be used for dimensionality reduction (DimRed).

It could also be advantageous in understanding how human brain works.

Of course, it can be combined with supervised learning for more effective classification/regression.

REFERENCES

1. https://arxiv.org/pdf/1206.5538.pdf
2. http://www.deeplearningbook.org/contents/autoencoders.html
3. http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

Acknowledgement

- Guy Golan Presentation at Reichman U