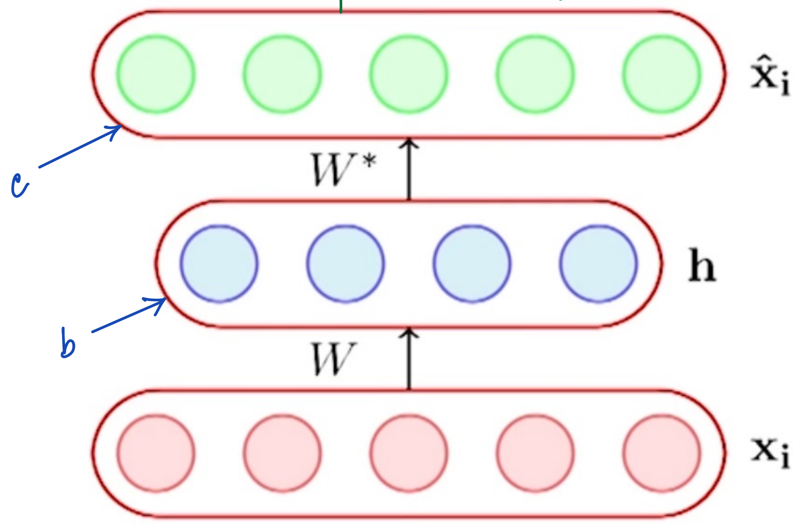
Autoencoders

(unsupervised model)



A special type of feedforward neural network which does:

Encodes its input x_i into a hidden representation h

$$h = g(Wx_i + b) \qquad \text{activation } f^n$$

Decodes the input again from this hidden representation

$$\hat{x}_i = f(W^*h + c) \qquad \text{output funct}^n$$

Why are we doing this?

We want to capture the most important characteristics of the input x_i.

The model is trained to minimize a loss function which ensures $\hat{x}_i$ and x_i are close.

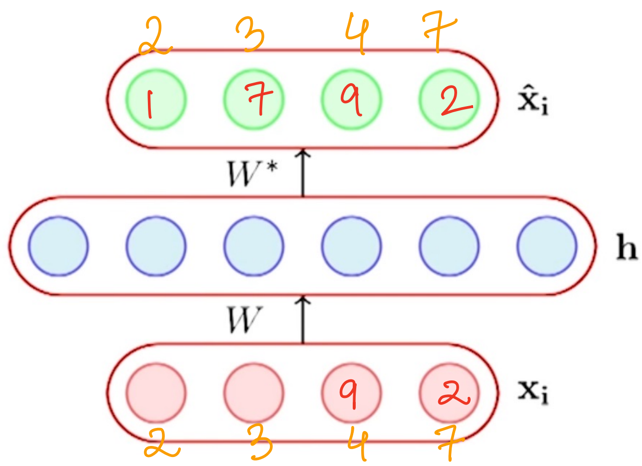
Case 1: $\dim(h) < \dim(x_i)$ (undercomplete auto-encoder)

If we are able to reconstruct $\hat{x}_i$ perfectly from h then h is a loss free encoding of x_i.

- h captures all important characteristics of x_i.

Case 2: $\dim(h) > \dim(x_i)$ (over complete auto-encoder)

In this case, auto-encoder could learn a trivial encoding - by simply copying x_i.



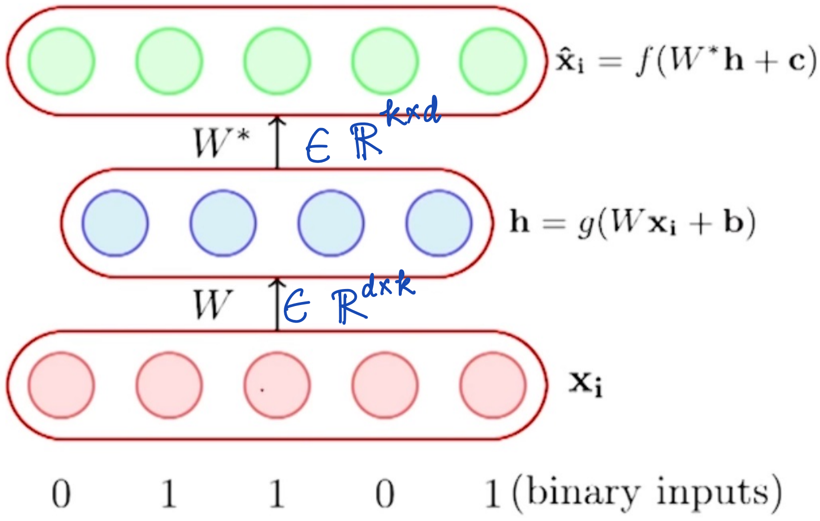
$$\mathbf{h} = g(W\mathbf{x_i} + \mathbf{b})$$

$$\hat{\mathbf{x}}_\mathbf{i} = f(W^*\mathbf{h} + \mathbf{c})$$

Choices of f and g

Case 1: Inputs are binary

$x_{ij} \in \{0, 1\}$



Since i/p is binary, we would want the o/p to also be binary.

What should f be?

- sigmoid : $\sigma(W^*h + c)$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Case 2 : Real inputs

$x_{ij} \in \mathbb{R}$

f : Linear

usual choice of g : tanh, sigmoid.

Choice of loss function

$x_i \in \mathbb{R}^d$

$(x_1, x_2, \ldots, x_d)$

Case 1: When input is real

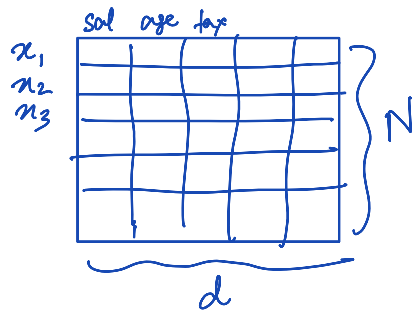
(i) Cross entropy

(ii) Mean squared error

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} \|x_i - \hat{x}_i\|_2^2 = \frac{1}{N}\sum_{i=1}^{N} \underbrace{\sum_{j=1}^{d} (x_{ij} - \hat{x}_{ij})^2}_{L_i(}$$

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} L_i(\theta)$$

$$L_i(\theta) = \sum_{j=1}^{d} (x_{ij} - \hat{x}_{ij})^2$$



Case 2: When input is binary

use cross entropy.

$$L(\theta) = \frac{1}{N}\sum L_i(\theta)$$

$$L_i(\theta) = -\left(\sum_{j=1}^{d} x_{ij} \log \hat{x}_{ij} + (1 - x_{ij}) \log (1 - \hat{x}_{ij})\right)$$

Train the NN using backpropagation with the objective

$$\min_{W, W^*, b, c} \mathcal{L}(\theta)$$

Link to PCA

Encoder part of auto-encoder is equivalent to PCA if:

1. We use a linear encoder
2. We use a linear decoder
3. Use squared error loss
4. Normalize the inputs:

$$\tilde{x}_{ij} = \frac{1}{\sqrt{N}}\left(x_{ij} - \bar{x}_{ij}\right)$$

Regularization

(i) l_2-regularization

$$\min \; \mathcal{L}(\theta) + \lambda \|\theta\|_2^2$$

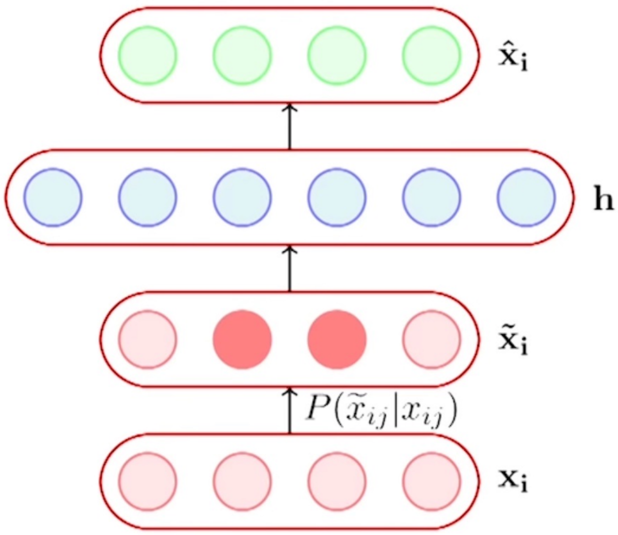
(ii) Weight Sharing

$$W^* = W^T$$

Other types of Auto-encoders

1. De-noising Auto-encoders.

Corrupts the input data using some probabilistic process $P(\tilde{x}_{ij} \mid x_{ij})$ before feeding it to the network



Consider binary input

$$P(\tilde{x}_{ij}=0 \mid x_{ij}) = 1-p \qquad\qquad P(\tilde{x}_{ij} = \text{not}(x_{ij}) \mid x_{ij}) = 1-p$$
$$P(\tilde{x}_{ij}=x_{ij} \mid x_{ij}) = p \qquad\qquad P(\tilde{x}_{ij} = x_{ij} \mid x_{ij}) = p$$

- It does not make sense for the network to memorize the input.
- Instead the model has capture good characteristics of the input.

2. <u>Sparse Auto-encoder</u>

A hidden neuron has values between 0 and 1.

activated when value is close to 1.

deactivated " " " " " 0.

A sparse auto-encoder tries to ensure that the neuron is deactivated most of the time.

Another way of doing regularization.