Deep Learning

Neural networks are cornerstone of deep learning

- Large data sets (digitization in science and industry)
- Capacities of computations + new architectures

- Image/video classification
- Speech/text modelling

- Basics of DL
- Convolutional neural networks
- Recurrent neural networks
- Limits of DL

Quick history



1958 Perceptron

1974 Backpropagation





Convolution Neural Networks for Handwritten Recognition

1998



Google Brain Project on 16k Cores

2012

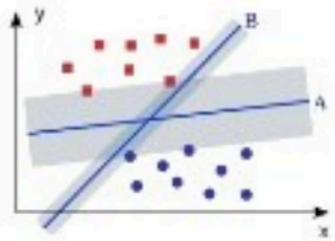


awkward silence (AI Winter)

1969
Perceptron criticized



1995
SVM reigns



2006
Restricted Boltzmann Machine



2012
AlexNet wins ImageNet

IMAGENET

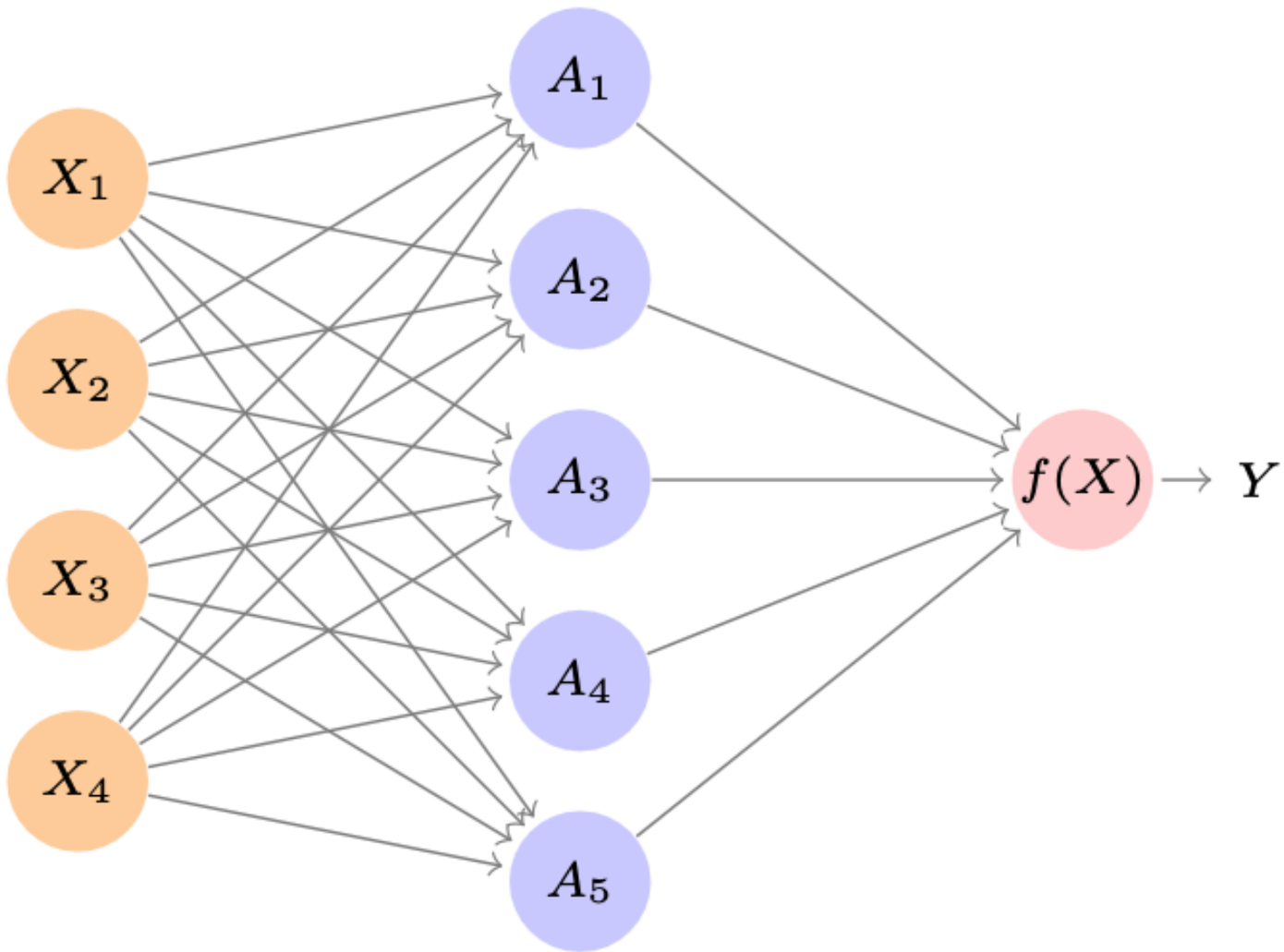
Image from `https://www.slideshare.net/LuMa921/deep-learning-a-visual-introduction`.

Single-layer neural networks

$$X = (X_1, X_2, \ldots, X_p) \mapsto f(X) \ \textit{predicts}\ Y$$

f : feed forward neural network

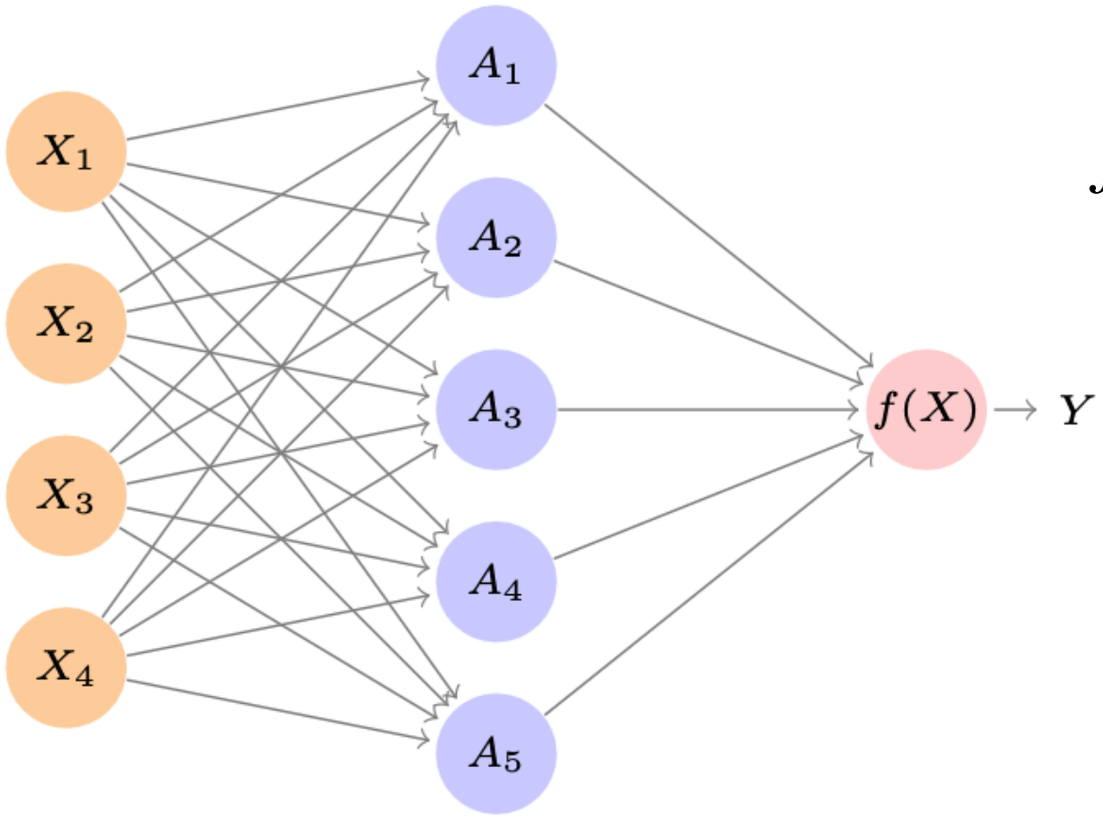
input layer hidden layer output layer



$$A_k = h_k(X)$$

K hidden units (neurons)

Single-layer neural networks



$$f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k h_k(X)$$

$$= \beta_0 + \sum_{k=1}^{K} \beta_k g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} X_j\right)$$

g : activation function (non-linear)

$$A_k = h_k(X) = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} X_j\right)$$

$$f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k A_k$$

Single-layer neural networks

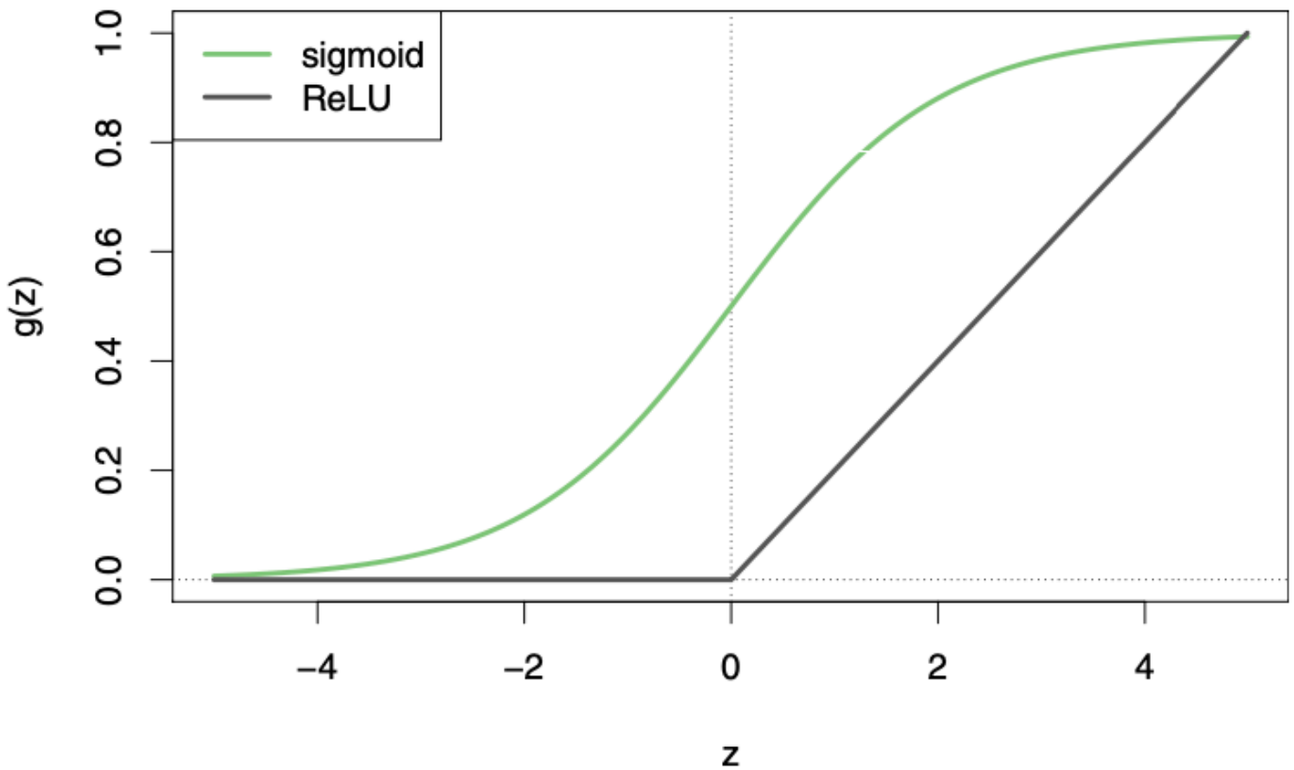
activation function g : non-linearity to capture
large classes of functions

Early Deep Learning: sigmoid

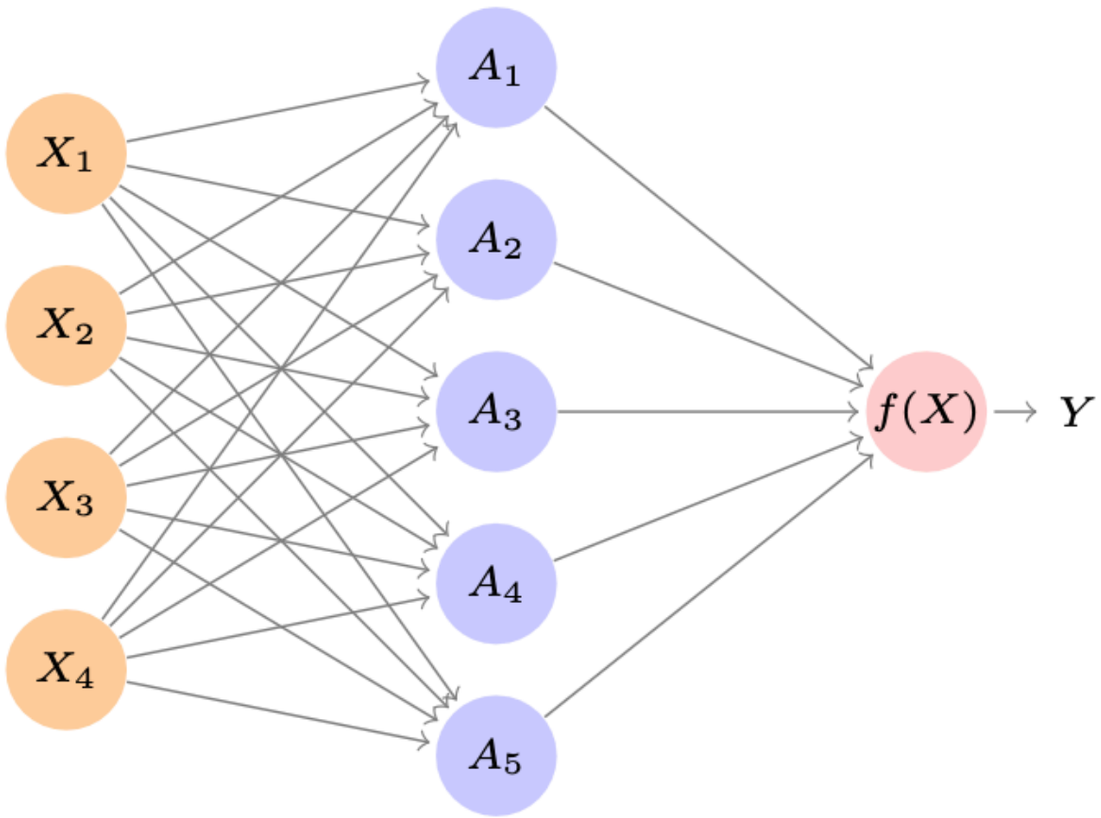
$$g(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$

Modern Deep Learning: ReLu

$$g(z) = (z)_+ = \begin{cases} z \text{ if } z > 0 \\ 0 \text{ o.w} \end{cases}$$



Single-layer neural networks



$$A_k = h_k(X) = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} X_j\right)$$

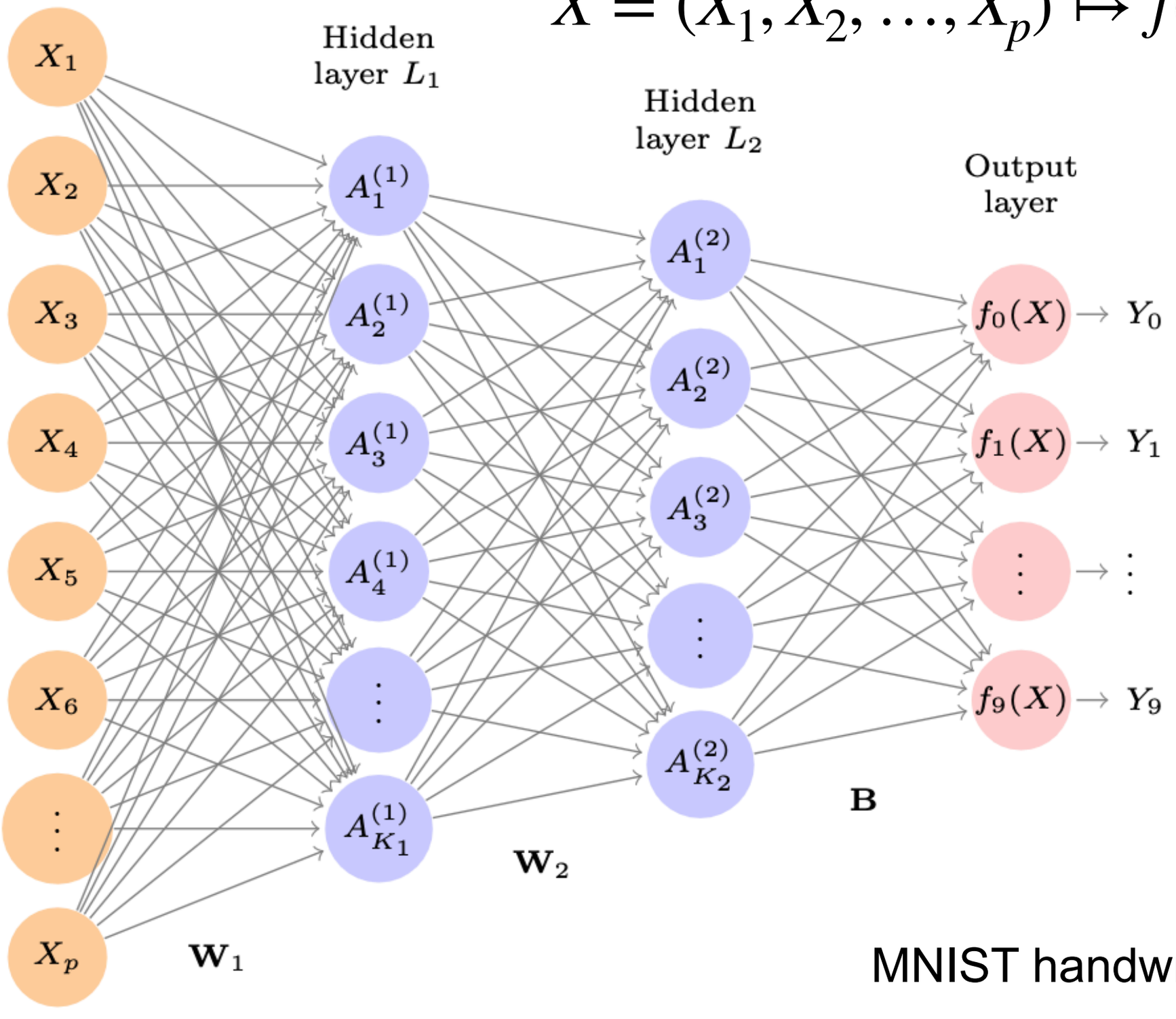
$$f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k A_k$$

Fitting the neural network:
estimating parameters β, w

Squared-error loss: $\min \sum_{i=1}^{n} (y_i - f(x_i))^2$

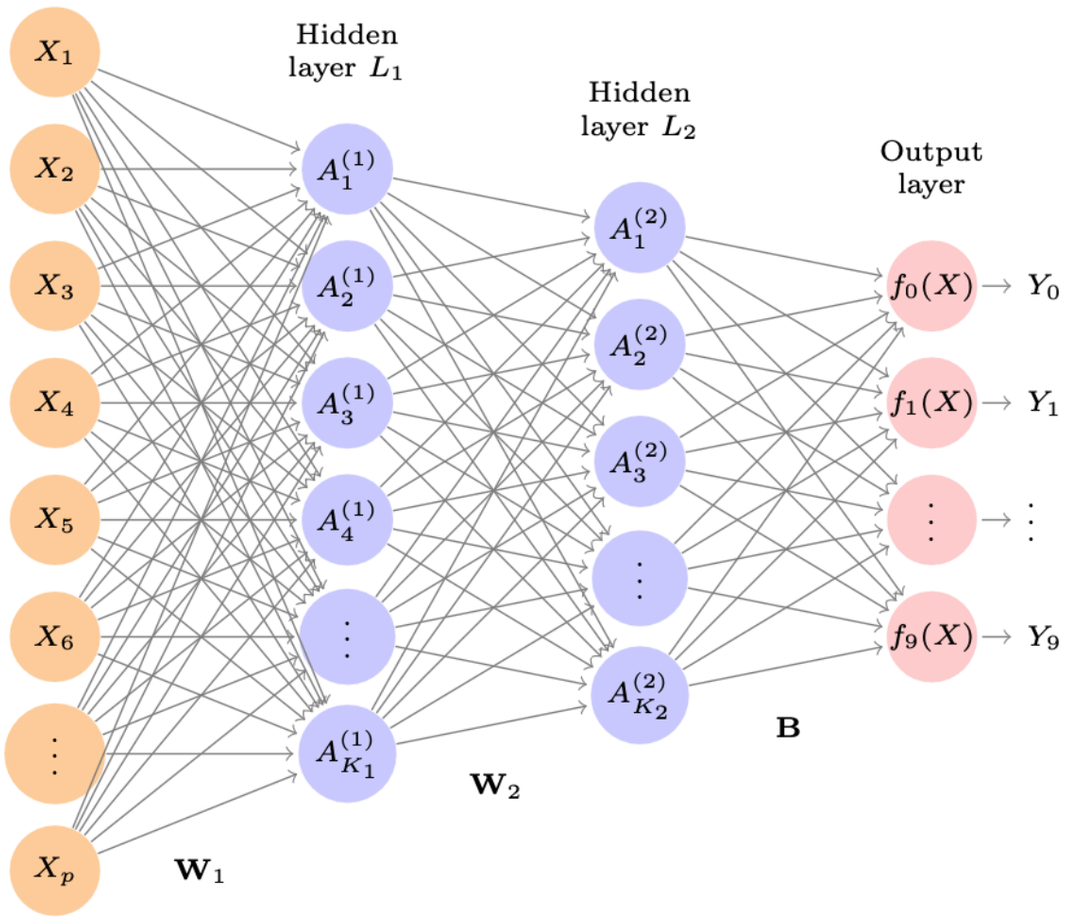
Multilayer neural networks

$$X = (X_1, X_2, \ldots, X_p) \mapsto f(X) \text{ predicts } Y$$



MNIST handwritten-digit problem

Multilayer neural networks



$$A_k^{(1)} = h_k^{(1)}(X) = g\left(w_{k0}^{(1)} + \sum_{j=1}^{p} w_{kj}^{(1)} X_j\right)$$

$$A_\ell^{(2)} = h_\ell^{(2)}(X) = g\left(w_{\ell 0}^{(2)} + \sum_{j=1}^{p} w_{\ell j}^{(2)} A_k^{(1)}\right)$$

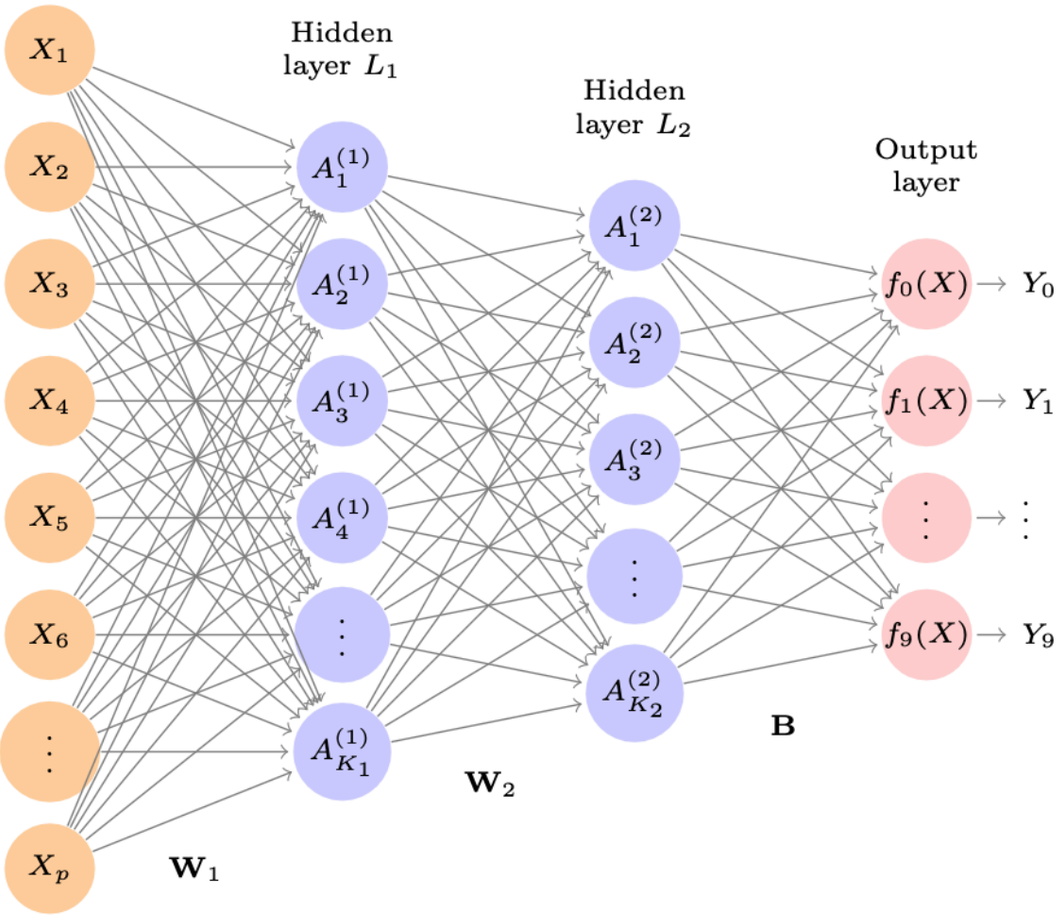
$$Z_m = \beta_{m0} + \sum_{\ell=1}^{K_2} \beta_{m\ell} A_\ell^{(2)}\Big)$$

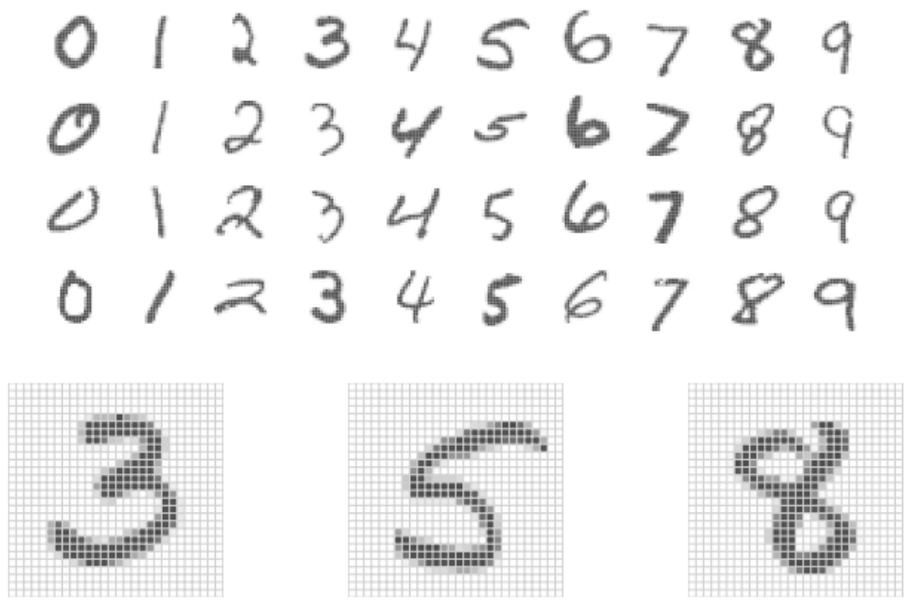
$f_m(X)$ represents $f_m(X) = \mathbb{P}\left[Y = m \,|\, X\right] = \dfrac{e^{Z_m}}{e^{Z_1} + e^{Z_1} + \ldots + e^{Z_9}}$

log-likelihood cost: (cross-entropy)

$$\min - \sum_{i=1}^{n} \sum_{m=0}^{9} y_{im} \log(f_m(x_i))$$

Multilayer neural networks





Performance

Method	Test Error
Neural Network + Ridge Regularization	2.3%
Neural Network + Dropout Regularization	1.8%
Multinomial Logistic Regression	7.2%
Linear Discriminant Analysis	12.7%