NYU

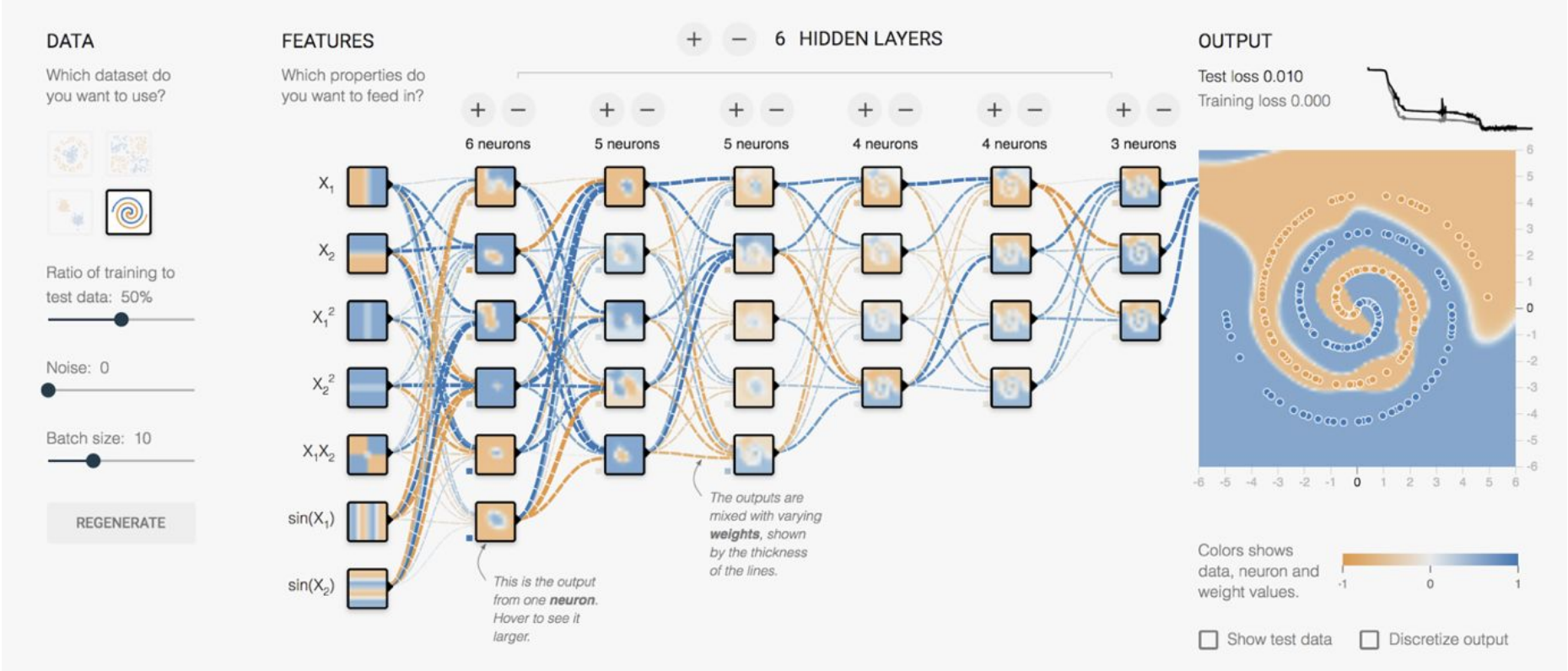
Introduction to Data Science

Center for Data Science
Iddo Drori, Spring 2019

NYU

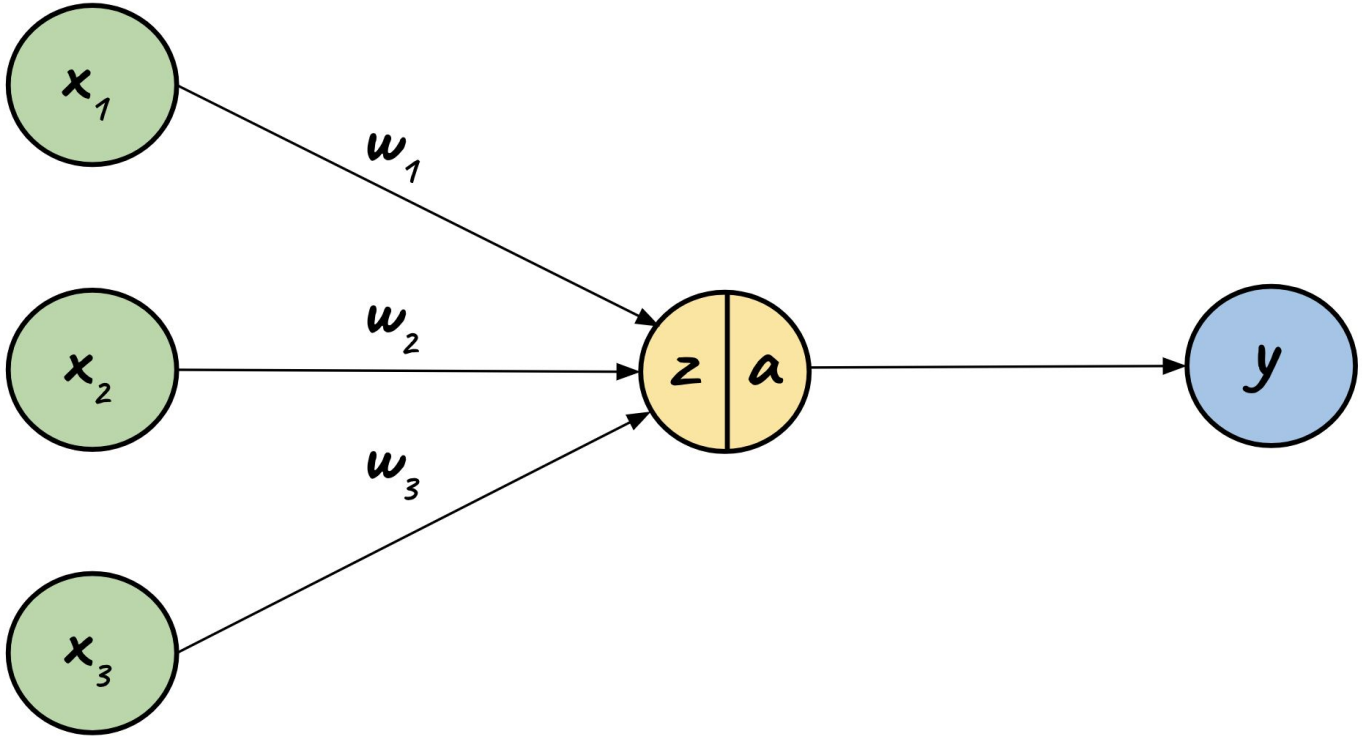
Neural Networks: Introduction

Fully Connected Neural Network

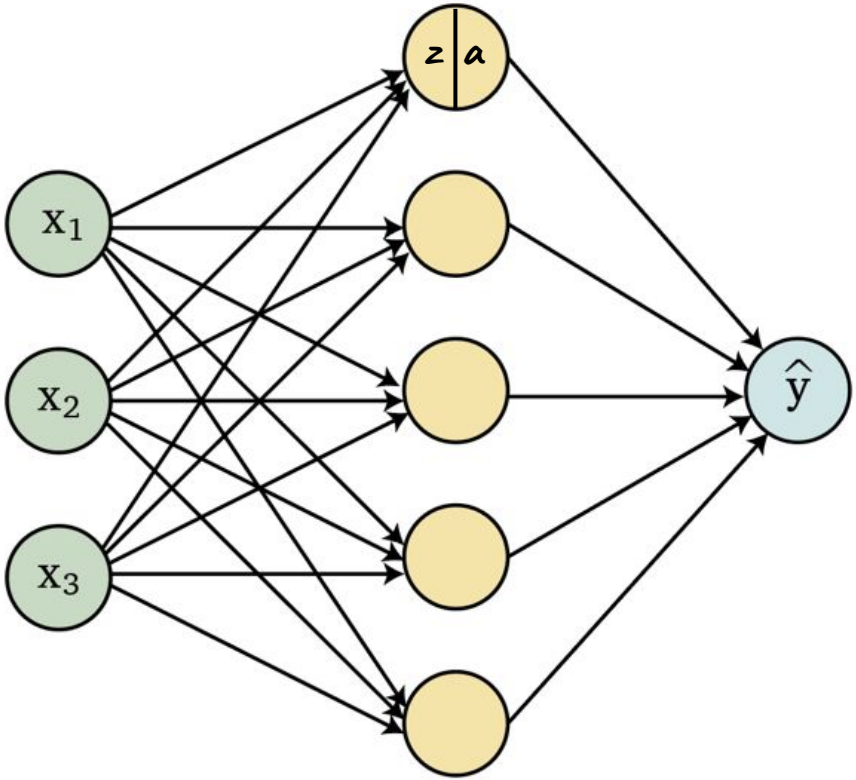


Source: playground.tensorflow.org

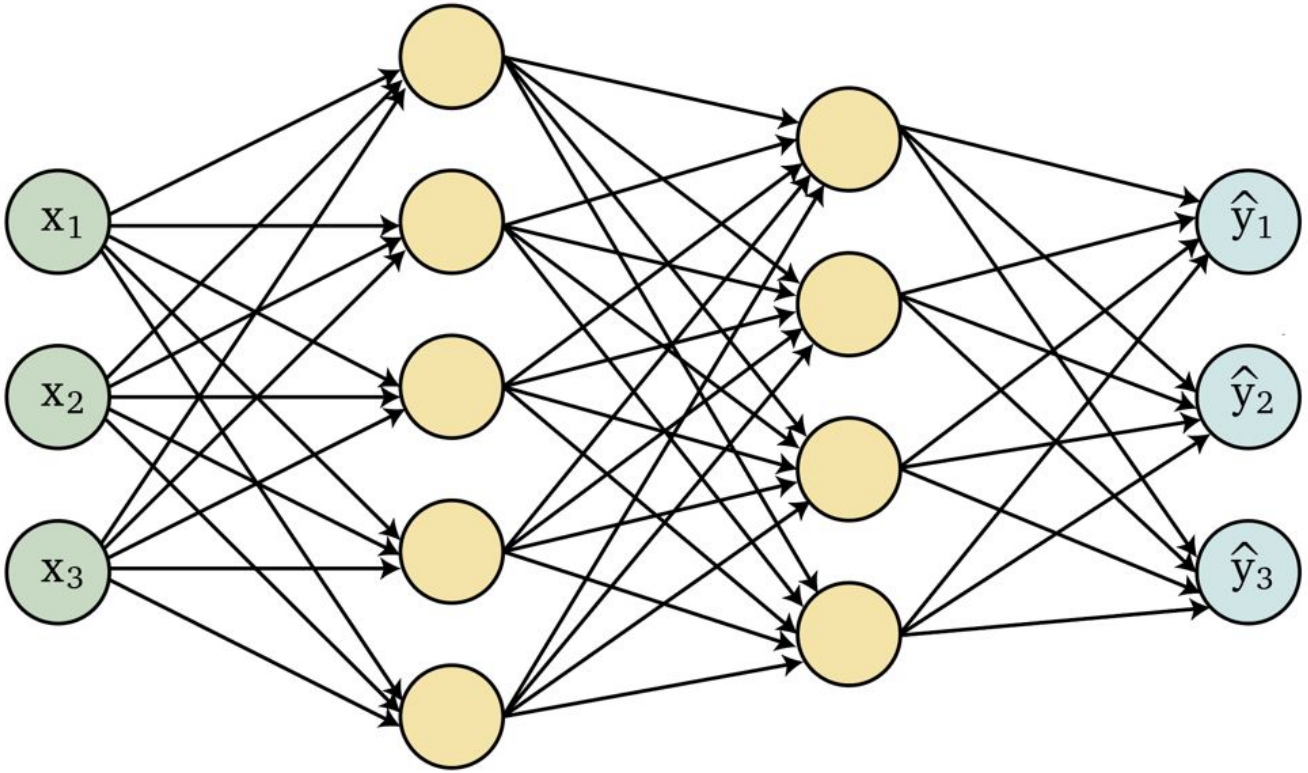
Perceptron



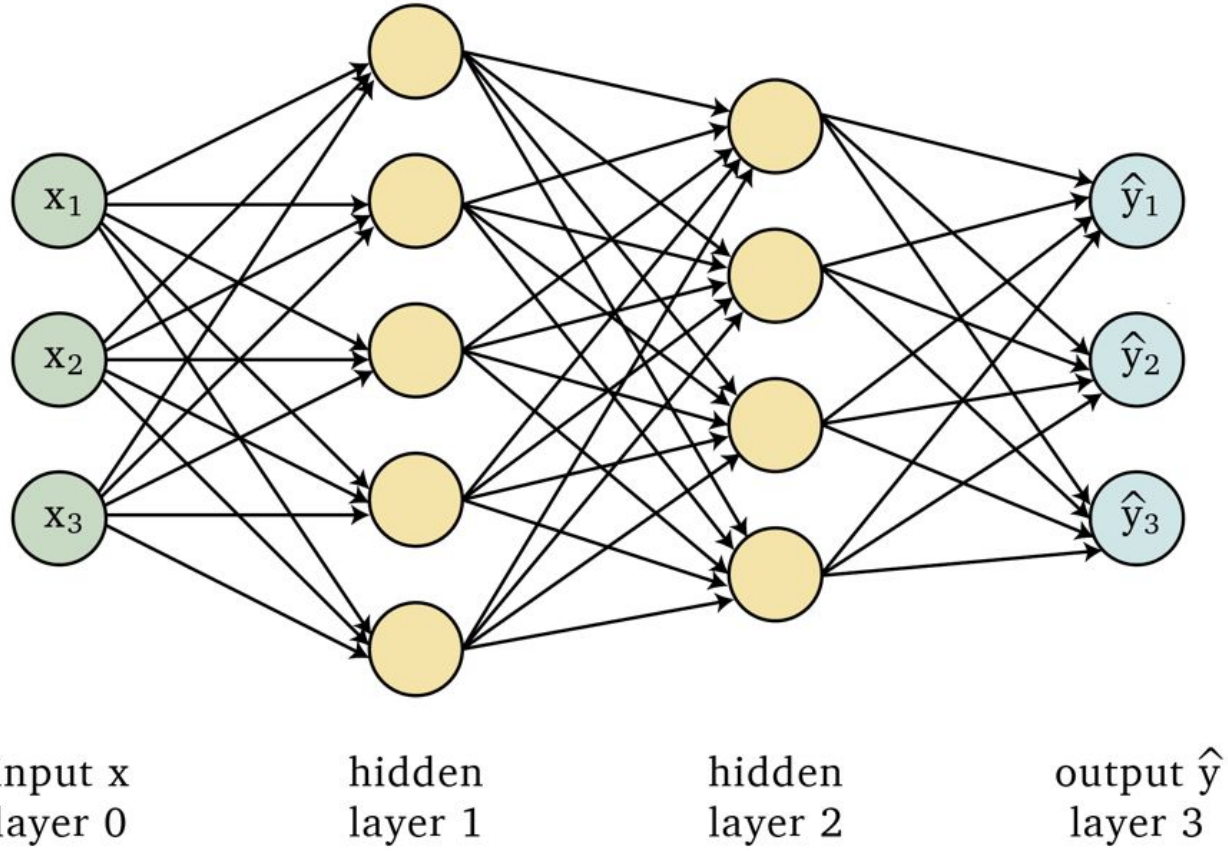
Fully Connected Neural Network



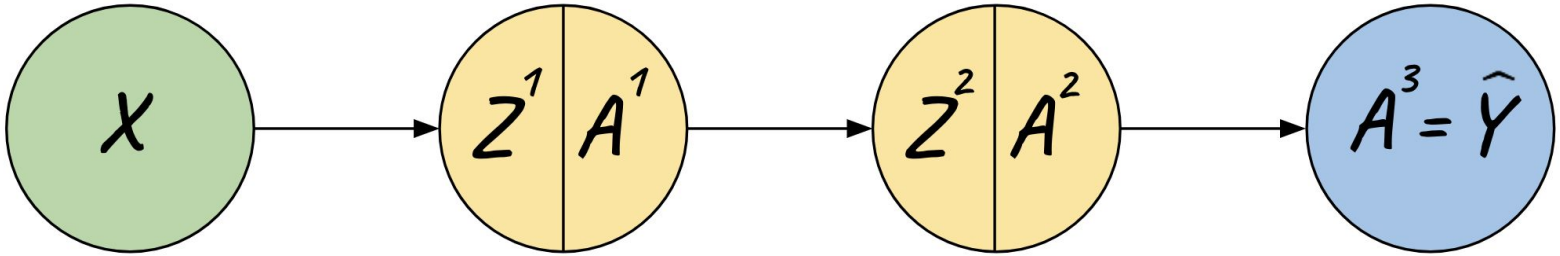
Neural Network



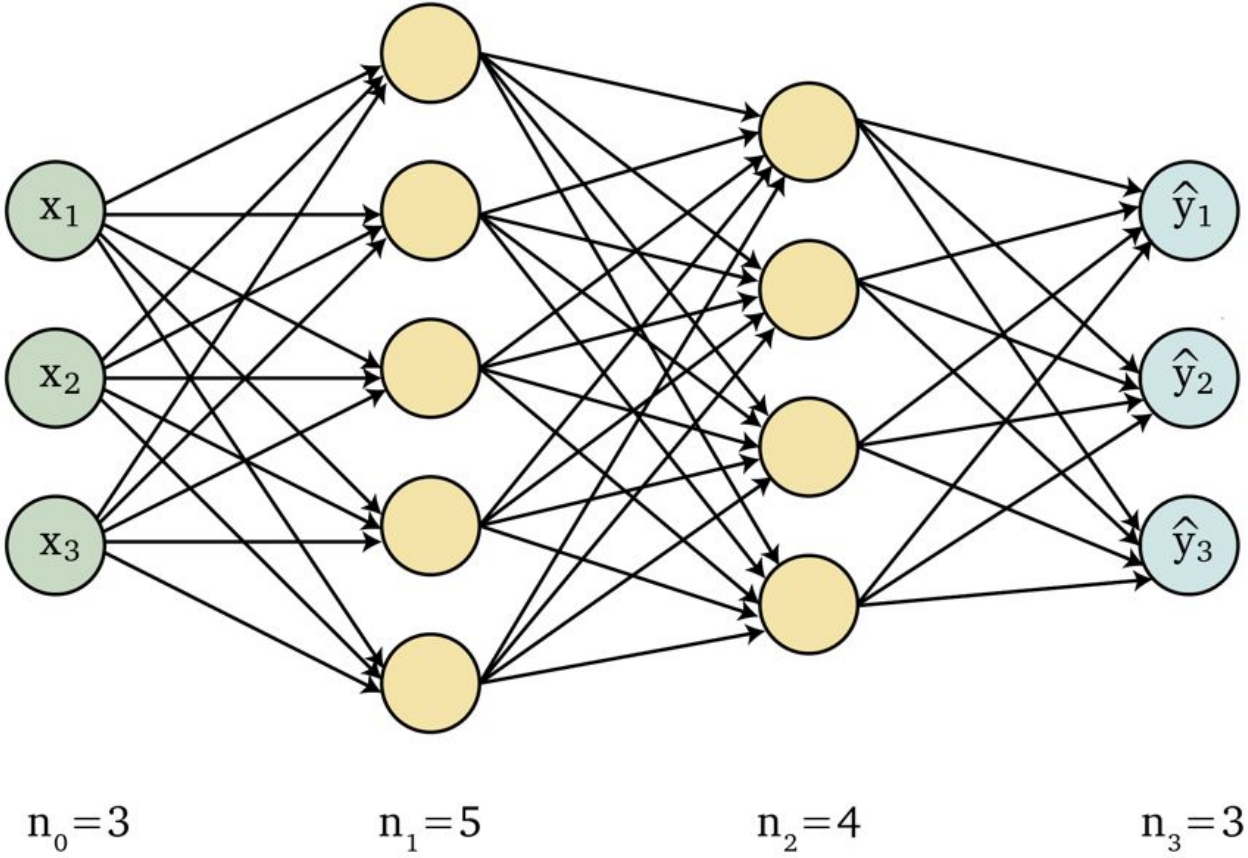
Neural Network



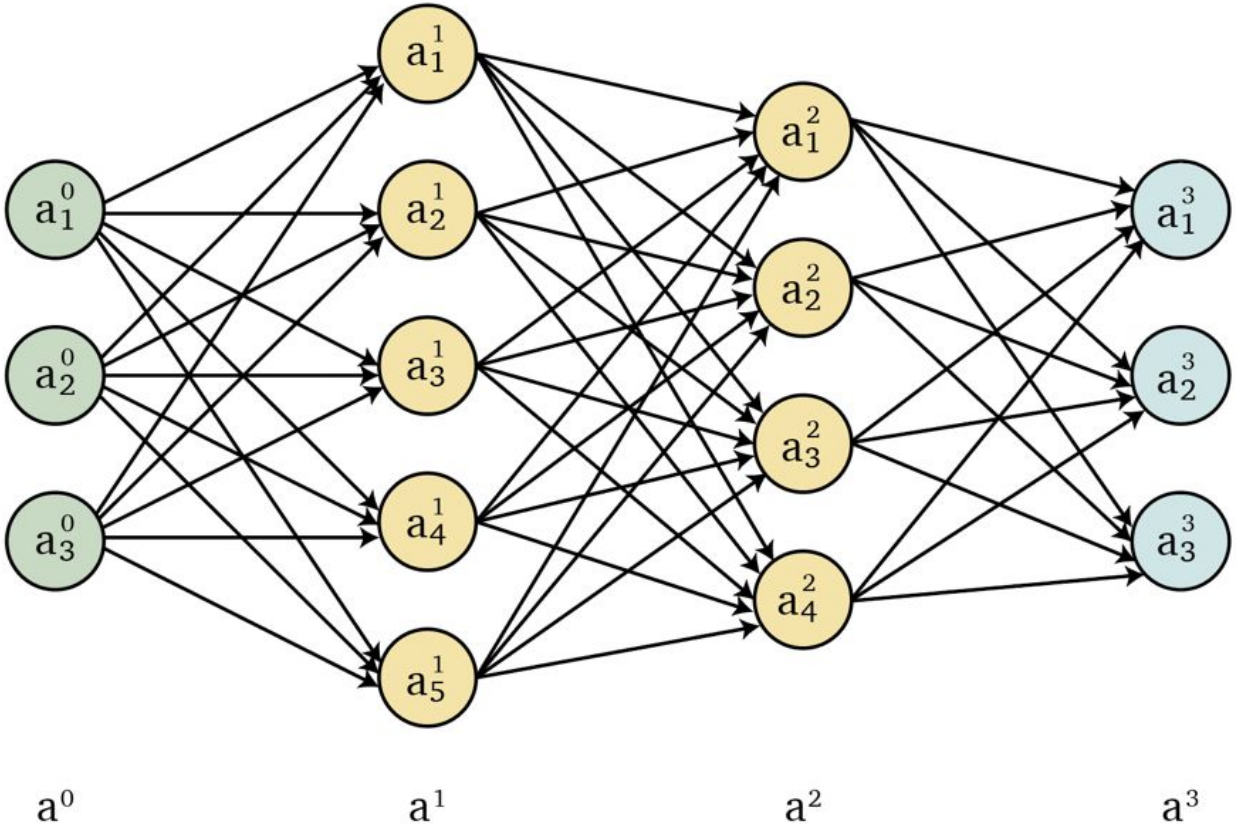
Markov Chain



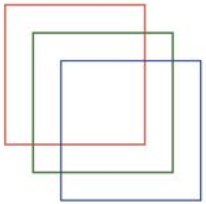
Neural Network



Neural Network

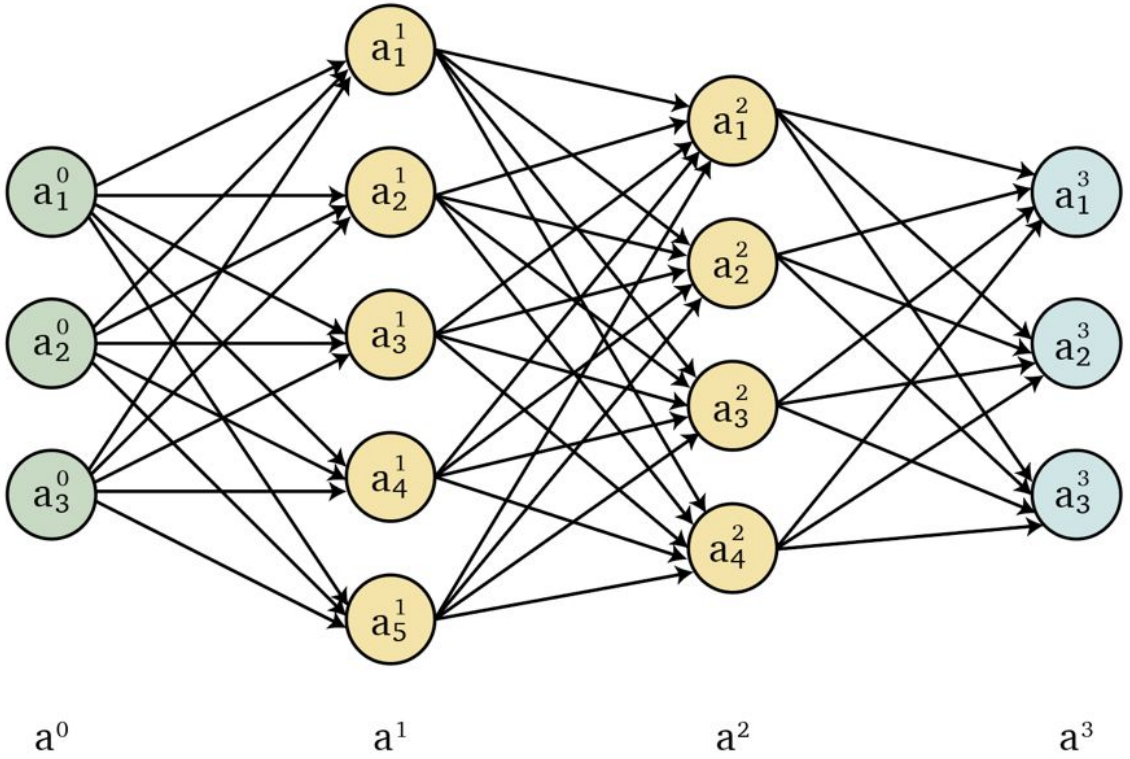


Neural Network

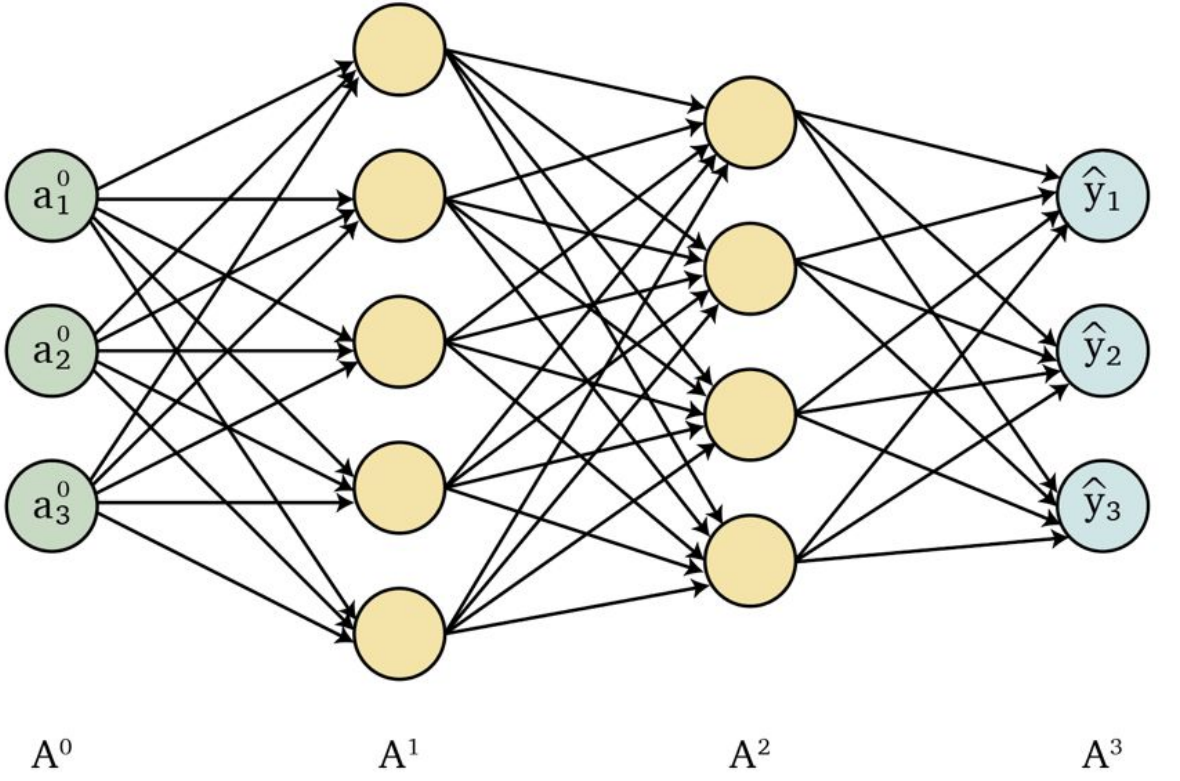


wxh image

$$a^{\circ} = \begin{bmatrix} a^{\circ}_{1} \\ \vdots \\ a^{\circ}_{wxh} \\ a^{\circ}_{wxh+1} \\ \vdots \\ a^{\circ}_{2(wxh)} \\ a^{\circ}_{2(wxh)+1} \\ \vdots \\ a^{\circ}_{3(wxh)} \end{bmatrix}$$



Neural Network

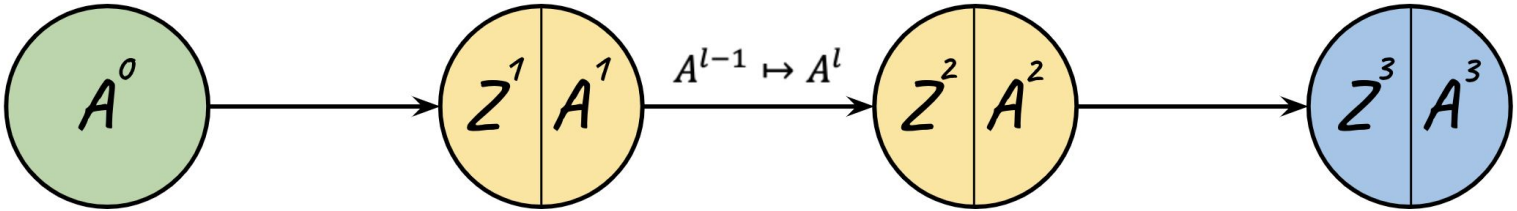


$$A^l_{n_l \times m} = [a^{l1} \cdots a^{lm}]$$

Forward Propagation

$$f(x) = f^3\left(f^2\left(f^1(x)\right)\right)$$

composition of functions



$$A^l = f^l\left(A^{l-1}\right)$$

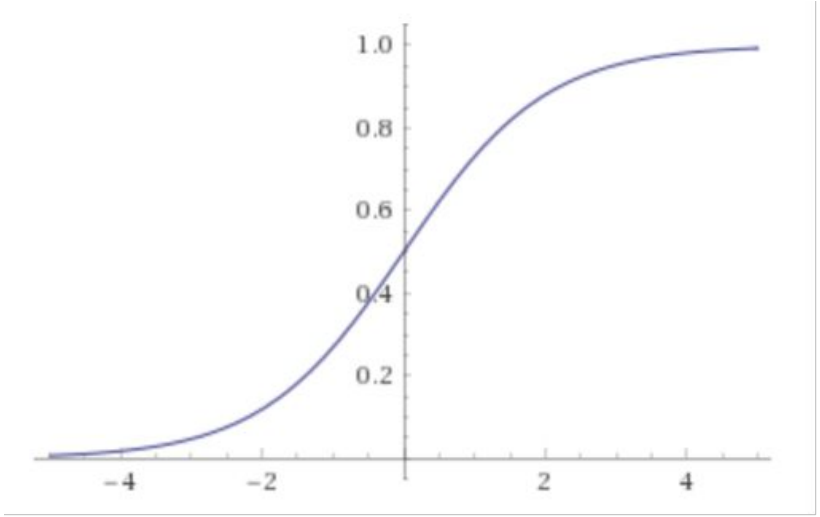
$$Z^l = W^l A^{l-1} \qquad A^l = g^l\left(Z^l\right)$$

linear non-linear

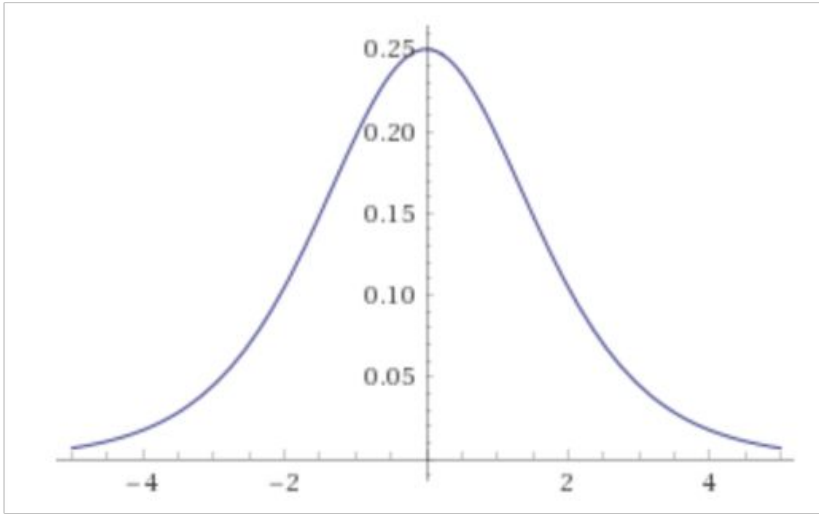
$$A^3 = g^3\left(W^3 g^2\left(W^2 g^1\left(W^1 A^0\right)\right)\right)$$

if g is identity then f is linear in x

Sigmoid

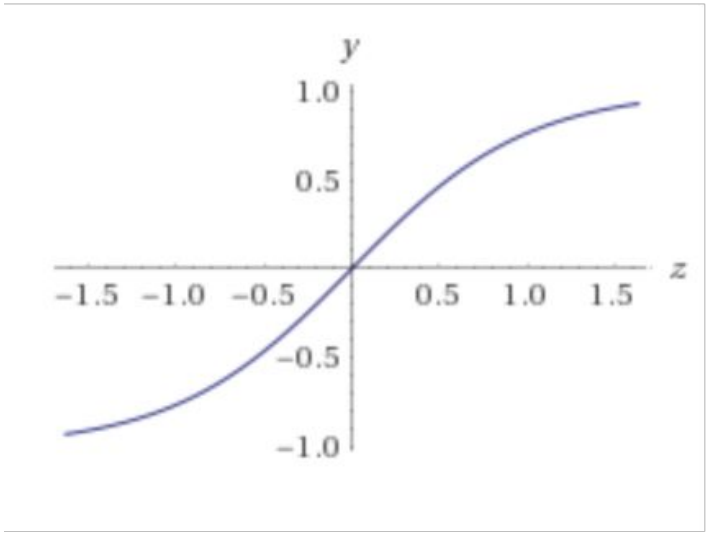


$$g(z) = \frac{1}{1 + e^{-z}}$$

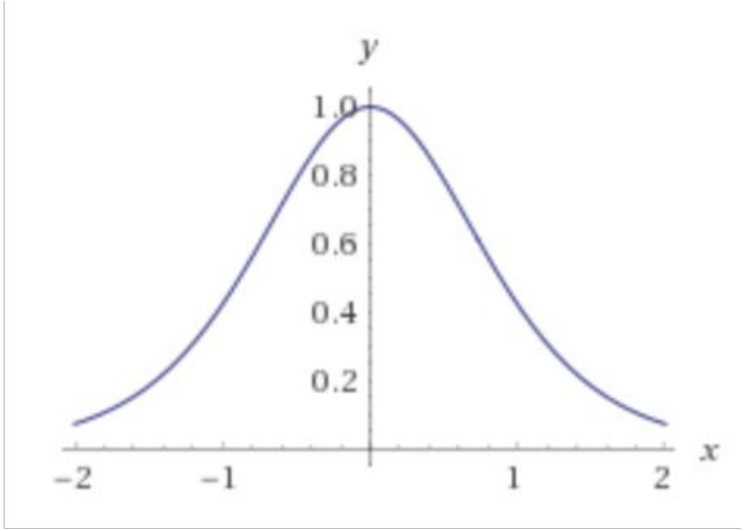


$$g'(z) = g(z)(1 - g(z))$$

Hyperbolic Tangent

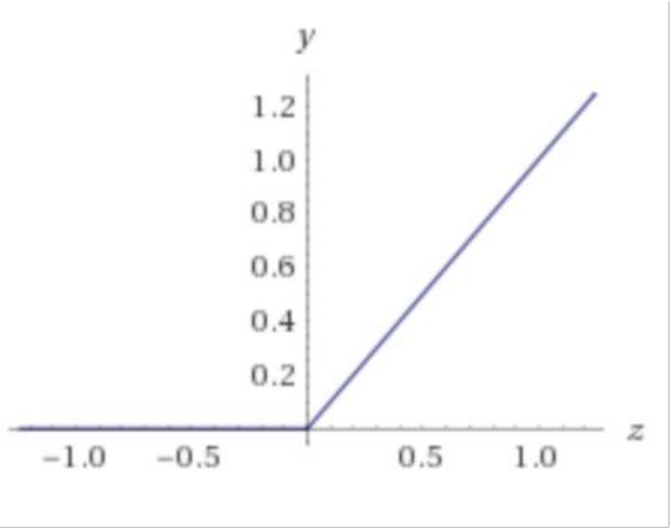


$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

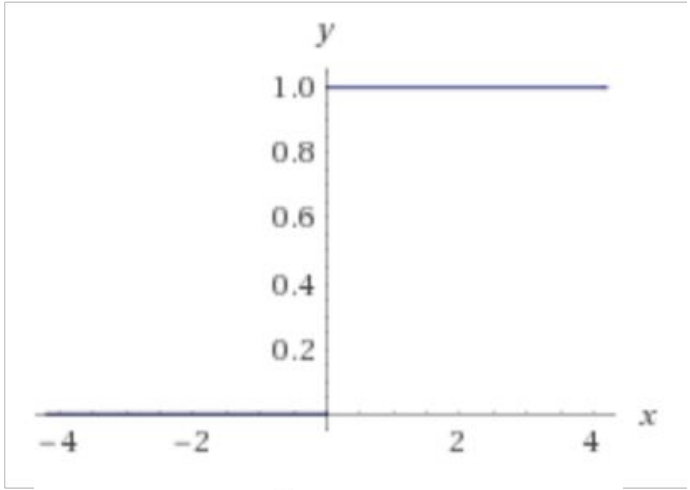


$$g'^{(z)} = \frac{4}{(e^{-z} + e^z)^2}$$

Rectified Linear Unit (ReLU)

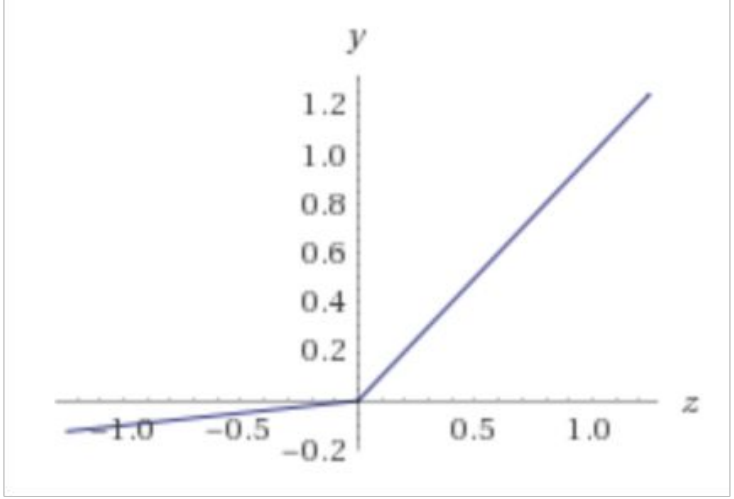


$$g(z) = z_+ = \max(0, z)$$



$$g'(z) = \begin{cases} 0, & \text{if } z < 0 \\ 1, & \text{if } z > 0 \end{cases}$$

Leaky ReLU



$$g(z) = z_+ - \alpha z_- = \max(0, z) - \alpha \max(0, -z)$$

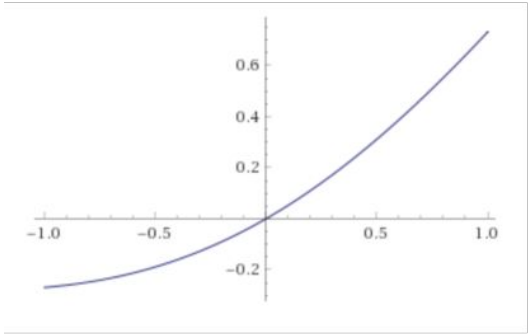
$$g'(z) = \begin{cases} \alpha, & \text{if } z < 0 \\ 1, & \text{if } z > 0 \end{cases}$$

Activation Functions in Python

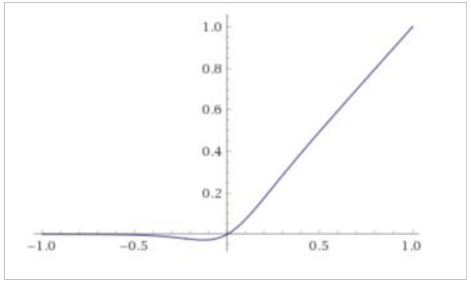
```python
import numpy as np

def sigmoid(Z):
    A = 1/(1 + np.exp(-Z))
    return A

def tanh(Z):
    return np.tanh(Z)

def relu(Z):
    A = np.maximum(0, Z)
    return A

def leaky_relu(Z, alpha):
    A = relu(Z) - alpha * relu(-Z)
    return A
```

Swish

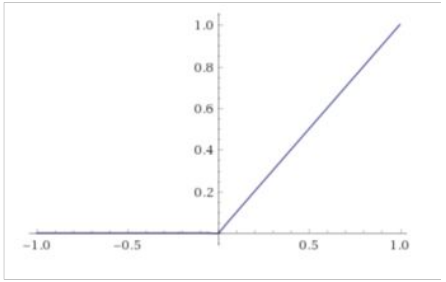
$\beta = 1$



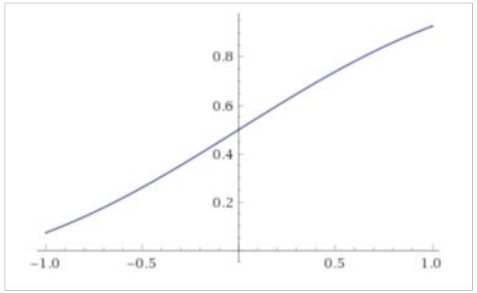
$\beta = 10$

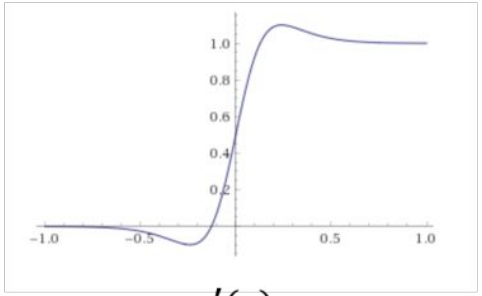


$\beta = 100$

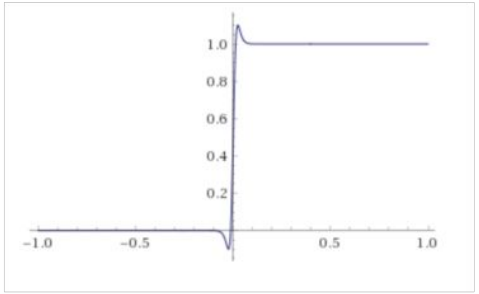


$$g(z) = z\sigma(\beta z)$$





$g'(z)$



Swish in Python

```
def sigmoid(Z):
    return 1/(1 + np.exp(-Z))

def swish(X, beta):
    return X * sigmoid(beta * X)
```

Softmax

Non-linear activation function used in last layer for multi-class classification

Generalization of sigmoid from scalars to vectors

Mapping $g: R^C \mapsto [0,1]^C$

$$g^L(z^L)_i = \frac{e^{z_i^L}}{\sum_{c=1}^{C} e^{z_c^L}} \qquad \sum_{c=1}^{C} g(z)_c = 1$$

```
def softmax(X):
    return np.exp(X) / np.sum(np.exp(X), axis = 0)
```

Loss Functions

Compare predicted output with ground truth label

For a single example $\mathcal{L}(y, f(x, W))$

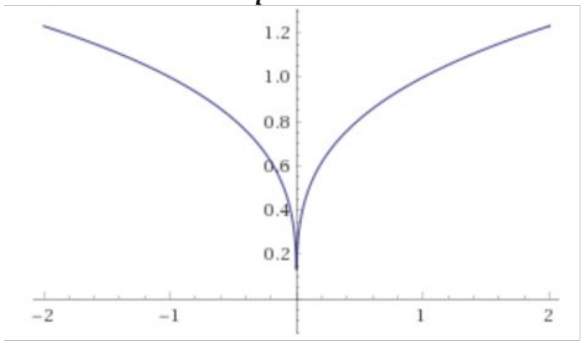
Average loss over all examples $\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(y^{(i)}, \hat{y}^{(i)})$ $\hat{y}^{(i)} = f(x^{(i)}, W)$

Optimization $\underset{W}{\text{minimize}}\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\left(y^{(i)}, f(x^{(i)}, W)\right)$

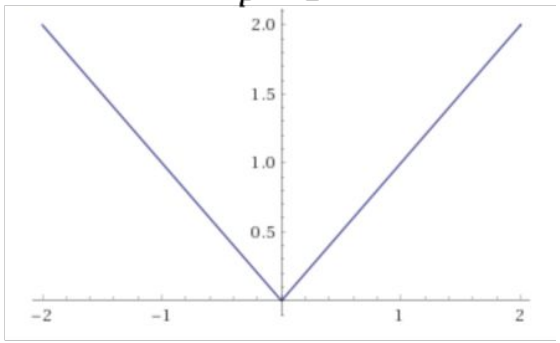
Regularization term may be added to prefer simple models, avoid overfitting.

Loss Functions

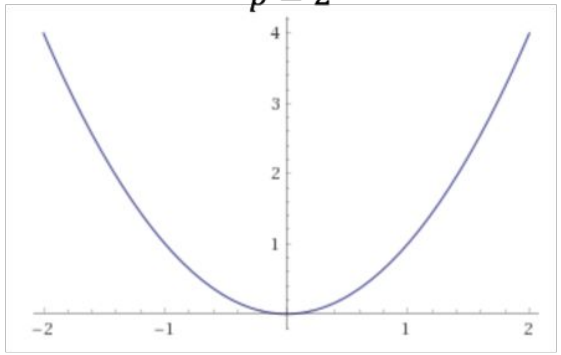
$p = 0.3$



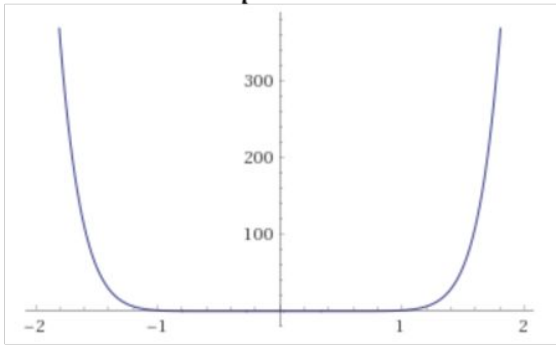
$p = 1$



$p = 2$



$p = 10$



$$\mathcal{L}(y, a) = |y - a|^p$$

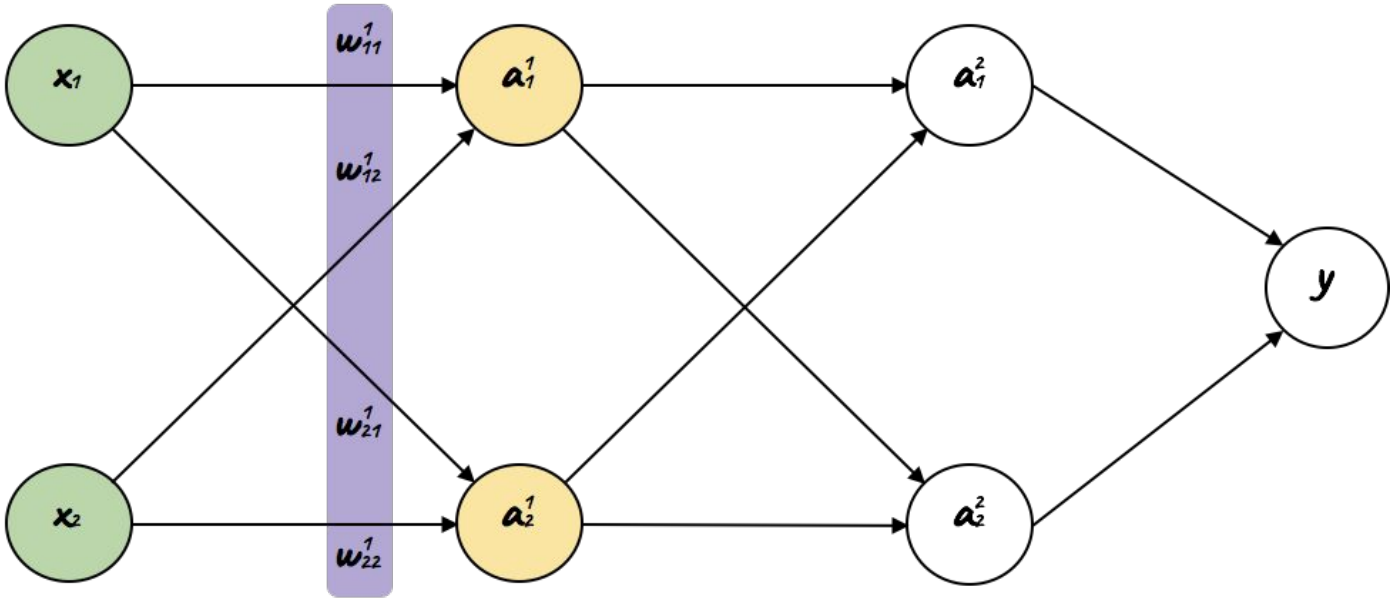
Loss Functions

$$\mathcal{L}(y, a) = -y\log(a) - (1 - y)\log(1 - a)$$

Cost Function

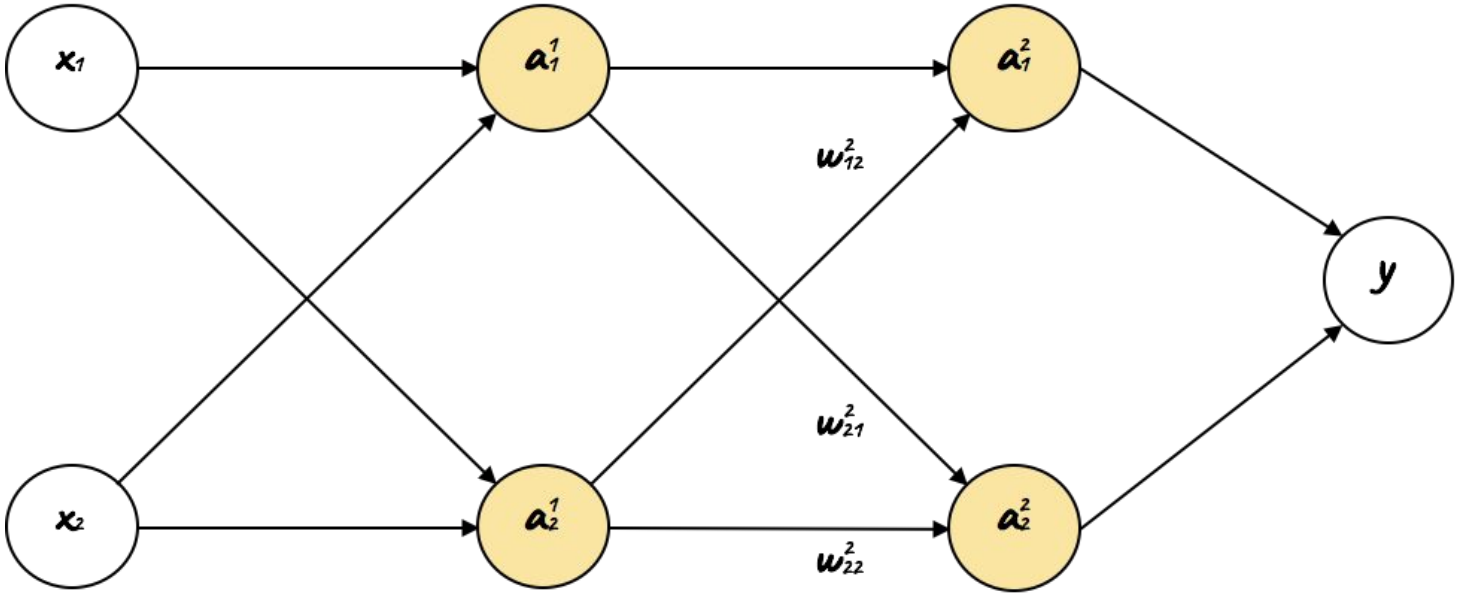
```
def costFunction(AL, y):
    m = y.shape[0]
    AL_softmax = softmax(AL)
    correct_label_prob = AL_softmax[y, range(m)]
    cost = - np.sum(np.log(correct_label_prob)) / m
    return cost
```

Forward Propagation Example



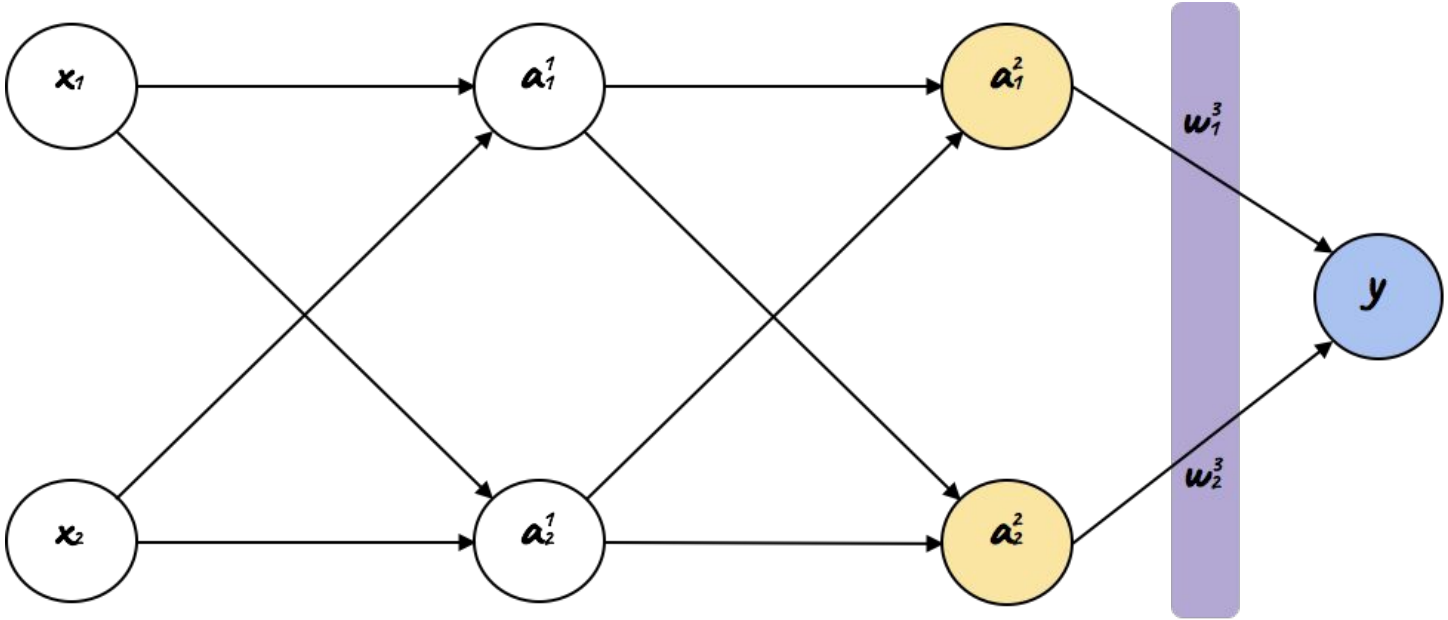
$$z^1 = \begin{pmatrix} w_{11}^1 & w_{12}^1 \\ w_{21}^1 & w_{22}^1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} w_{11}^1 x_1 & w_{12}^1 x_2 \\ w_{21}^1 x_1 & w_{22}^1 x_2 \end{pmatrix} = \begin{pmatrix} z_1^1 \\ z_2^1 \end{pmatrix} \qquad a^1 = \begin{pmatrix} \sigma(z_1^1) \\ \sigma(z_2^1) \end{pmatrix} = \begin{pmatrix} a_1^1 \\ a_2^1 \end{pmatrix}$$

Forward Propagation Example



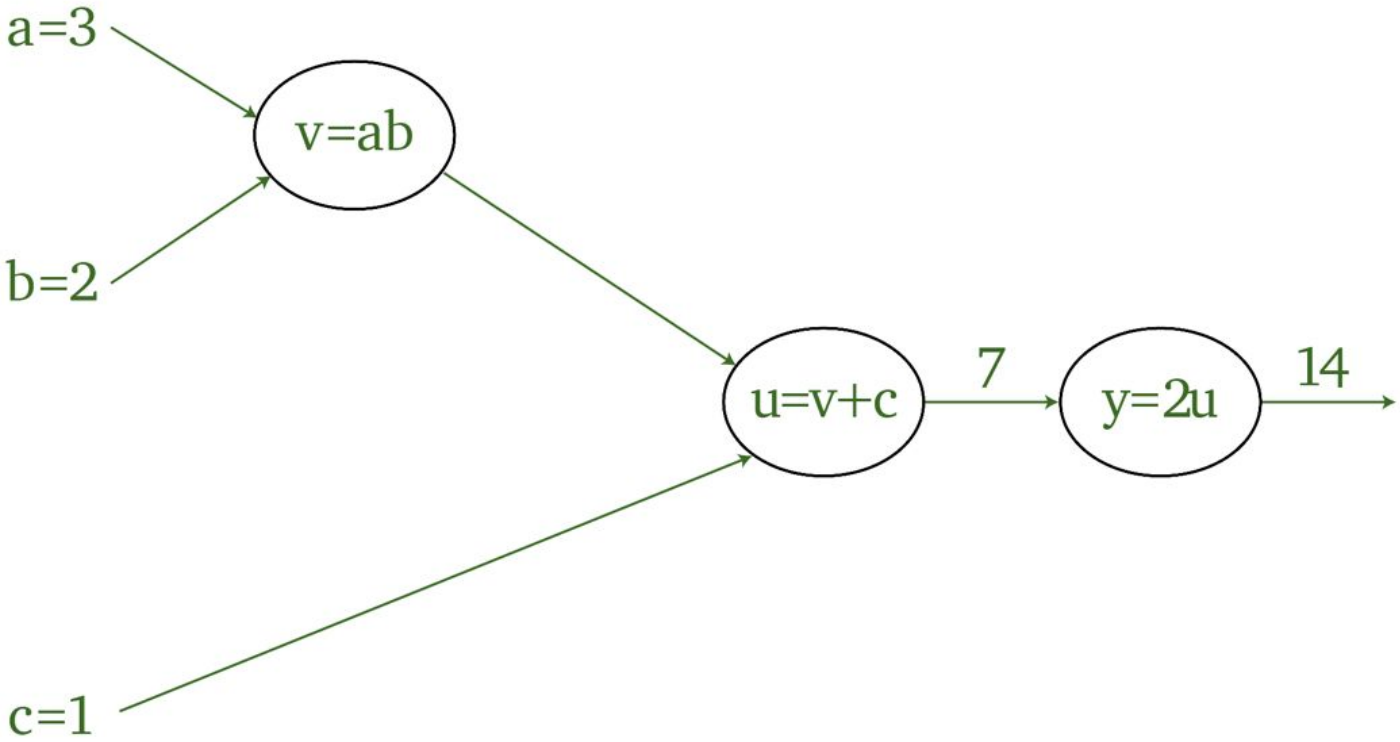
$$z^2 = \begin{pmatrix} w_{11}^2 & w_{12}^2 \\ w_{21}^2 & w_{22}^2 \end{pmatrix} \begin{pmatrix} a_1^1 \\ a_2^1 \end{pmatrix} = \begin{pmatrix} w_{11}^2 a_1^1 & w_{12}^2 a_2^1 \\ w_{21}^2 a_1^1 & w_{22}^2 a_2^1 \end{pmatrix} = \begin{pmatrix} z_1^2 \\ z_2^2 \end{pmatrix} \qquad a^2 = \begin{pmatrix} \sigma(z_1^2) \\ \sigma(z_2^2) \end{pmatrix} = \begin{pmatrix} a_1^2 \\ a_2^2 \end{pmatrix}$$

Forward Propagation Example

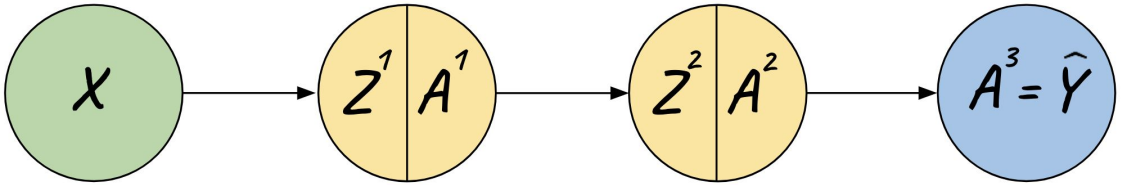


$$y = \begin{pmatrix} w_1^3 & w_2^3 \end{pmatrix} \begin{pmatrix} a_1^2 \\ a_2^2 \end{pmatrix} = w_1^3 a_1^2 + w_2^3 a_2^2$$

Computation Graph



Forward Propagation

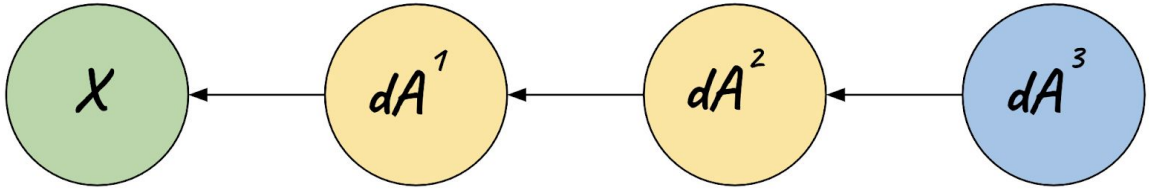


$$A^{l-1} \mapsto A^l$$

$$Z^l = W^l A^{l-1} \qquad A^l = g^l(Z^l)$$

$$\hat{Y} = g^3\left(W^3 g^2\left(W^2 g^1(W^1 X)\right)\right)$$

Backpropagation

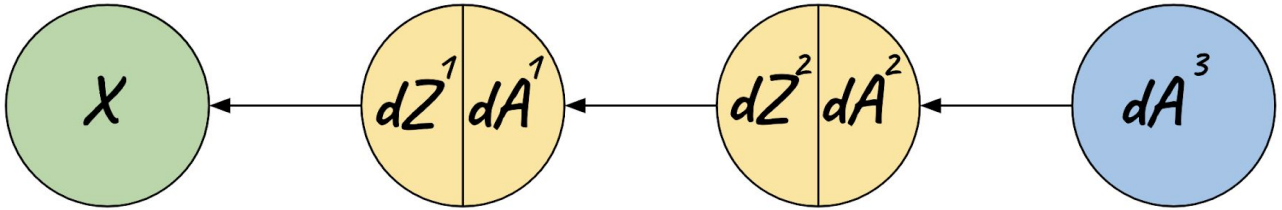


Map derivatives from layer l back to layer $l-1$ with respect to **activations** and **weights**

$$dA^l \mapsto dA^{l-1}$$

$$dA^l := \frac{d\mathcal{L}}{dA^l} \qquad dZ^l := \frac{d\mathcal{L}}{dZ^l} \qquad dW^l := \frac{d\mathcal{L}}{dW^l}$$

Backpropagation



Map derivatives from layer l back to layer $l-1$ with respect to **activations**

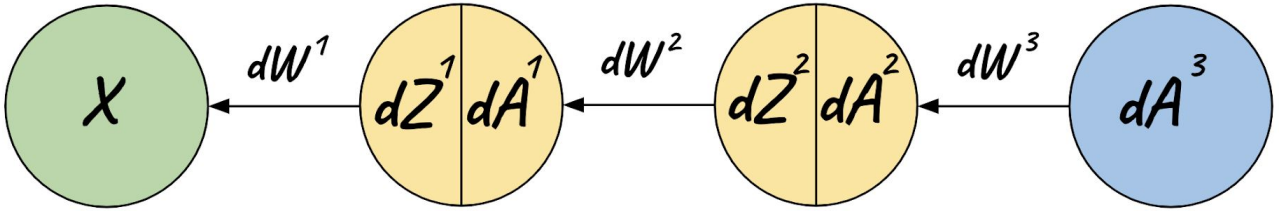
$$dA^{l-1} = \left(W^{l}\right)^{T} dZ^{l}$$

From chain rule of differentiation

$$\frac{d\mathcal{L}}{dZ^{l}} = \frac{d\mathcal{L}}{dA^{l}} \frac{dA^{l}}{dZ^{l}} = \frac{d\mathcal{L}}{dA^{l}} \cdot g^{l\prime}\left(Z^{l}\right)$$

$$dZ^{l} = dA^{l} \cdot g^{l\prime}\left(Z^{l}\right)$$

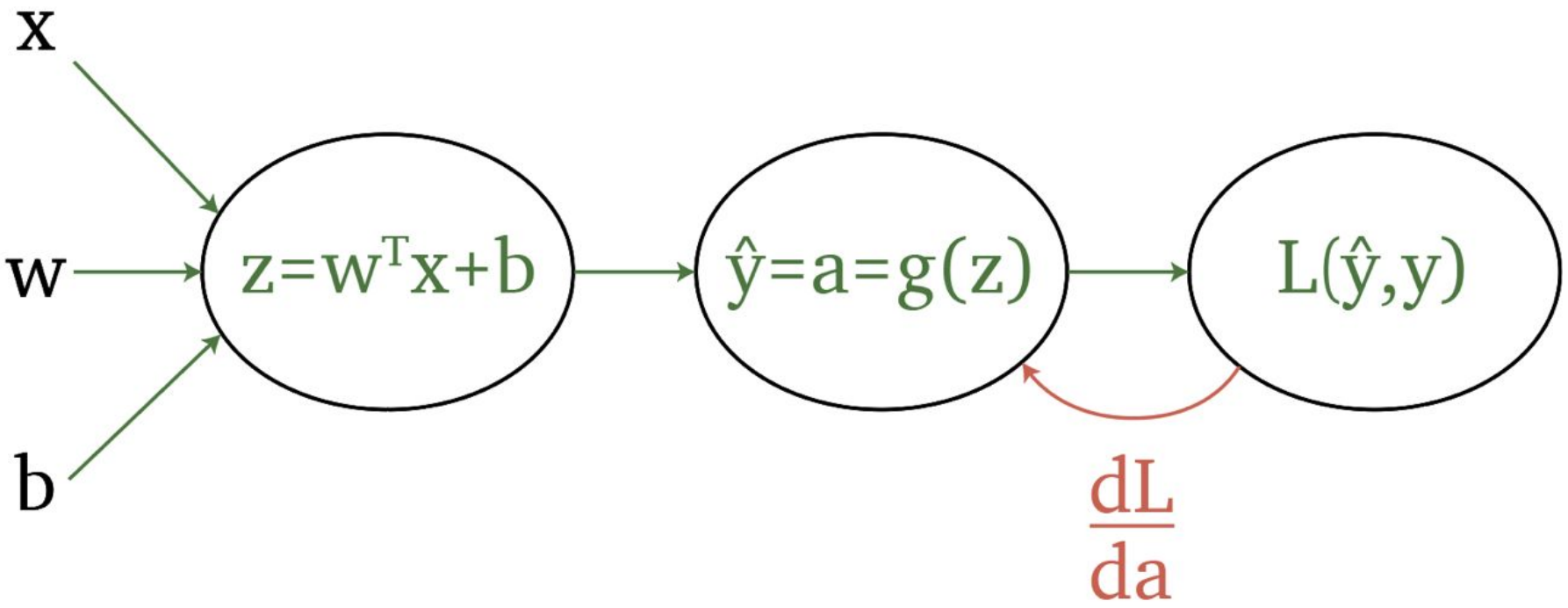
Backpropagation



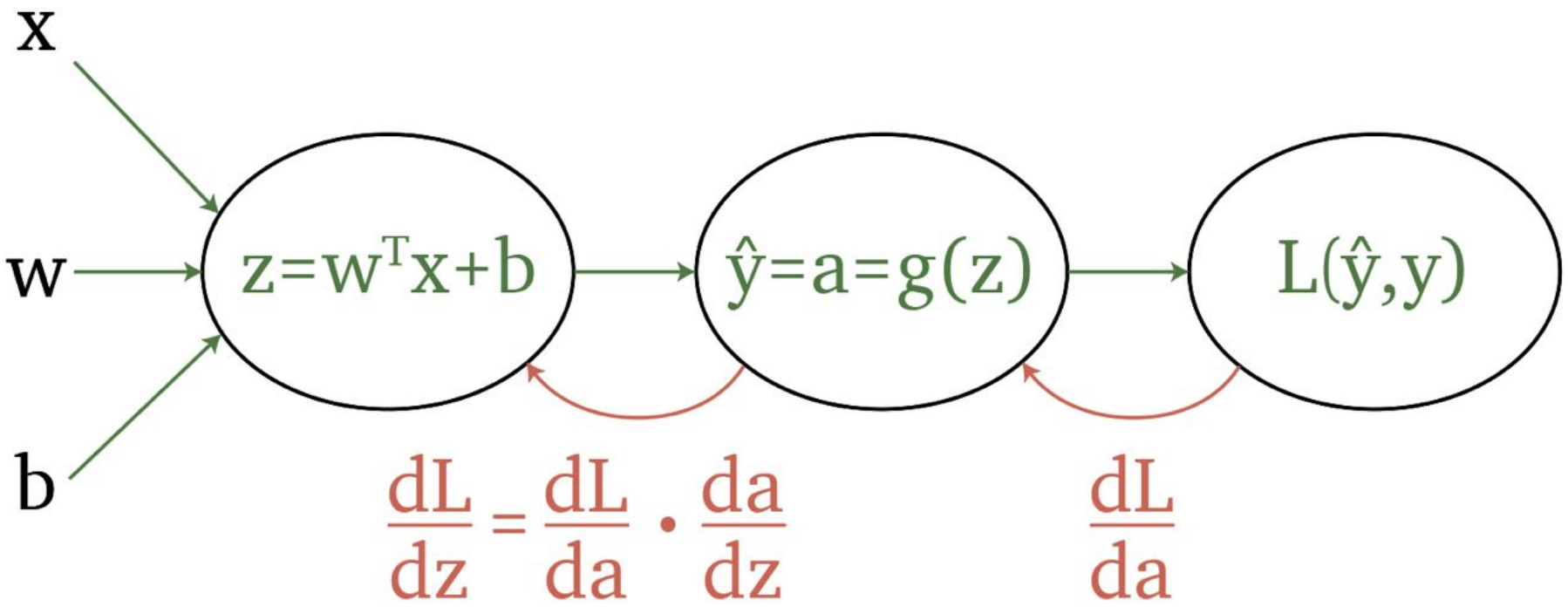
Map derivatives from layer l back to layer $l-1$ with respect to **weights**

$$dW^l = dZ^l \left(A^{l-1}\right)^T$$

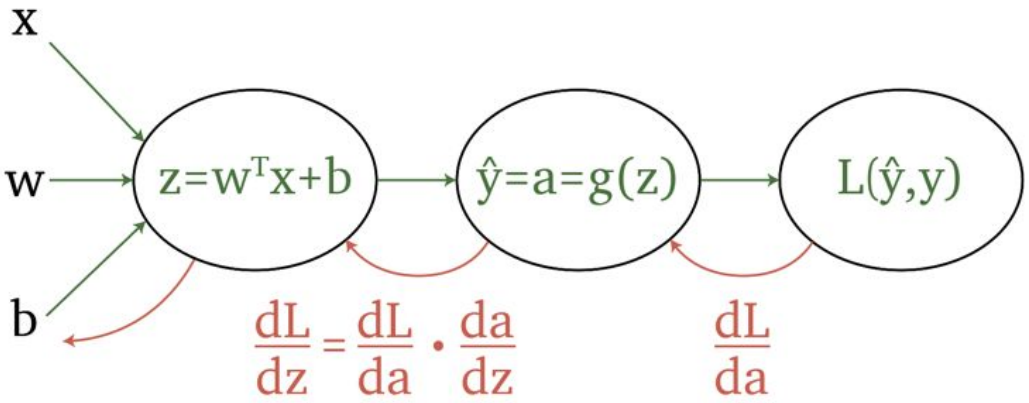
Logistic Regression



Logistic Regression



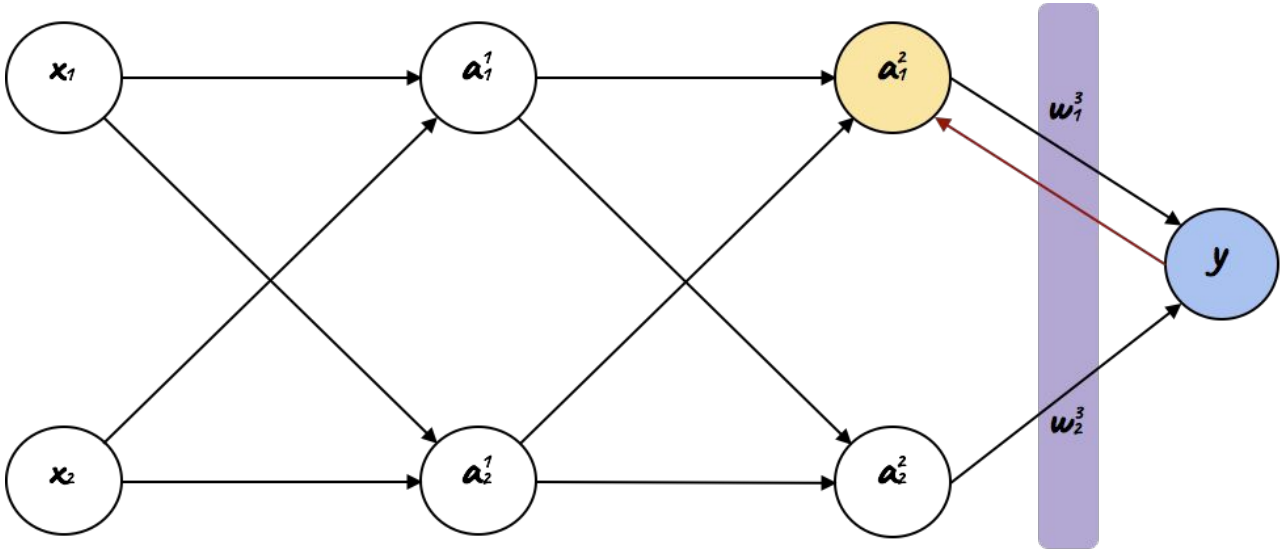
Logistic Regression



$$\frac{dL}{dw} = \frac{dL}{da} \cdot \frac{da}{dz} \cdot \frac{dz}{dw} = \frac{dL}{dz} \cdot x$$

$$\frac{dL}{db} = \frac{dL}{da} \cdot \frac{da}{dz} \cdot \frac{dz}{db} = \frac{dL}{dz}$$

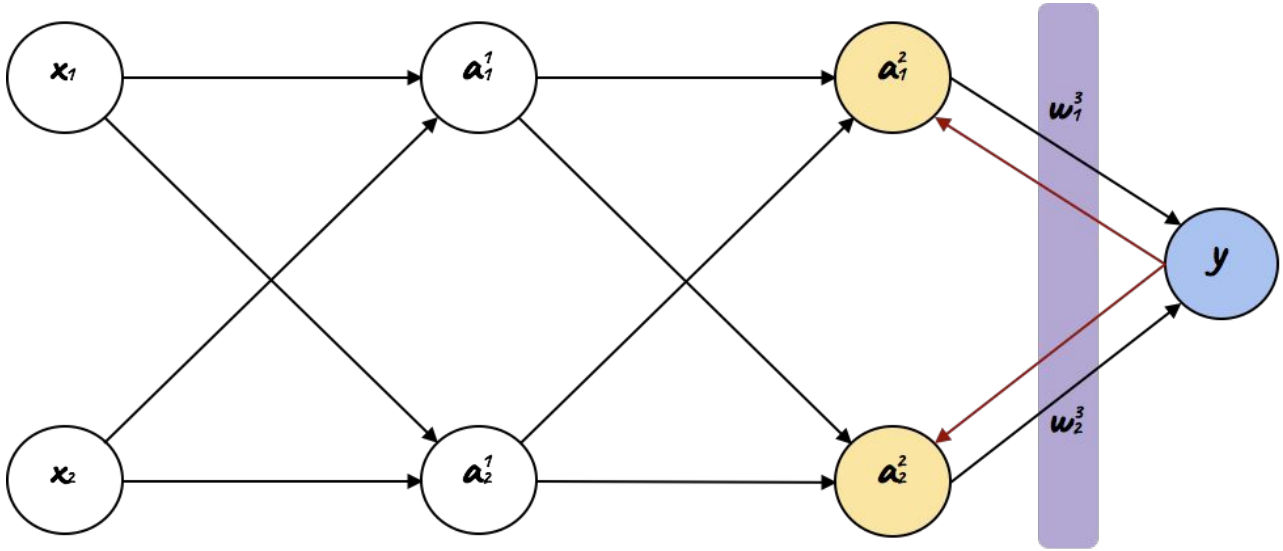
Backpropagation Example



$$y = \begin{pmatrix} w_1^3 & w_2^3 \end{pmatrix} \begin{pmatrix} a_1^2 \\ a_2^2 \end{pmatrix} = w_1^3 a_1^2 + w_2^3 a_2^2$$

$$\frac{\partial y}{\partial z_1^2} = w_1^3 \frac{\partial a_1^2}{\partial z_1^2} = w_1^3 \sigma(z_1^2)\left(1 - \sigma(z_1^2)\right) = w_1^3 a_1^2 (1 - a_1^2)$$

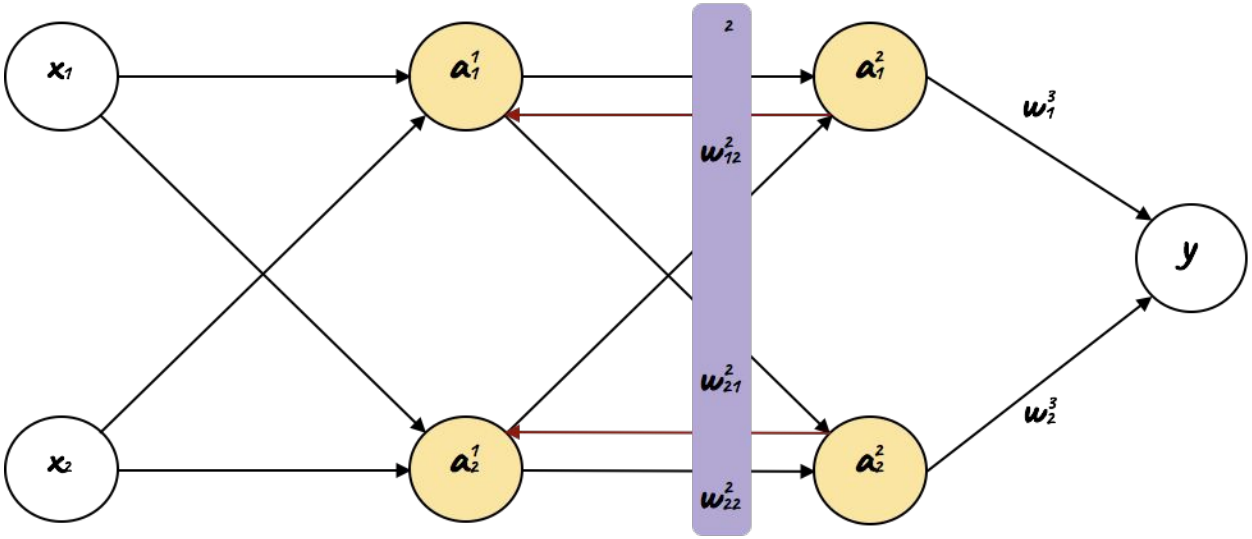
Backpropagation Example



$$i = 1,2 \qquad \frac{\partial y}{\partial z_i^2} = w_i^3 \frac{\partial a_i^2}{\partial z_i^2} = w_i^3 \sigma(z_i^2)\left(1 - \sigma(z_i^2)\right) = w_i^3 a_i^2 (1 - a_i^2)$$

$$\frac{\partial y}{\partial z^2} = \frac{\partial y}{\partial a^2}\frac{\partial a^2}{\partial z^2} = (w_1^3 \quad w_2^3)^T a^2 (1 - a^2)$$

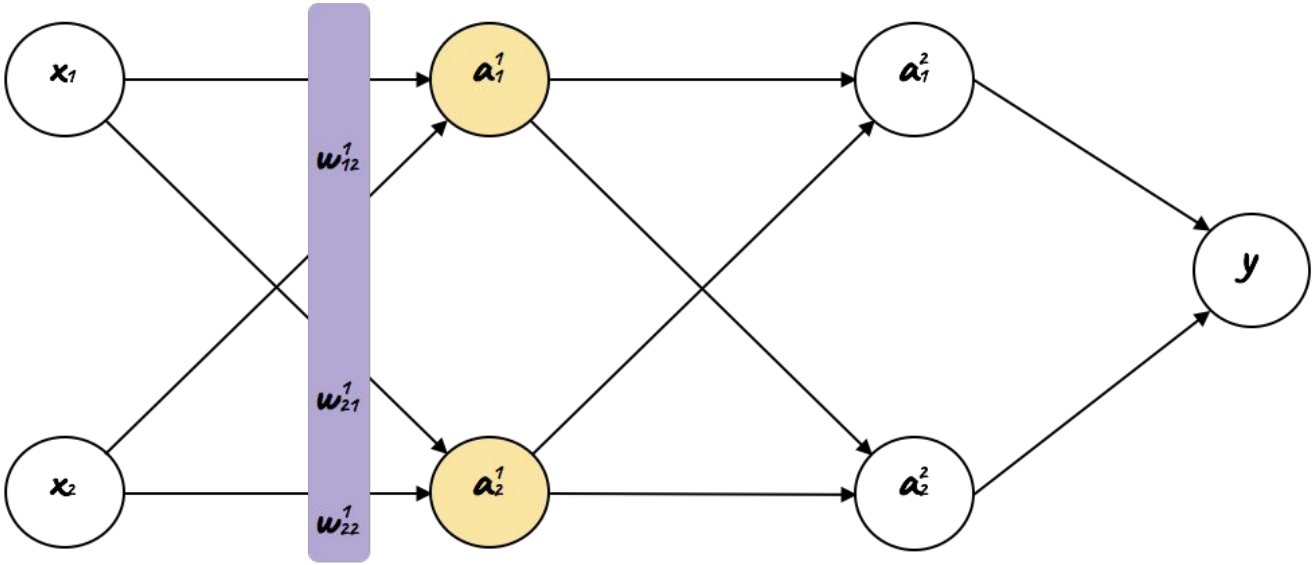
Backpropagation Example



$$\frac{\partial y}{\partial z^2} = \frac{\partial y}{\partial a^2}\frac{\partial a^2}{\partial z^2} = (w_1^3 \quad w_2^3)^T a^2(1-a^2)$$

$$\frac{\partial y}{\partial z^1} = \frac{\partial y}{\partial z^2}\frac{\partial z^2}{\partial z^1} = \frac{\partial y}{\partial z^2}\frac{\partial z^2}{\partial a^1}\frac{\partial a^1}{\partial z^1} = \begin{pmatrix} w_{11}^2 & w_{12}^2 \\ w_{21}^2 & w_{22}^2 \end{pmatrix}\frac{\partial y}{\partial z^2}a^1(1-a^1)$$

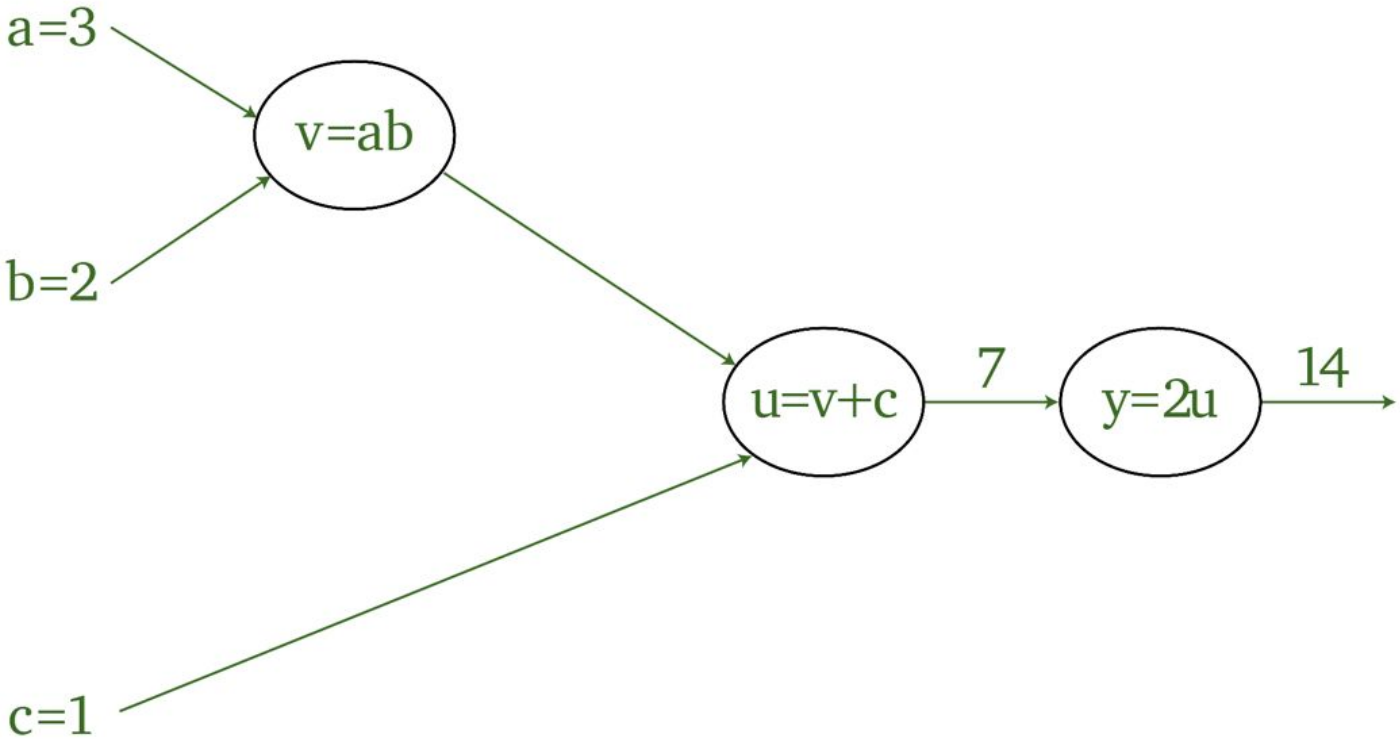
Backpropagation Example



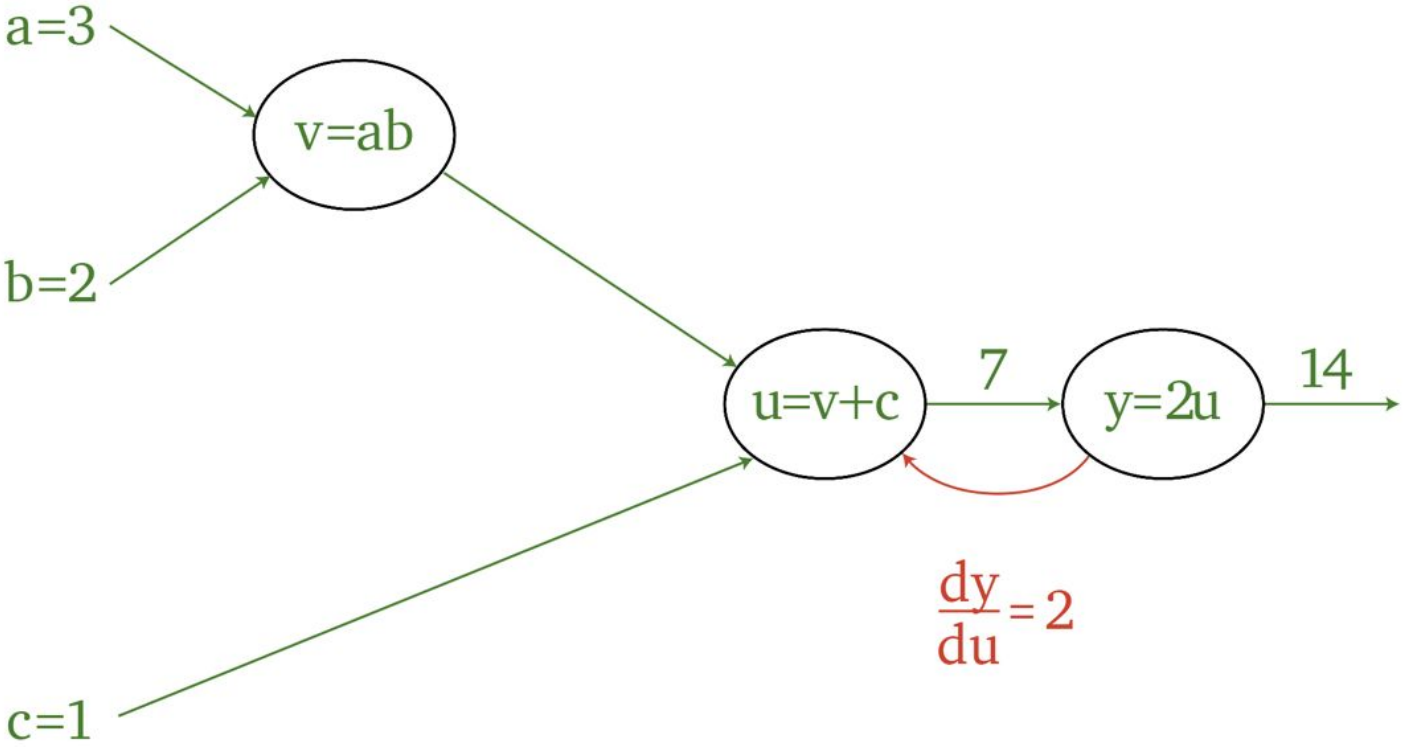
$$\frac{\partial y}{\partial z^1} = \frac{\partial y}{\partial z^2}\frac{\partial z^2}{\partial z^1} = \frac{\partial y}{\partial z^2}\frac{\partial z^2}{\partial a^1}\frac{\partial a^1}{\partial z^1} = \begin{pmatrix} w^2_{11} & w^2_{12} \\ w^2_{21} & w^2_{22} \end{pmatrix}\frac{\partial y}{\partial z^2}a^1(1-a^1)$$

$$\frac{\partial y}{\partial w^1_{11}} = \frac{\partial y}{\partial z^1}\frac{\partial z^1}{\partial w^1_{11}} = \frac{\partial y}{\partial z^1}\begin{pmatrix} x_1 \\ 0 \end{pmatrix} \quad z^1 = \begin{pmatrix} w^1_{11} & w^1_{12} \\ w^1_{21} & w^1_{22} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} w^1_{11}x_1 & w^1_{12}x_2 \\ w^1_{21}x_1 & w^1_{22}x_2 \end{pmatrix} = \begin{pmatrix} z^1_1 \\ z^1_2 \end{pmatrix}$$

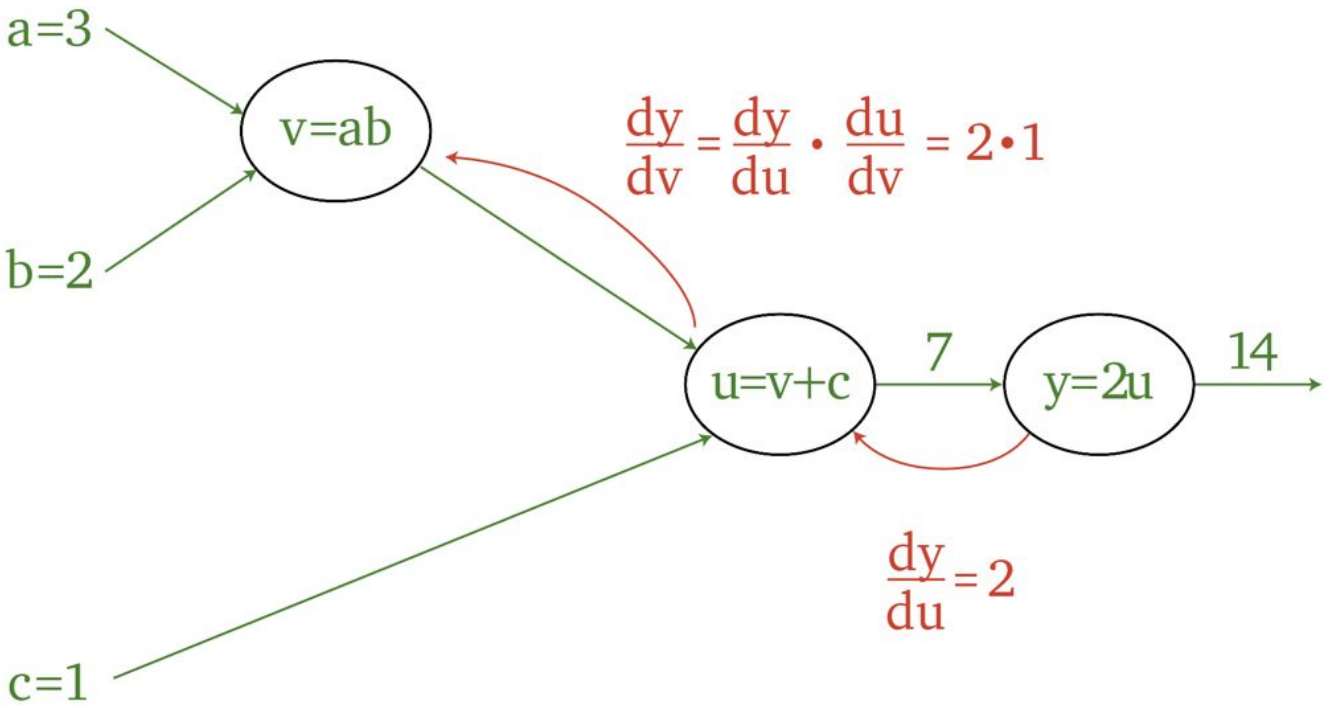
Computation Graph



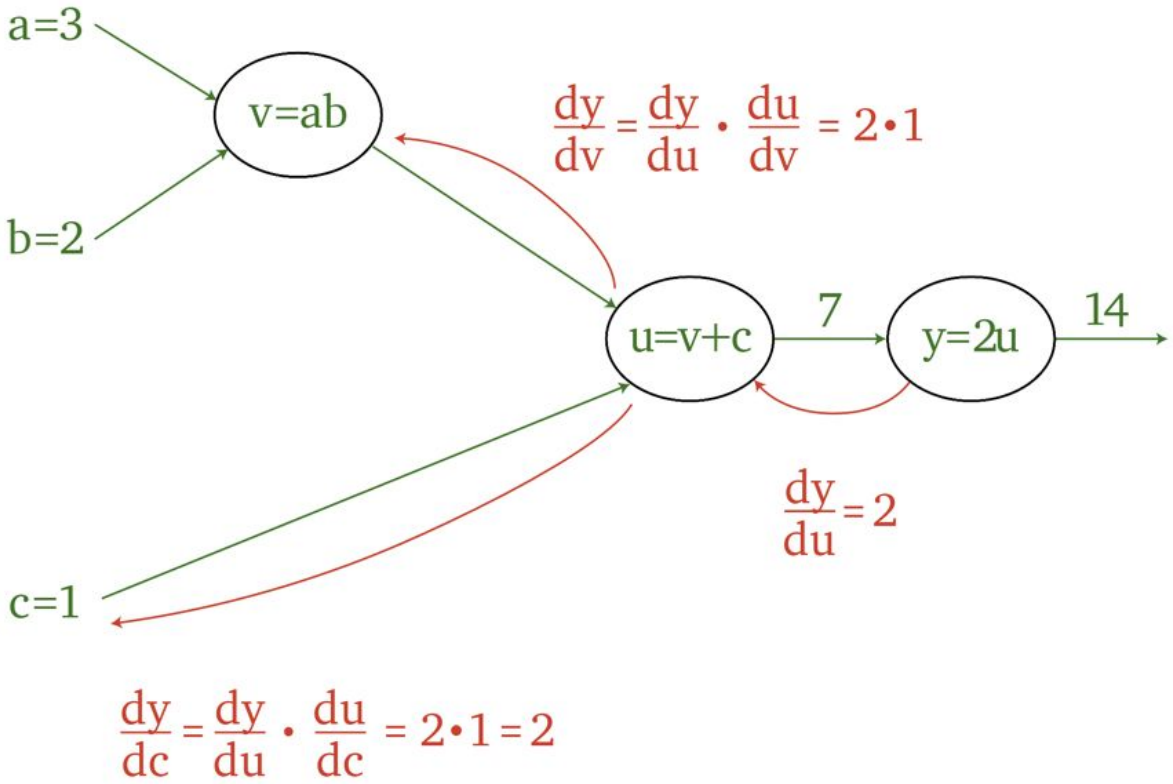
Computation Graph



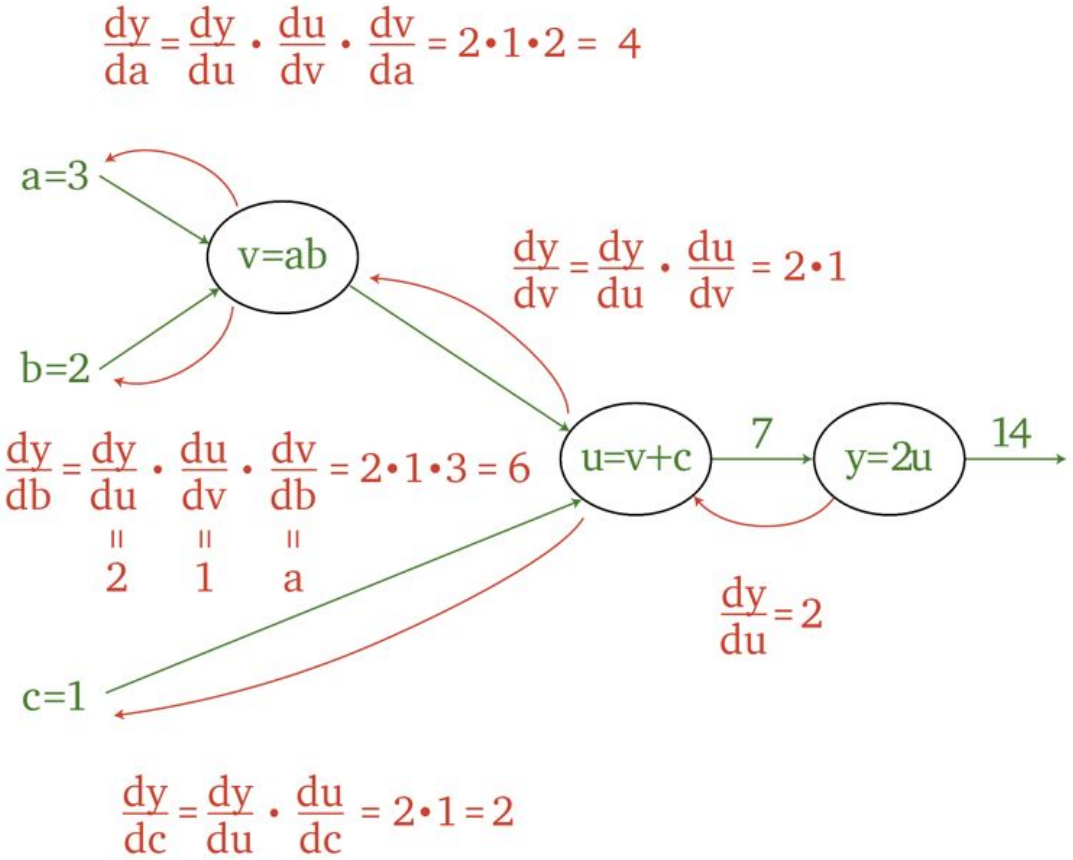
Computation Graph



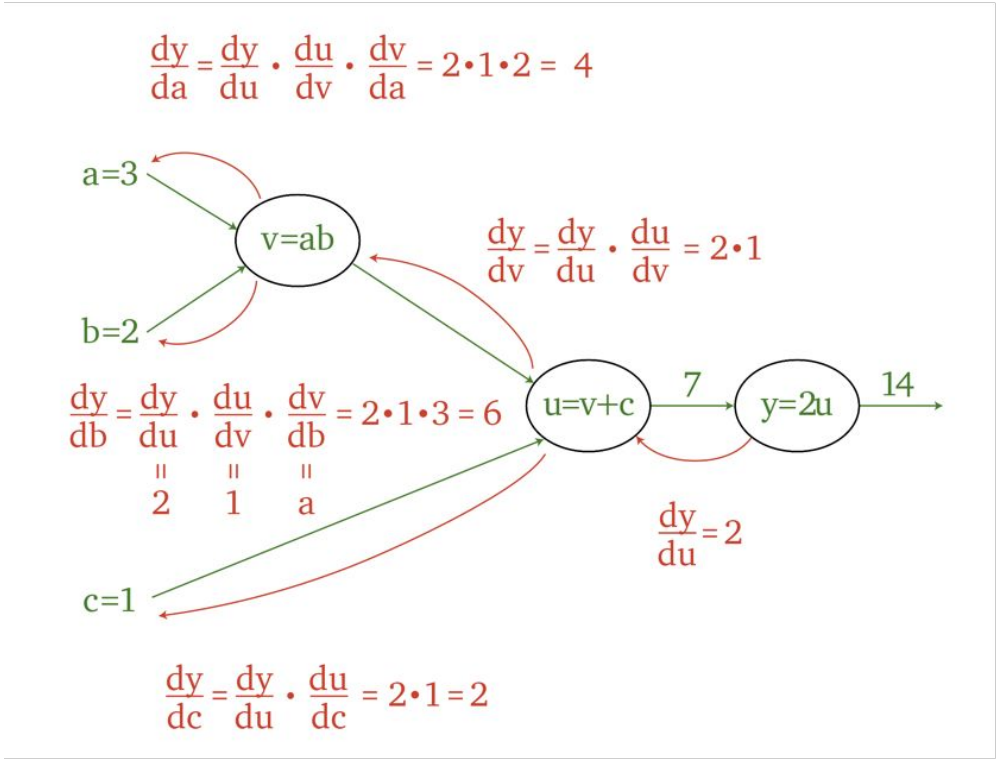
Computation Graph



Computation Graph



Computation Graph




```
a = 3
b = 2
c = 1

v = a * b
u = v + c
y = 2 * u

dydu = 2

dudv = 1
dudc = 1

dydv = dydu * dudv
dydc = dydu * dudc

dvda = b
dvdb = a

dyda = dydu * dudv * dvda
dydb = dydu * dudv * dvdb

print("dyda = ", dyda)
print("dydb = ", dydb)
print("dydc = ", dydc)
```

```
dyda =  4
dydb =  6
dydc =  2
```

Backpropagation Algorithm

1. **Forward propagation**: forward propagate the activations through all layers from input to output, reaching prediction.

2. **Compute loss function**: compute error between prediction and ground truth.

3. **Backpropagation**: use the chain rule for differentiation for backpropagating the gradients through the layers in the opposite direction from the output to the input.

Algorithm

Algorithm 1 Forward and Backward propagation

given an input-label pair x, y

for each layer $l = 1, \ldots, L$ **do**:

$$z^l = W^l a^{l-1}$$

$$a^l = g^l(z^l)$$

for each layer $l = L, \ldots, 1$ **do**:

$$\frac{\partial L(y, f(x, W))}{\partial z^l} = \frac{\partial L(y, f(x, W)}{\partial a^l} * g'^l(z^l)$$

$$\frac{\partial L(y, f(x, W)}{\partial W^l} = \frac{\partial L(y, f(x, W)}{\partial z^l} a^{l-1}$$

Two Functions in One Dimension

$$(f \circ g)' = (f' \circ g)g'$$

$$f(g(x))' = f'(g(x))g'(x)$$

$$\text{for } y = g(x) \text{ and } z = f(y) = f(g(x))$$

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx} = f'(y)g'(x) = f'(g(x))g'(x)$$

Three Functions in One Dimension

$$(f \circ g \circ h)' = (f' \circ g)g'$$

$$f(g(h(x)))' = f'(g(h(x)))g(h(x))' = f'(g(h(x)))g'(h(x))h'(x)$$

$$y = f(u) \text{ and } u = g(v) \text{ and } v = h(x)$$

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dv}\frac{dv}{dx}$$

Two Functions in Higher Dimensions

Let $f : R^m \to R^k$ and $g : R^n \to R^m$. Let $z = f(y)$ and $y = g(x)$. Then $z = f(g(x))$ maps the $n \times 1$ vector x to the $k \times 1$ vector z through the $m \times 1$ vector y.

For $f : R^n \to R^m$ and $y = f(x)$ mapping an $n \times 1$ vector x to an $m \times 1$ vector y, for example by multiplying by an A is an $m \times n$ matrix such that $y = Ax$. Define the Jacobian matrix as:

$$J_f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

and $J_{ij} = \frac{\partial f_i}{\partial x_j}$.

Two Functions in Higher Dimensions

For example, for the function:

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1^2 x_2 \\ 5x_1 + \sin(x_2) = f(x_1, x_2) \end{bmatrix}$$

The Jacobian is:

$$J_f = \begin{bmatrix} 2x_1x_2 & x_1^2 \\ 5 & cos(x_2) = f(x_1, x_2) \end{bmatrix}$$

Two Functions in Higher Dimensions

$$J_{f \circ g}(x) = J_f(g(x)) J_g(x)$$

Two Functions in Higher Dimensions

For $z = f(y) = (f_1(y), \ldots, f_k(y))$ and $y = g(x) = (g_1(x), \ldots, g_m(x))$ applying the chain rule we get:

$$\frac{\partial(z_1, \ldots, z_k)}{\partial(x_1, \ldots, x_n)} = \frac{\partial(z_1, \ldots, z_k)}{\partial(y_1, \ldots, y_m)} \frac{\partial(y_1, \ldots, y_m)}{\partial(x_1, \ldots, x_n)}$$

Gradient of Loss Function

For a neural network, given a single training example, consider the loss function $L(y, f(x, W))$. Writing $\partial L = \partial L(y, f(x, W))$ then by the chain rule of differentiation for the output L we get:

$$\delta^L = \frac{\partial L(y, f(x, W)}{\partial z^L} = \frac{\partial L(y, f(x, W)}{\partial a^L} * g'^L(z^L)$$

and for a squared error loss:

$$L(y, f(x, W)) = \frac{1}{2}\|y - f(x, W)\|_2^2$$

$$\delta^L = -(y - a^L) * g'^L(z^L)$$

Gradient Descent

$$\underset{\mathbf{x}}{\mathbf{minimize}} f(x)$$

Algorithm 1 Gradient descent

given a starting point $x \in \mathbf{dom} f$

repeat

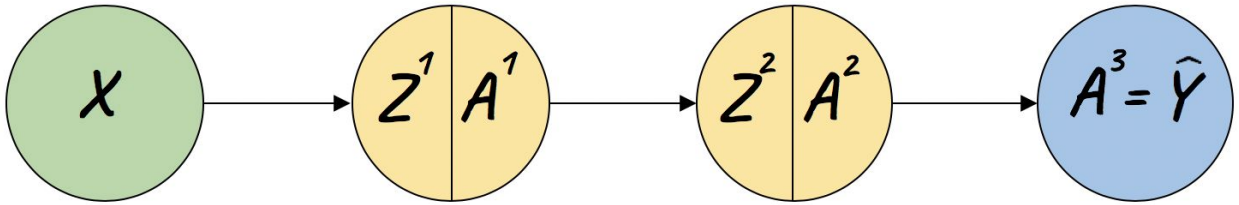
determine descent direction $\Delta x = -\nabla f(x)$

choose step size α

update $x = x + \alpha \Delta x$

until stopping criterion is satisfied.

Forward Propagation

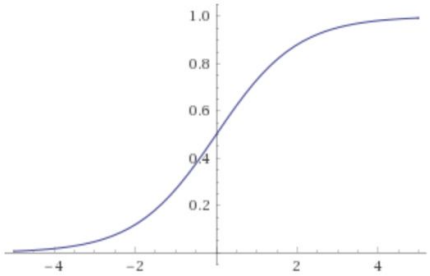


$$A^{l-1} \mapsto A^l$$

$$Z^l = W^l A^{l-1} \qquad A^l = g^l(Z^l)$$

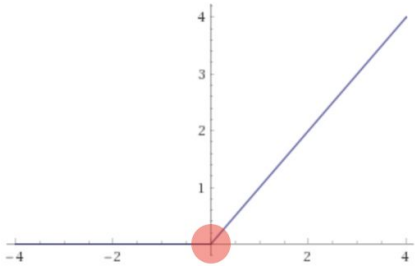
Activation Functions

Sigmoid



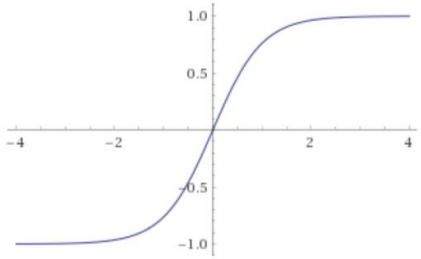
$$\sigma(z) = g(z) = \frac{1}{1 + e^{-z}}$$

ReLU



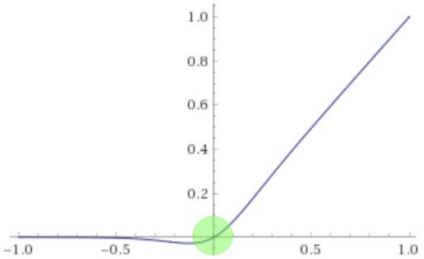
$$g(z) = z_+ = \max(0, z)$$

Hyperbolic Tangent



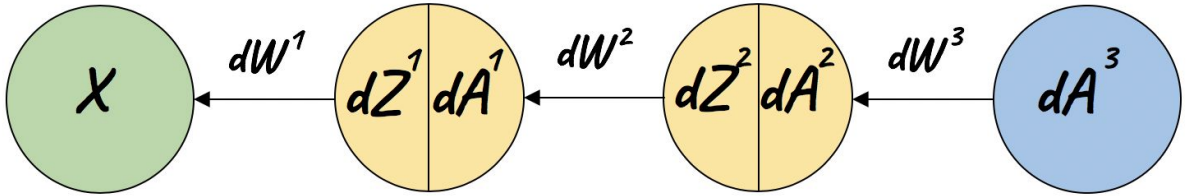
$$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Swish



$$g(z) = z\sigma(\beta z)$$

Backpropagation



Map derivatives from layer l to layer $l-1$ with respect to **activations** and **weights**

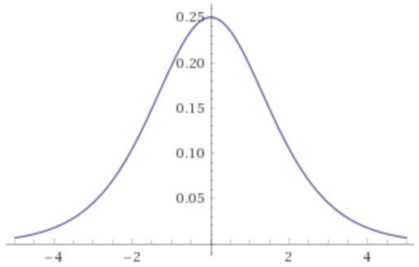
$$dA^{l-1} = \left(W^{l}\right)^{T} dZ^{l}$$

$$dZ^{l} = dA^{l} \cdot g^{l\prime}\left(Z^{l}\right)$$

$$dW^{l} = dZ^{l}\left(A^{l-1}\right)^{T}$$

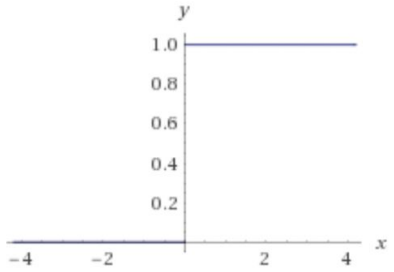
Activation Function Derivatives

Sigmoid derivative



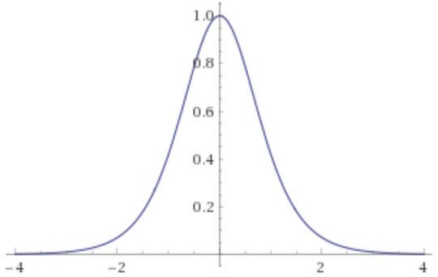
$$g'(z) = \sigma(z)\big(1 - \sigma(z)\big)$$

ReLU derivative



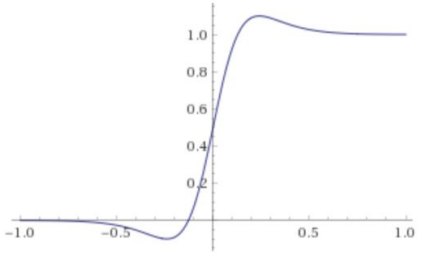
$$g'(z) = \begin{cases} 1 & z > 0 \\ 0 & z < 0 \end{cases}$$

Hyperbolic Tangent derivative



$$g'(z) = \frac{4}{(e^{-z} + e^{z})^2}$$

Swish derivative



$$g'(z) = \sigma(\beta z) + \beta z \sigma(\beta z)\big(1 - \sigma(\beta z)\big)$$

Neural Network Pros

1. Representations sharing weights

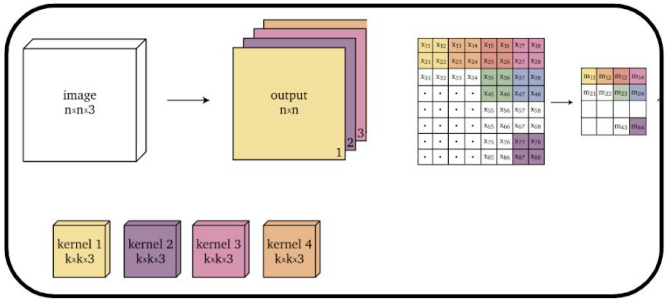
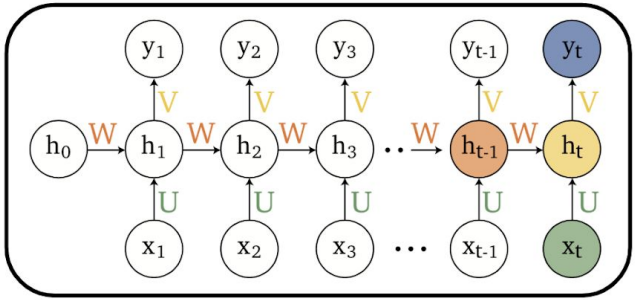
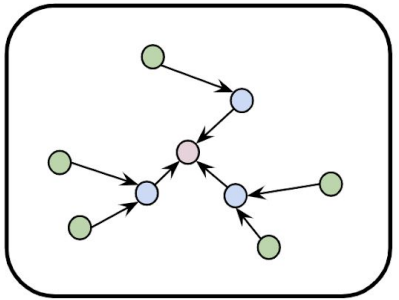


Image (CNN/ResNet/ODENet)
Across **Space**



Sequence (RNN/LSTM/GRU)
Across **Time**



Graph (GNN)
Across **Neighborhoods**

2. Backpropagation by reverse differentiation, O(n) speedup

3. Differentiable programming

NYU

Sequence Models

Text Mining

- Natural language modeling
  - Bag of words
  - N-gram
  - Feature vector
  - Markov model
  - Recurrent neural network
  - Word embeddings

Time Series

Sentence of words; sequence of characters

Image of handwritten words or characters

Notes of music

Sequence of helices/beta-sheets/loops; sequence of amino acids

Sequence of DNA

Audio spectrogram

Video of image frames

...

Applications

Machine translation

Protein amino acid secondary structure prediction

DNA sequence analysis

Speech recognition

Music synthesis

Image captioning

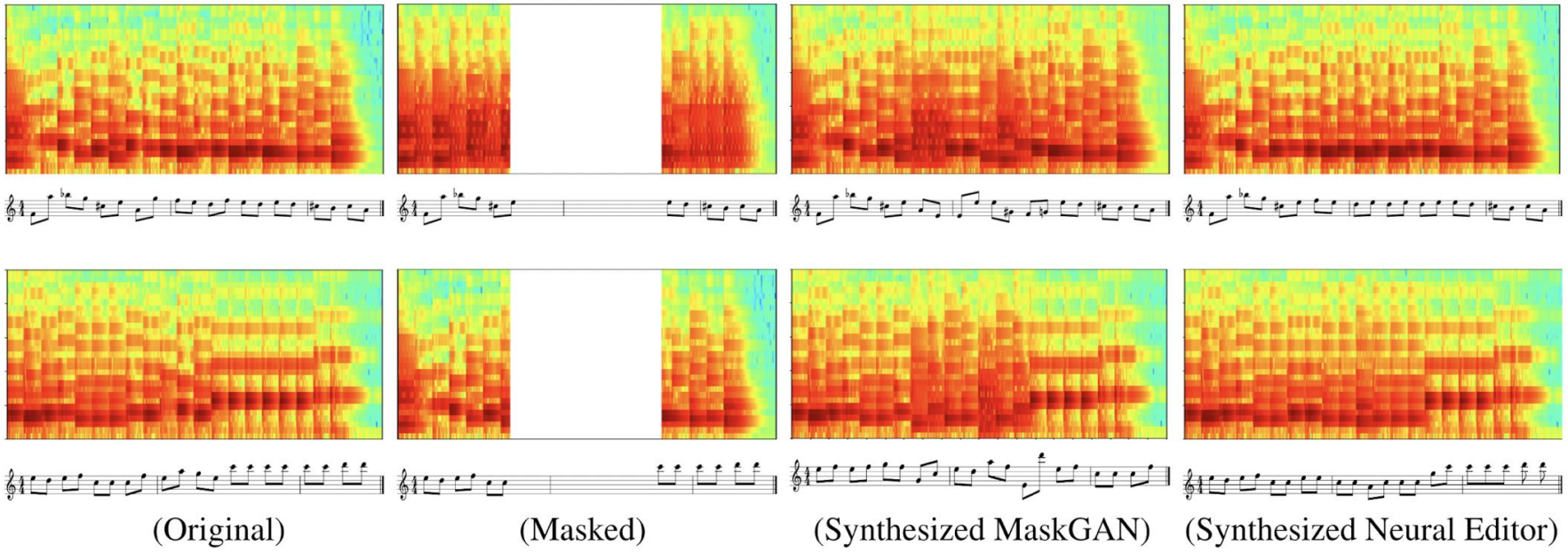
Sentiment classification

Video action recognition

Text handwritten recognition

Self driving cars

Music Synthesis



Johnsen, Lin, Yu, Dempsey, Mhasawade, Jaroslawicz, Drori, 2018

Protein Structure Prediction

MARELEGKVAAVTGAASGIGLASAEAMLAAGARVVMVDRDEAALKALCNKHGDTVIPLVVDLLDPEDCATLLP
RVLEKACQLDILHANAGTYVGGDLVDADTMAIDRMLNLNVNVVMKNVHDVLPHMIERRTGDIIVTSSLAAHFPT
PWEPVYASSKWAINCFVQTVRRQVFKHGIRVGSISPGPVVSALLADWPPEKLKEARDSGSLLEASDVAEVVM
FMLTRPRGMTIRDVLMLPTNFDL

amino acid sequence

|LLLLL|EEEEE|LLLLL|HHHHHHHHHHHHH|LL|EEEEEE|LL|HHHHHHHHHH|LLLL|EEEEE|LLLLL|HHHHHHHH
HHHHHH|LLL|EEEEEE|LLLLLLLLLLLLL|HHHHHHHHHHHHHHHHHHHHHHHHHHHHHH|LL|EEEEEE|LL|HH
HH|LLLLLL|HHHHHHHHHHHHHHHHHHHH|L|HHH|L|EEEEEEEE|LL|E|LL|HHHHH|LLLLLLLLLLLLLL|E|L|H
HHHHHHHHHH|LLLLLLLLLL|EE|LLLLLLLLL|

secondary structure: *loop, helix, sheet*

tertiary structure

Drori et al, 2018

Image Captioning

Vinyals et al, 2015

Natural Language Models

1. **Bag of words** does not preserve order information

2. **Feature vector** requires learning each word order

3. **Markov model** does not model long term dependencies

Natural Language Models

4. **Recurrent neural network** maintains word order and models long term dependencies by sharing parameters across time.

Allows for inputs and outputs of different lengths.

Difficult to train: error signals flowing back in time explode or vanish.

Term Frequency

- How prevalent a term is in a single document

d1 jazz music has a swing rhythm

d2 swing is hard to explain

d3 swing rhythm is a natural rhythm

	a	explain	hard	has	is	jazz	music	natural	rhythm	swing	to
d1	1	0	0	1	0	1	1	0	1	1	0
d2	0	1	1	0	1	0	0	0	0	1	1
d3	1	0	0	0	1	0	0	1	2	1	0

Source: DSB

Term Frequency

- Normalization (to lower case), stemming (remove suffix), stopwords (removed)

Microsoft Corp and Skype Global today announced that they have entered into a definitive agreement under which Microsoft will acquire Skype, the leading Internet communications company, for $8.5 billion in cash from the investor group led by Silver Lake. The agreement has been approved by the boards of directors of both Microsoft and Skype.

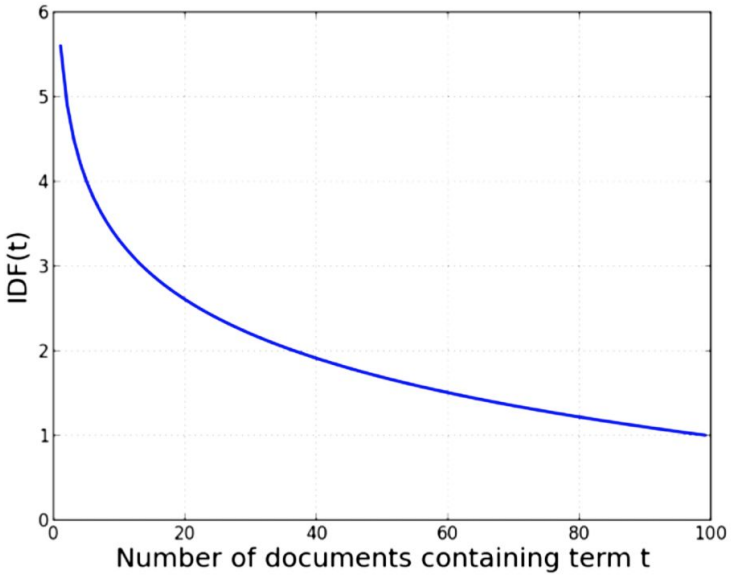
Term	Count	Term	Count	Term	Count	Term	Count
skype	3	microsoft	3	agreement	2	global	1
approv	1	announc	1	acquir	1	lead	1
definit	1	lake	1	communic	1	internet	1
board	1	led	1	director	1	corp	1
compani	1	investor	1	silver	1	billion	1

Source: DSB

Inverse Document Frequency

- Boost of term for being rare in entire corpus of documents

$$\text{IDF}(t) = 1 + \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing } t}\right)$$



Source: DSB

TFIDF

- Product of term frequency and inverse document frequency
- Example: “famous jazz saxophonist born in Kansas who played bebop and latin”. Query is treated as a document.

$$\text{TFIDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

Charlie Parker

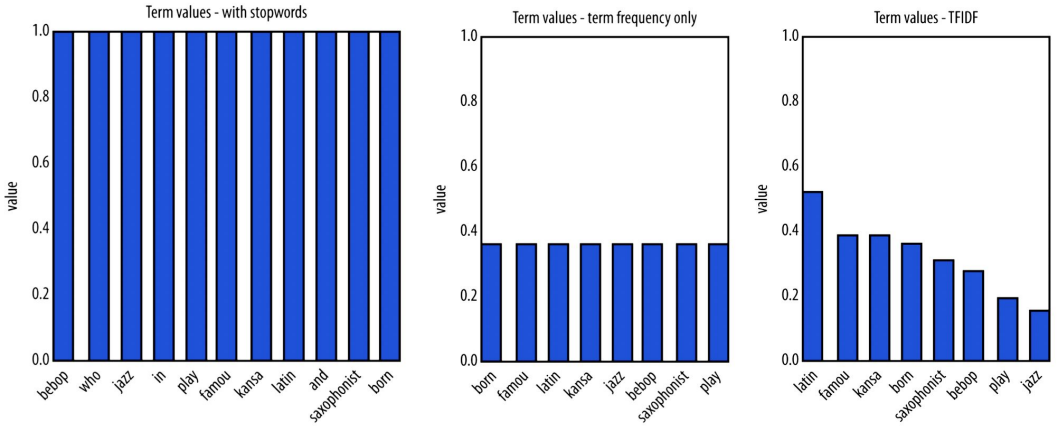
Charles “Charlie” Parker, Jr., was an American jazz saxophonist and composer. Miles Davis once said, “You can tell the history of jazz in four words: Louis Armstrong. Charlie Parker.” Parker acquired the nickname “Yardbird” early in his career and the shortened form, “Bird,” which continued to be used for the rest of his life, inspired the titles of a number of Parker compositions, […]

Duke Ellington

Edward Kennedy “Duke” Ellington was an American composer, pianist, and big-band leader. Ellington wrote over 1,000 compositions. In the opinion of Bob Blumenthal of *The Boston Globe*, “in the century since his birth, there has been no greater composer, American or otherwise, than Edward Kennedy Ellington.” A major figure in the history of jazz, Ellington’s music stretched into various other genres, including blues, gospel, film scores, popular, and classical.[…]

Miles Davis

Miles Dewey Davis III was an American jazz musician, trumpeter, bandleader, and composer. Widely considered one of the most influential musicians of the 20th century, Miles Davis was, with his musical groups, at the forefront of several major developments in jazz music, including bebop, cool jazz, hard bop, modal jazz, and jazz fusion.[…]



Source: DSB

Cosine Similarity

- Between TFIDF of each musician's biography and query.

Musician	Similarity	Musician	Similarity
Charlie Parker	0.135	Count Basie	0.119
Dizzie Gillespie	0.086	John Coltrane	0.079
Art Tatum	0.050	Miles Davis	0.050
Clark Terry	0.047	Sun Ra	0.030
Dave Brubeck	0.027	Nina Simone	0.026
Thelonius Monk	0.025	Fats Waller	0.020
Charles Mingus	0.019	Duke Ellington	0.017
Benny Goodman	0.016	Louis Armstrong	0.012

Source: DSB

Entropy as Expected Value

- Of IDF(T) and IDF(not T)

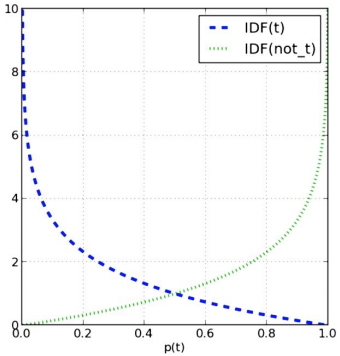
$$p(t) = \frac{\text{Number of documents containing } t}{\text{Total number of documents}}$$

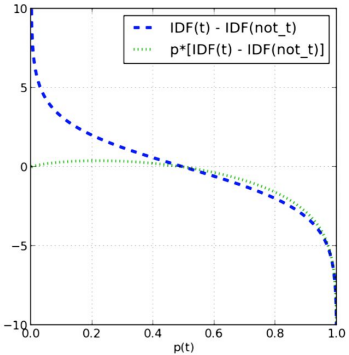
$$\text{IDF}(t) = 1 + \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing } t}\right)$$

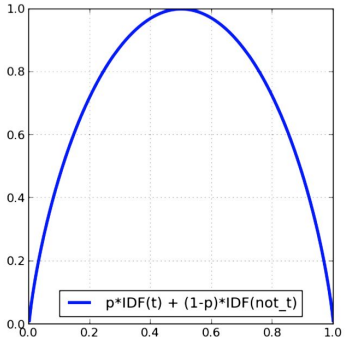
$\text{IDF}(t)$ is basically $log(1/p)$

$$\text{IDF}(not\_T) = \log 1/(1-p) = -\log(1-p)$$

$$\begin{aligned}\text{entropy}(t) &= -p\log(p) - (1-p)\log(1-p)\\ &= p\cdot\text{IDF}(t) - (1-p)[-\text{IDF}(not\_t)]\\ &= p\cdot\text{IDF}(t) + (1-p)[\text{IDF}(not\_t)]\end{aligned}$$







Source: DSB

N-grams

A sequence of n adjacent words.

A bag of words is a 1-gram (unigram) model: $p(x_1, \ldots, x_n) \approx \prod_i p(x_i)$

A Markov model is a 2-gram (bigram) model: $p(x_n|x_1, \ldots, x_{n-1}) \approx p(x_n|x_{n-1})$

Usually k=3,4,5 gram models are computed and stored given a large corpus.

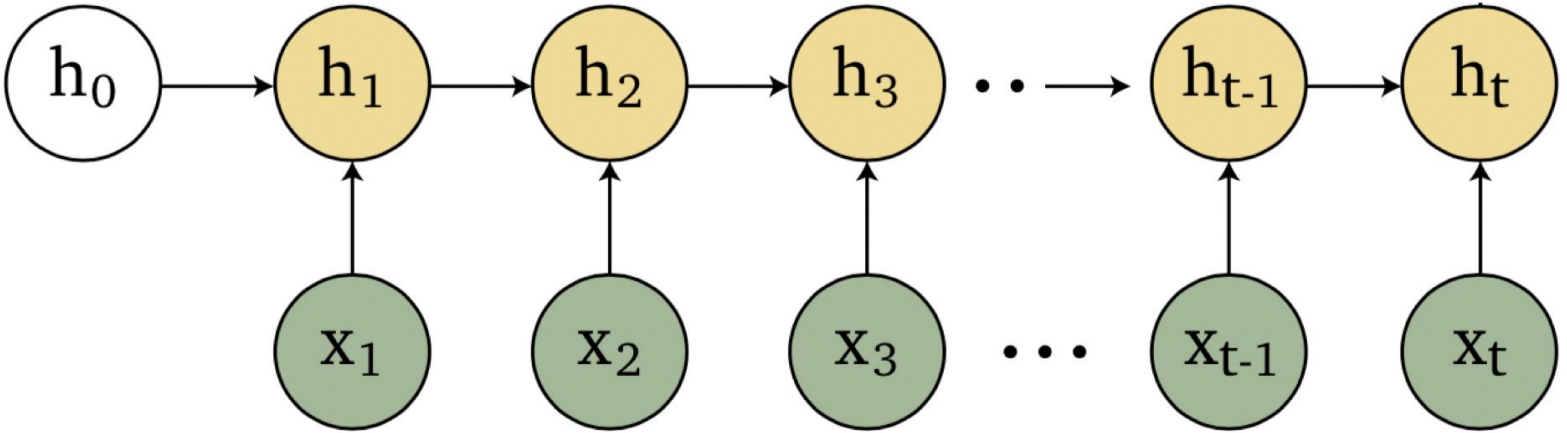
$$p(x_n|x_1, \ldots, x_{n-1}) \approx p(x_n|x_{n-1}, \ldots, x_{n-k+1})$$

Recurrent Neural Network (RNN)

Process sequence $x_1, \ldots, x_t$

Share parameters across time

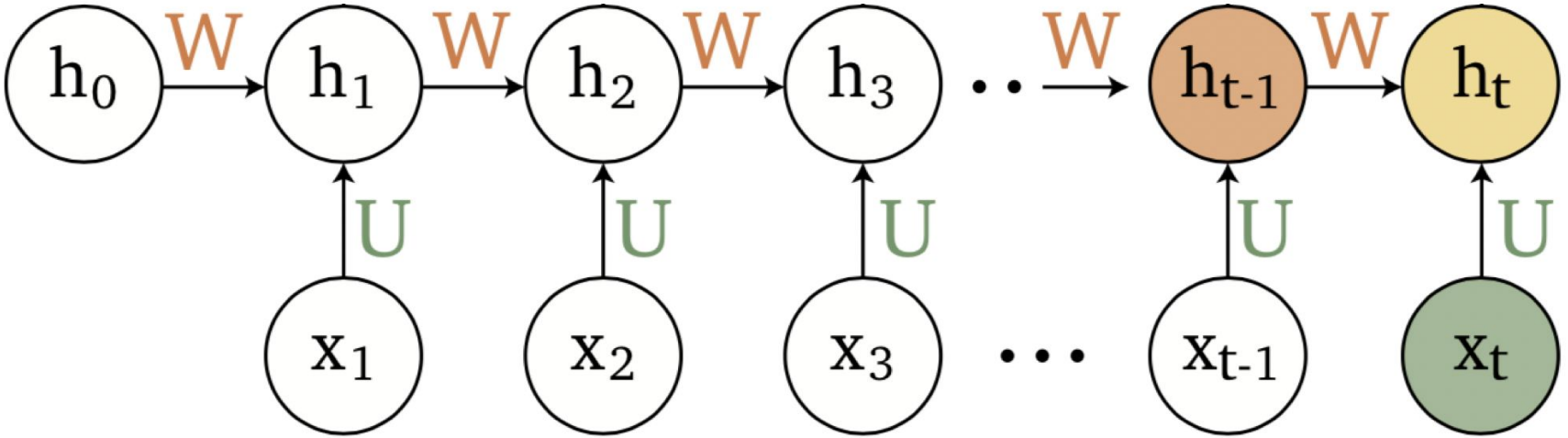
$$h_t = f(h_{t-1}, x_t)$$



RNN

Share weight matrices W, U for all t

$$h_t = f(h_{t-1}, x_t) \qquad h_t = g(Wh_{t-1} + Ux_t) \qquad g(z) = \tanh(z)$$

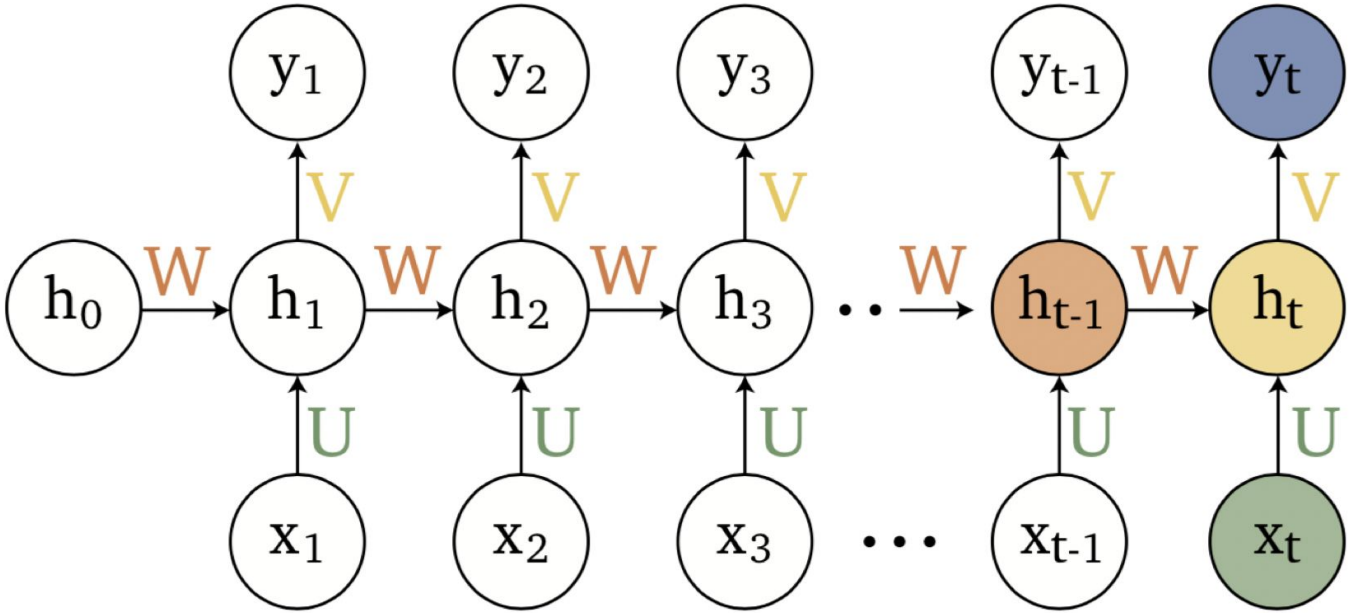


RNN

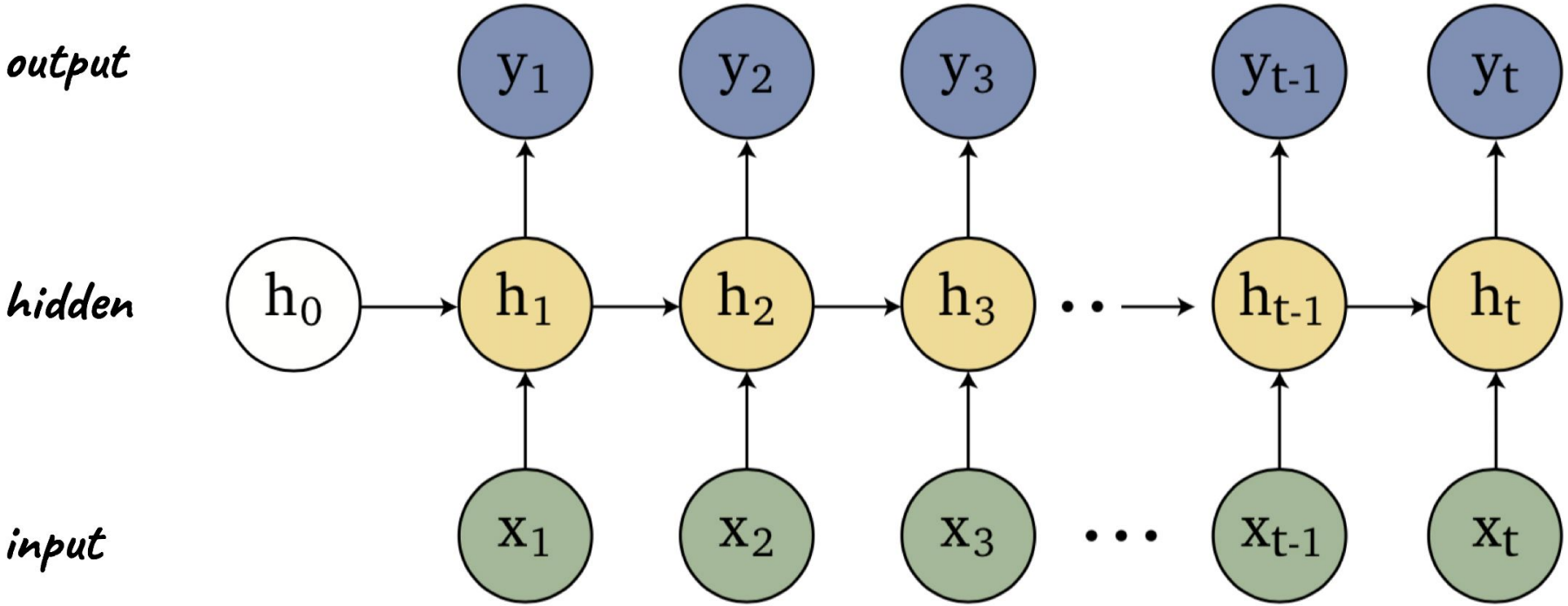
Share weight matrices W, U, V for all t

$y_t = g(Vh_t)$

$h_t = g(Wh_{t-1} + Ux_t)$



RNN


output
hidden
input
y_1
y_2
y_3
y_{t-1}
y_t
h_0
h_1
h_2
h_3
h_{t-1}
h_t
x_1
x_2
x_3
x_{t-1}
x_t

RNN

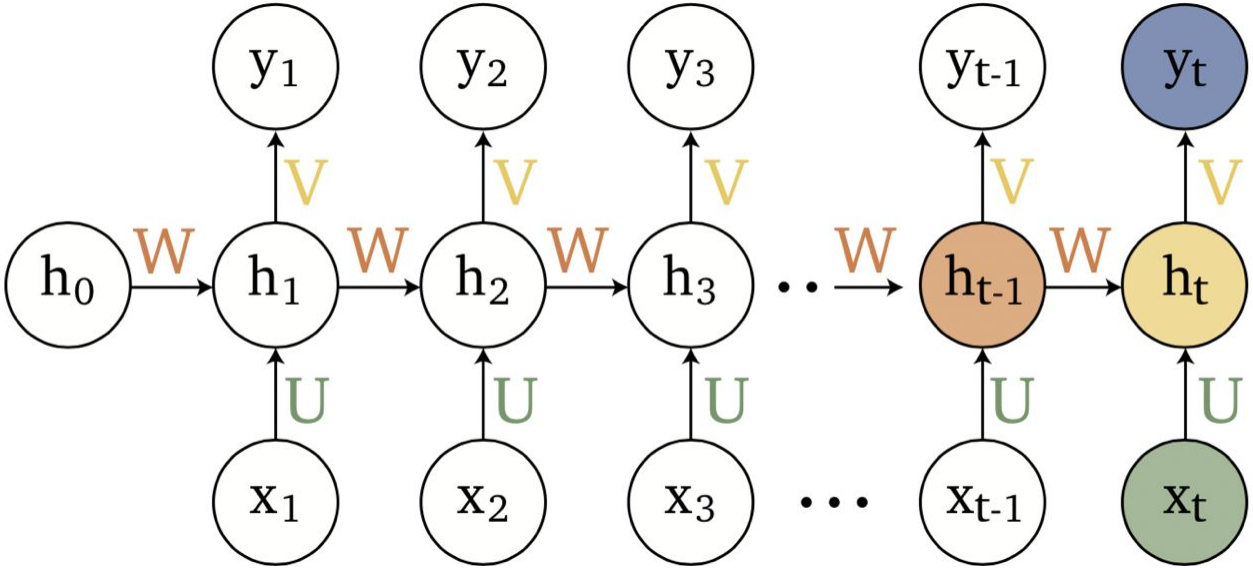
Share weight matrices W, U, V for all t

U input to hidden

W hidden to hidden

V hidden to output

RNN



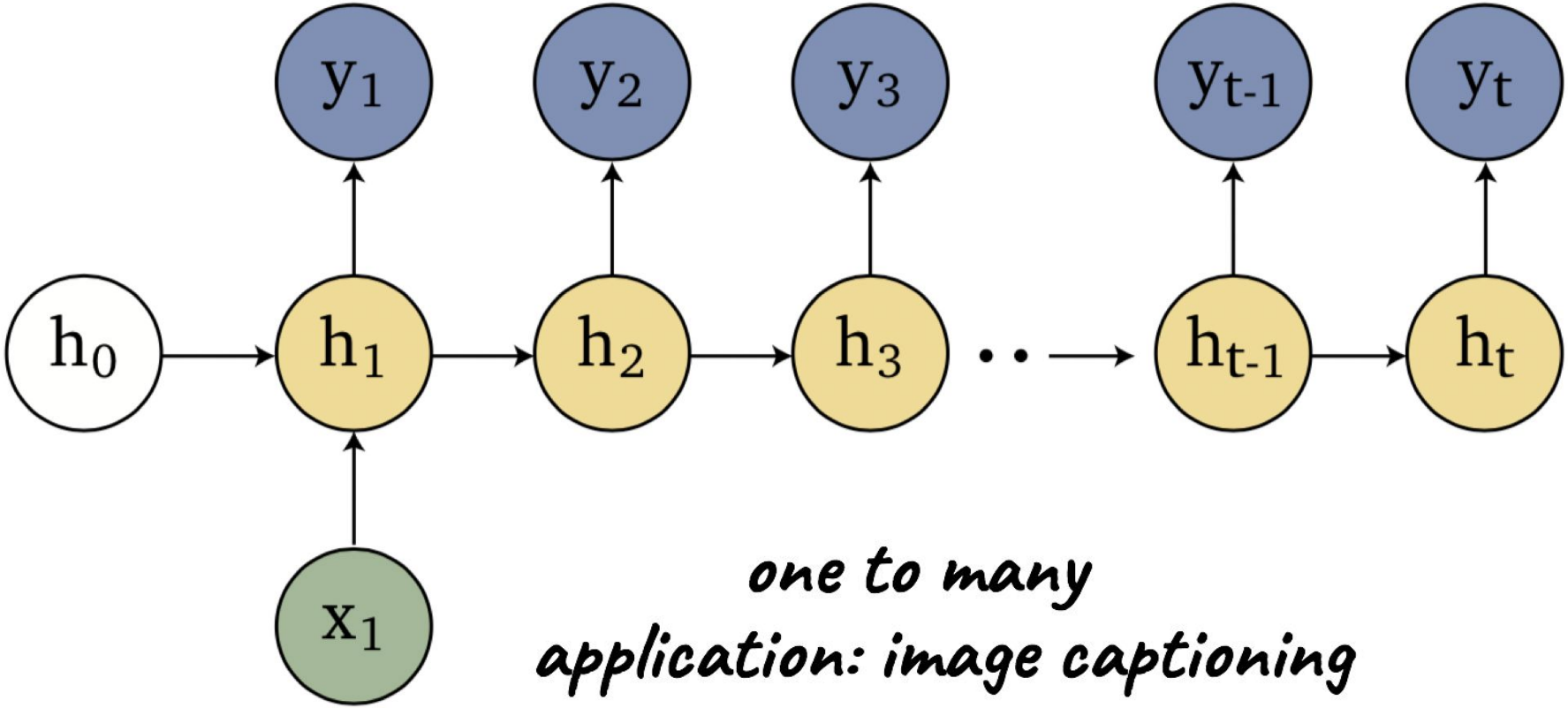
$$y_t = g(Vh_t)$$

$$h_t = g(Wh_{t-1} + Ux_t)$$

$$[W \quad U]\begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}$$

Y

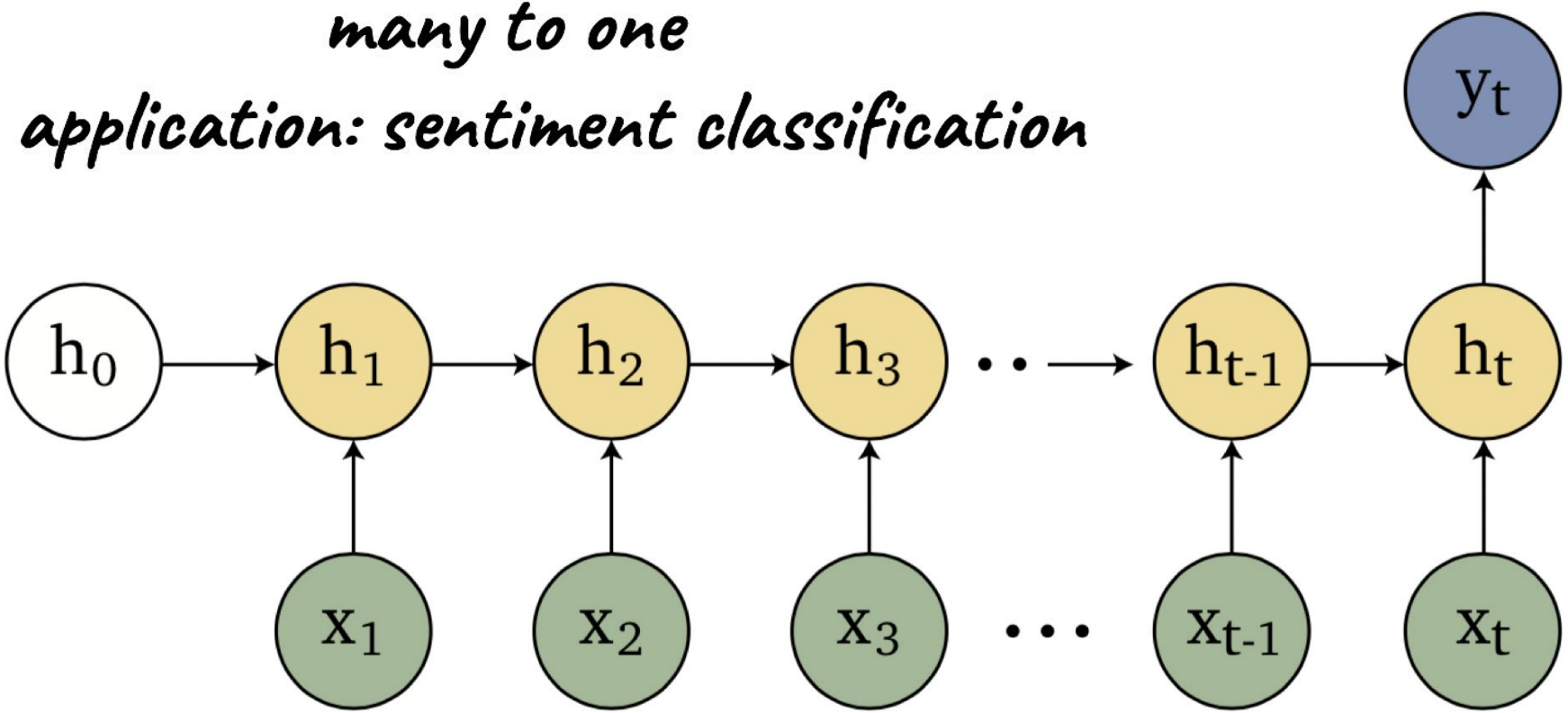
RNN Architectures



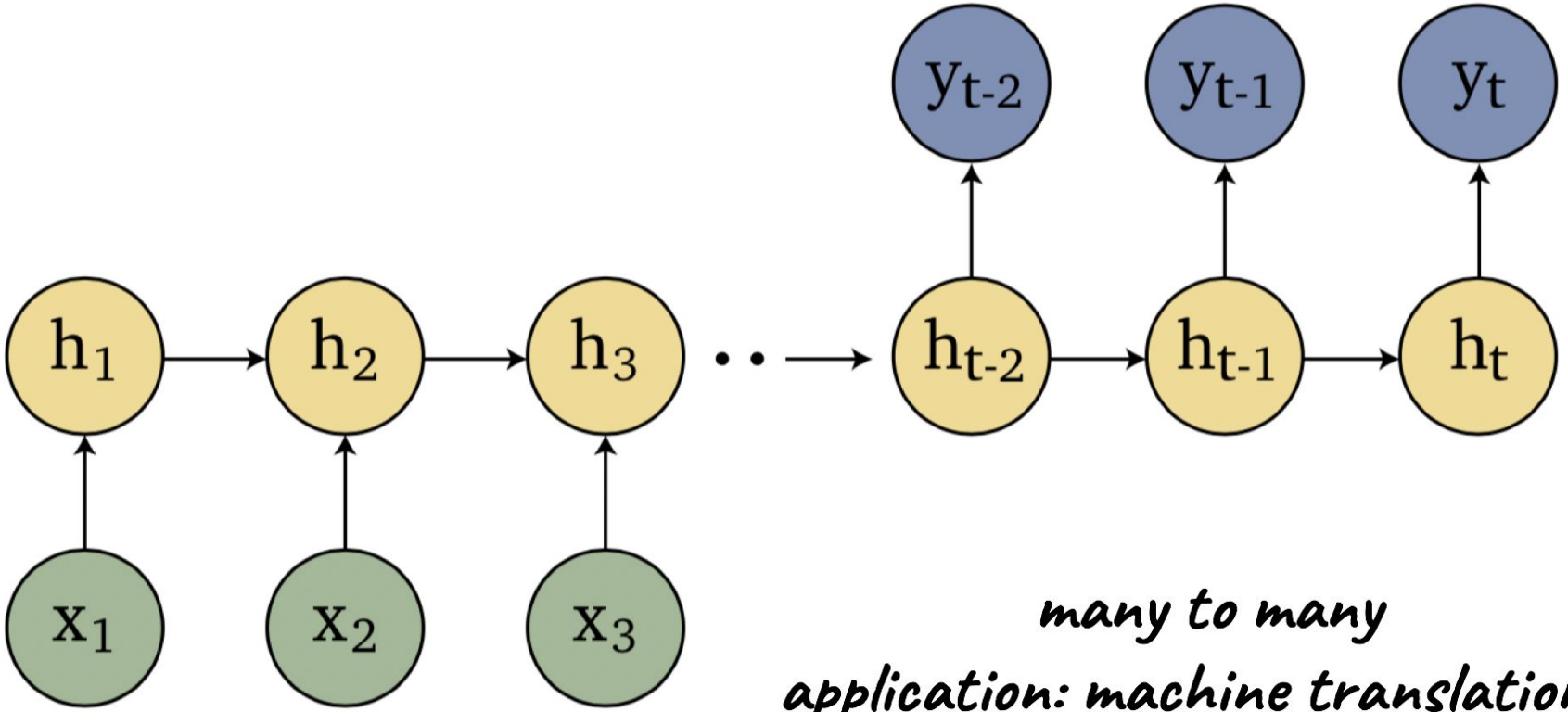
RNN Architectures

many to one

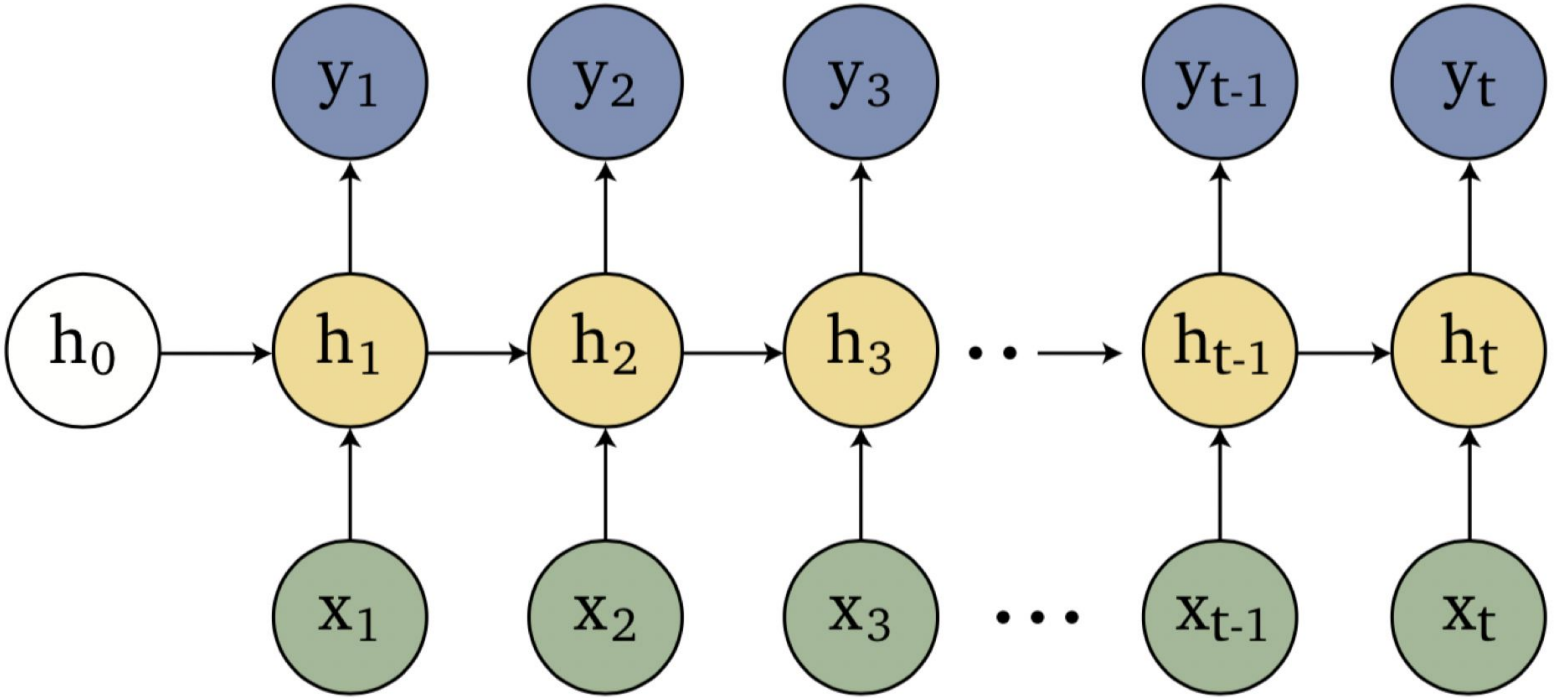
application: sentiment classification



RNN Architectures



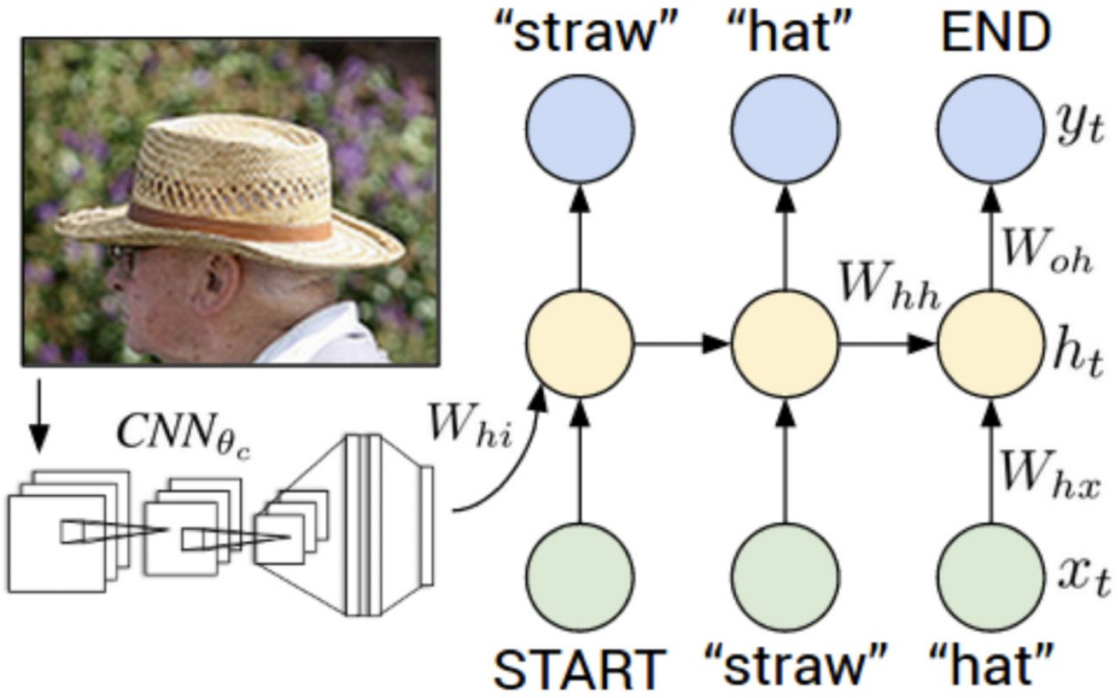
RNN Architectures



many to many

application: video classification

RNN Architectures



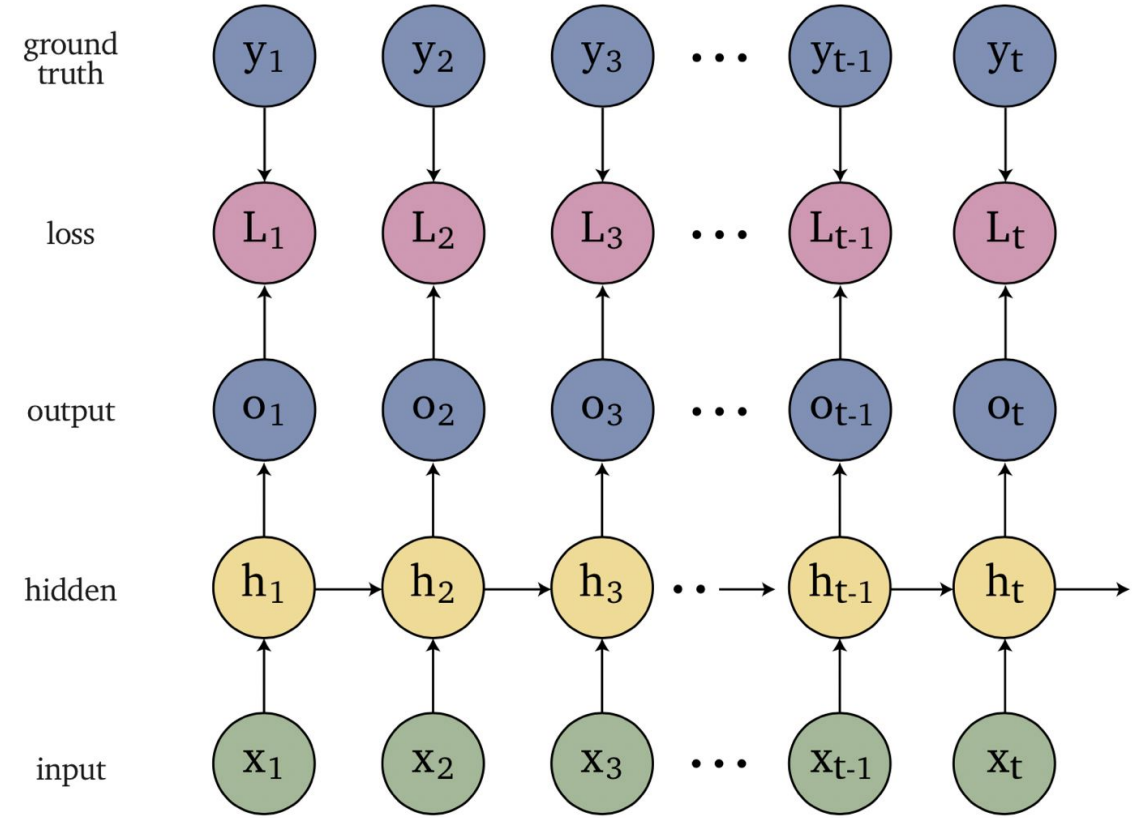
Karpathy et al, 2015

Loss Function

$$\hat{y}_t = softmax(o_t) \qquad \mathcal{L} = \sum_t \mathcal{L}_t$$

$$o_t = Vh_t$$

$$h_t = g(Wh_{t-1} + Ux_t)$$

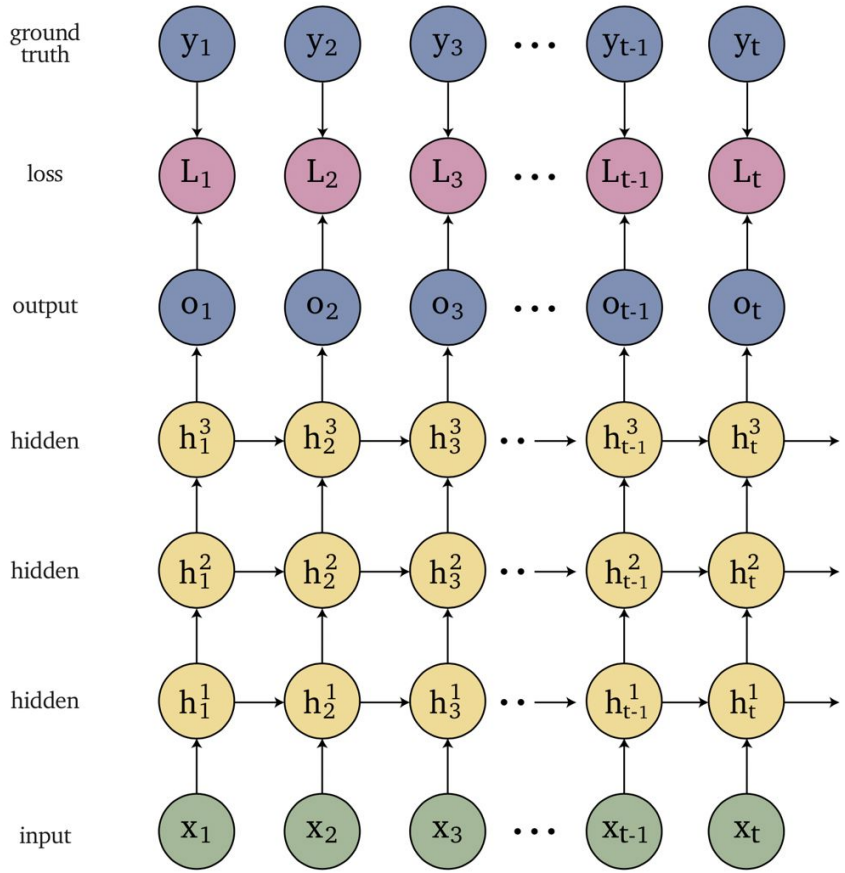


Deep RNN

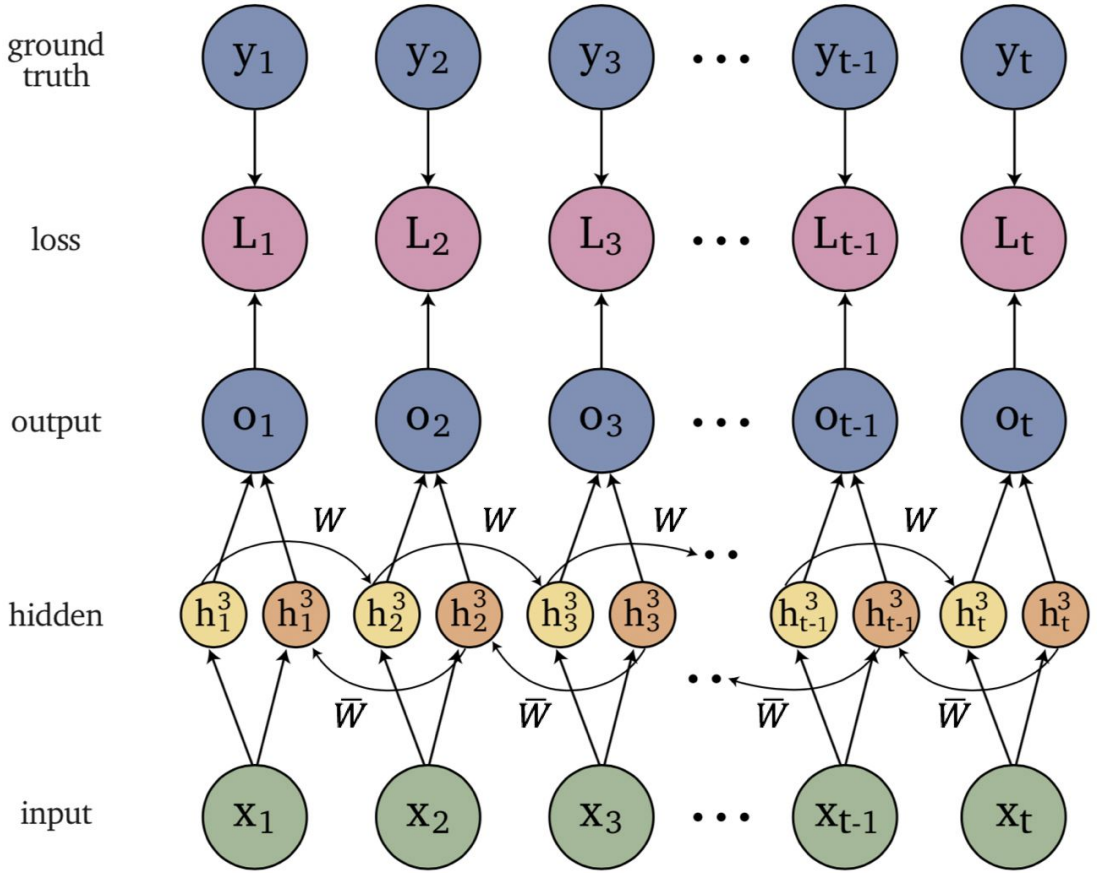
$$o_t = V \begin{bmatrix} h_t \\ \bar{h}_t \end{bmatrix}$$

$$h_t = g(W h_{t-1} + U x_t)$$

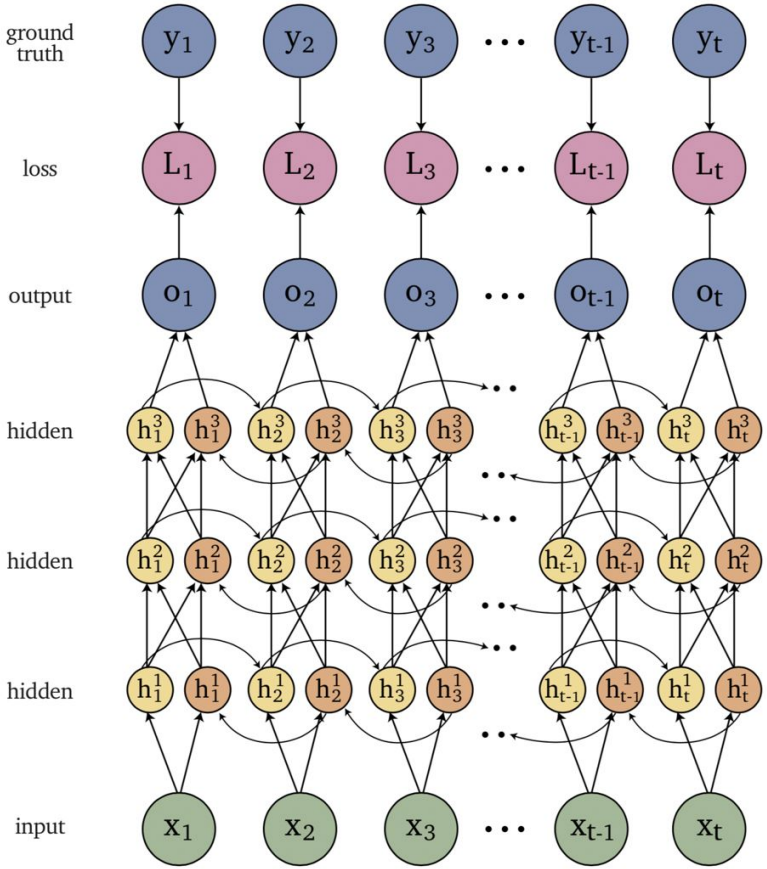
$$\bar{h}_t = g(\bar{W} \bar{h}_{t-1} + U x_t)$$



Bidirectional RNN



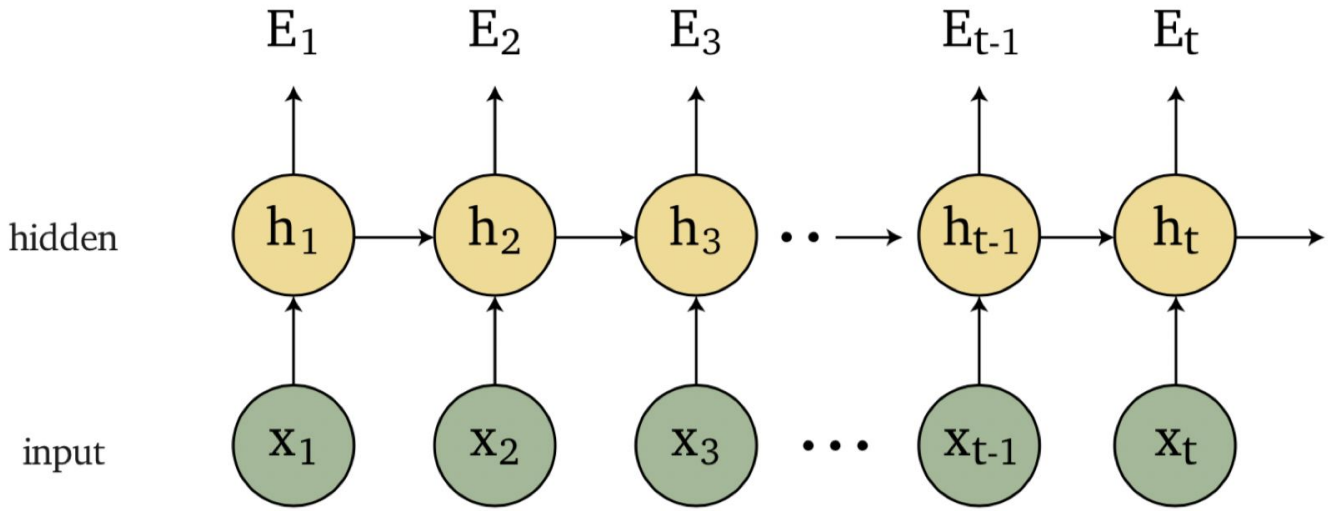
Deep Bidirectional RNN



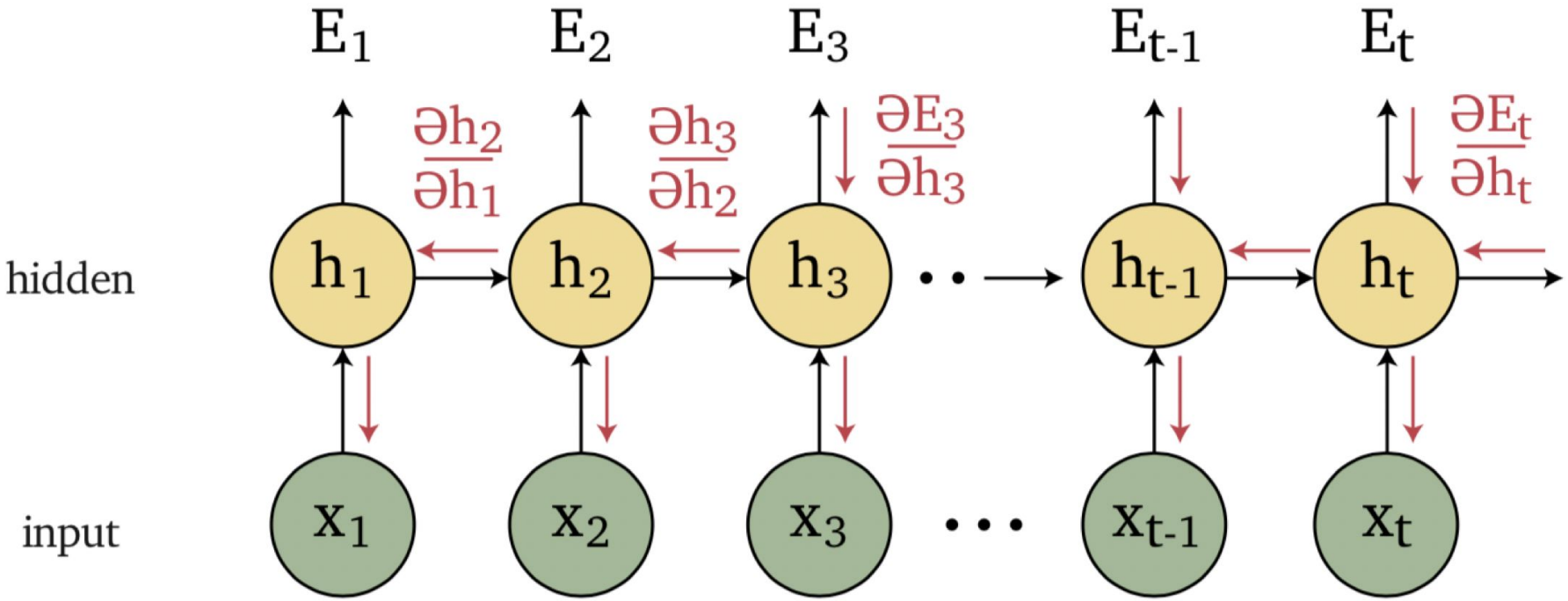
Backpropagation Through Time

$$E_t(y_t, \hat{y}_t) = -y_t log\hat{y}_t$$

$$E(y, \hat{y}) = \sum_t E_t(y_t, \hat{y}_t) = -\sum_t y_t log\hat{y}_t$$



Backpropagation Through Time

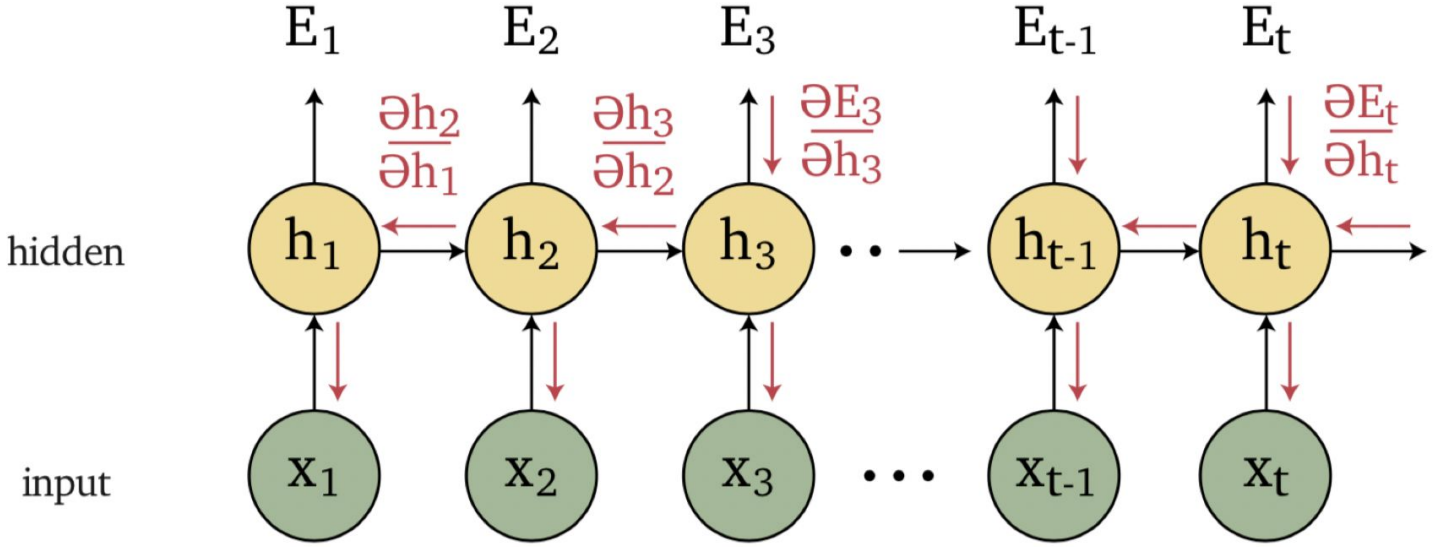


Backpropagation Through Time

$$\frac{\partial E_3}{\partial V} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial V} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial z_3}\frac{\partial z_3}{\partial V}$$

$$z_3 = V h_3$$

only depends on t=3



Backpropagation Through Time

$$\frac{\partial E_3}{\partial W} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\frac{\partial h_3}{\partial W} \qquad h_3 = \tanh(W h_2 + U x_3)$$

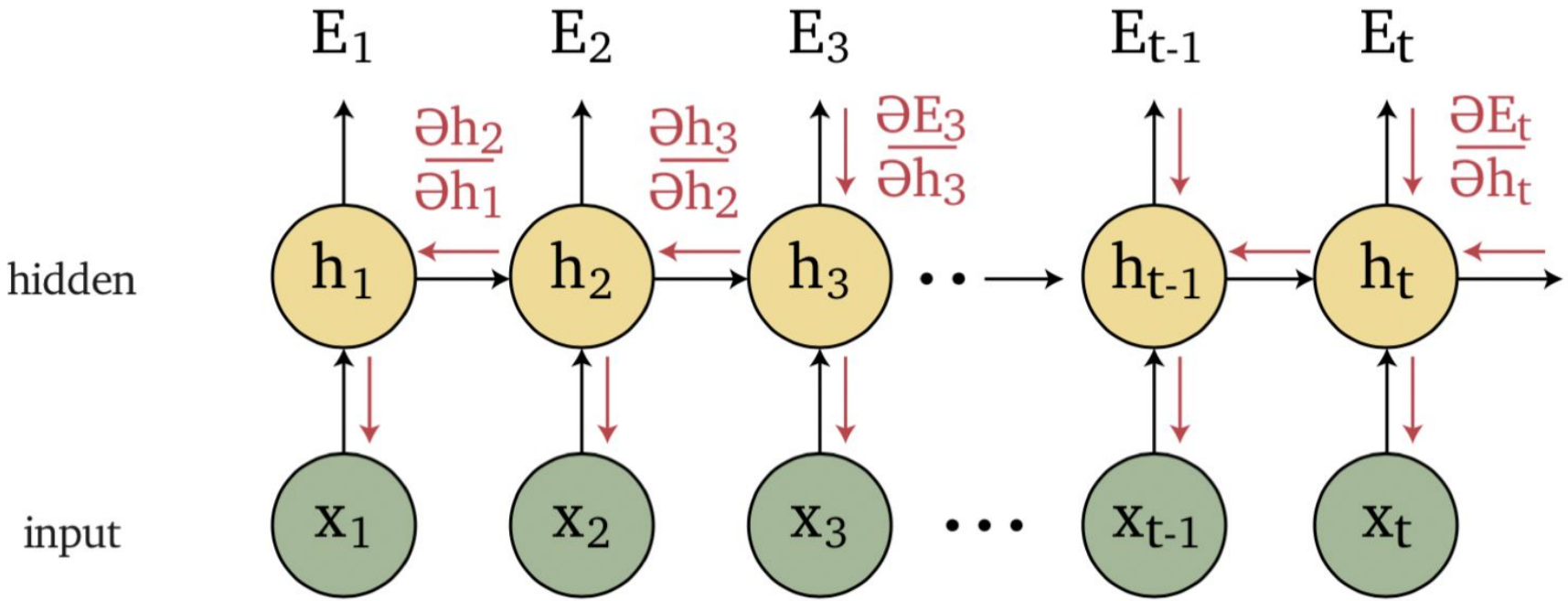
$$\sum_{i=1}^{3} \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\frac{\partial h_3}{\partial h_i}\frac{\partial h_i}{\partial W} \qquad \frac{\partial h_3}{\partial h_1} = \frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}$$

depends on t=3,2,1

$$\frac{\partial E_3}{\partial W} = \sum_{i=1}^{3} \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\left(\prod_{j=i+1}^{3} \frac{\partial h_j}{\partial h_{j-1}}\right)\frac{\partial h_i}{\partial W}$$

$$\prod W^T diag(\tanh'(h_{t-1}))$$

Backpropagation Through Time

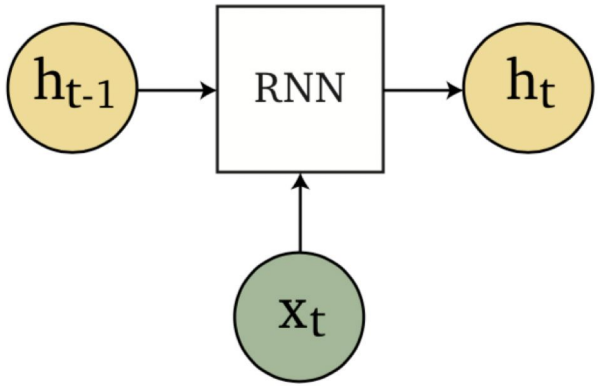


RNN

Simple model

Hard to train: recurrent weight matrix raised to high power causes gradients to vanish or explode.

1986

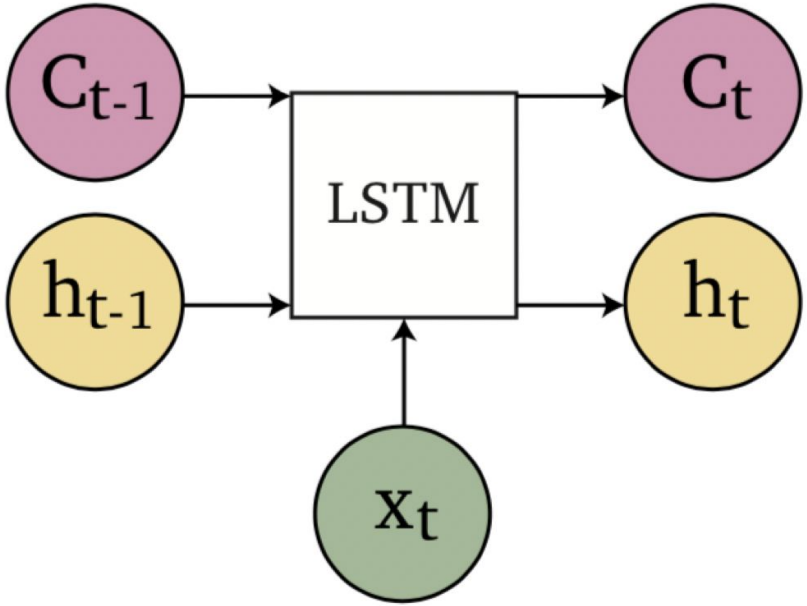


Long Short Term Memory (LSTM)

Separate memory cell

Easy to train

1997

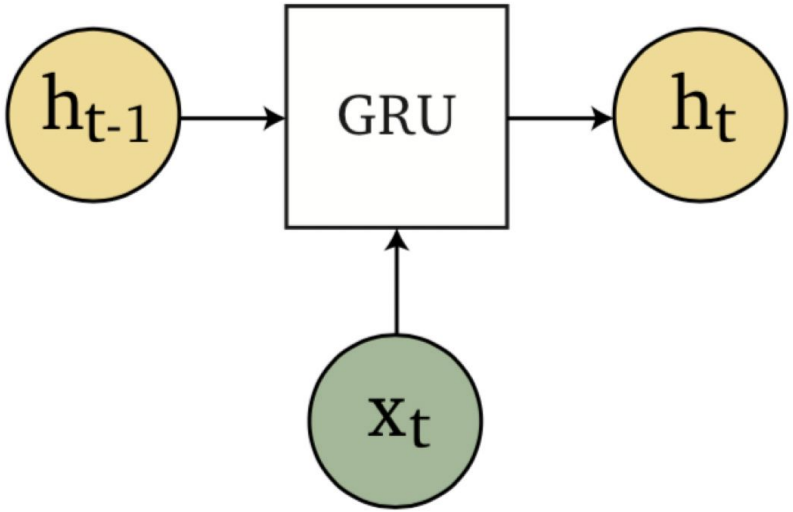


Gated Recurrent Unit (GRU)

Simple

Easy to train

2014



NYU

Word Embedding

Word Embedding (Encoding)

- Replace 1-hot word representation with lower dimensional feature vector for each word.
- 1-hot representation is limited, dot product of any two 1-hot word representations is 0, so does not capture relationships among words.
- Finite dictionary of words, finite fixed encoding, embedding, of words, into lower dimensional space.
- Learn word embedding from large unsupervised text corpus.
- Use in supervised task, taking each word embedding as input.

Word Embedding

- Analogies

$$A : B :: A' : B'$$

  - *Eman : Ewoman :: Eking : Equeen* (Mikolov et al 2013)

$$\underset{w}{\operatorname{argmax}}\, d(e_{B'}, e_A - e_B + e_{A'}) \qquad d(u, v) = \frac{u^T v}{\|u\|\|v\|}$$

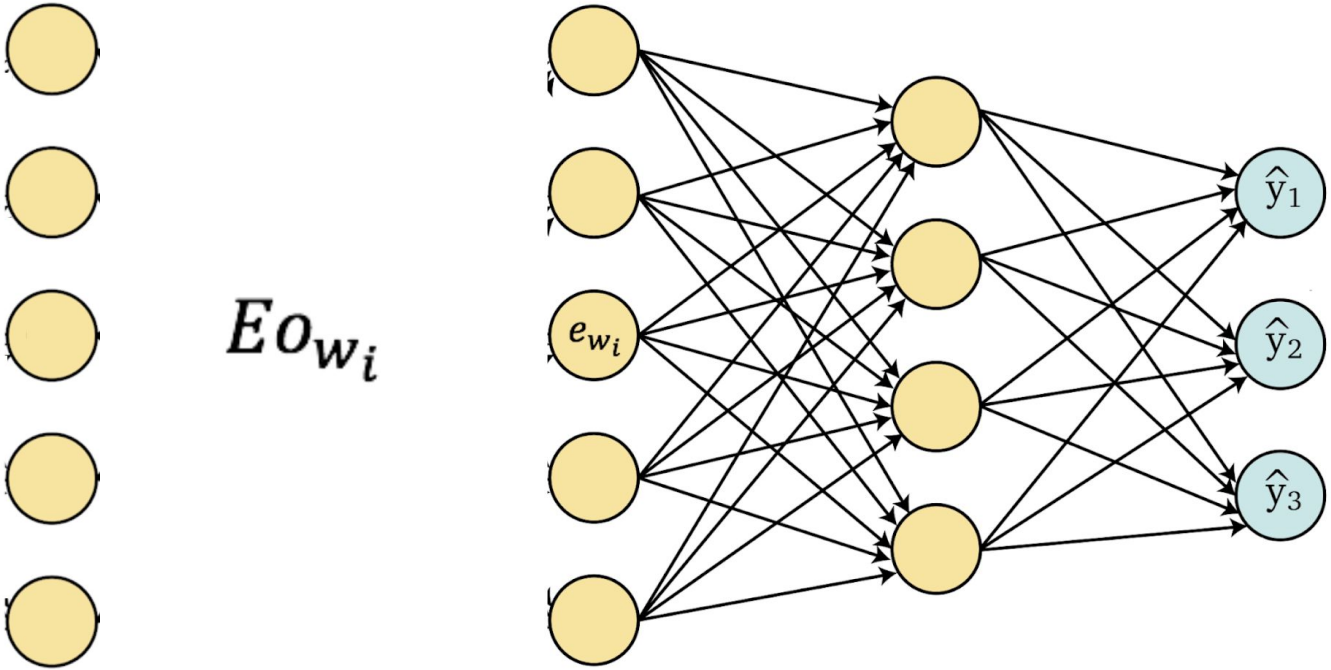
- Fix bias in corpus
  - *Eman : Ewoman :: Eprogrammer : Ehomemaker* (Bolukbasi 2016)
  - Project onto bias direction

Word Embedding

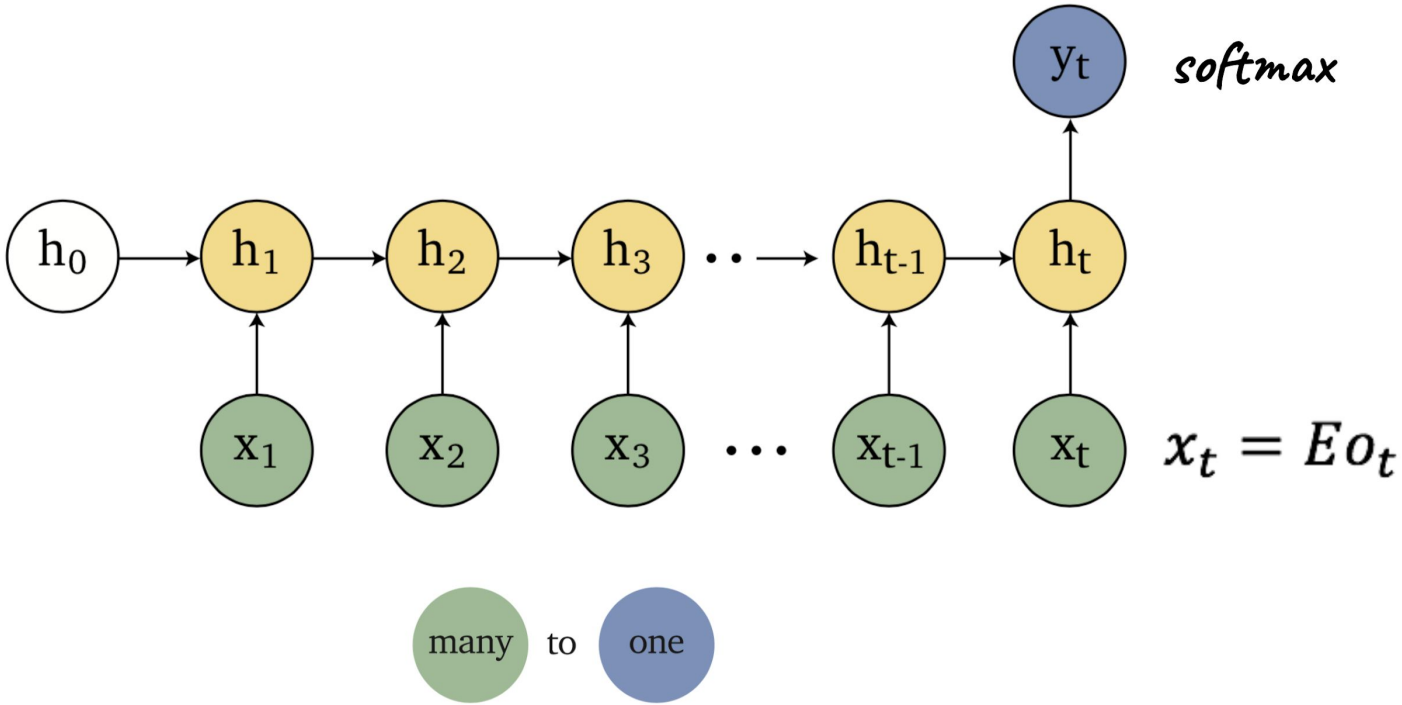
$$\underset{n\times p}{E}\ \underset{p\times 1}{o_w} = \underset{n\times 1}{e_w}$$

Word Embedding

Mary had a little lamb whose fleece was white as snow



Application



sentiment classification

Application: Reading Comprehension

Context (passage and previous question/answer pairs)

Tom goes everywhere with Catherine Green, a 54-year-old secretary. He moves around her office at work and goes shopping with her. "Most people don't seem to mind Tom," says Catherine, who thinks he is wonderful. "He's my fourth child," she says. She may think of him and treat him that way as her son. He moves around buying his food, paying his health bills and his taxes, but in fact Tom is a dog.

Catherine and Tom live in Sweden, a country where everyone is expected to lead an orderly life according to rules laid down by the government, which also provides a high level of care for its people. This level of care costs money.

People in Sweden pay taxes on everything, so aren't surprised to find that owning a dog means more taxes. Some people are paying as much as 500 Swedish kronor in taxes a year for the right to keep their dog, which is spent by the government on dog hospitals and sometimes medical treatment for a dog that falls ill. However, most such treatment is expensive, so owners often decide to offer health and even life _ for their dog.

In Sweden dog owners must pay for any damage their dog does. A Swedish Kennel Club official explains what this means: if your dog runs out on the road and gets hit by a passing car, you, as the owner, have to pay for any damage done to the car, even if your dog has been killed in the accident.

Q: How old is Catherine?
A: 54

Q: where does she live?
A:

Model answer: Stockholm
Turker answers: Sweden, Sweden, in Sweden, Sweden

Source: Language Models are Unsupervised Multitask Learners, Radford et al, 2019

Application: Reading Comprehension

Context (passage and previous question/answer pairs)

The 2008 Summer Olympics torch relay was run from March 24 until August 8, 2008, prior to the 2008 Summer Olympics, with the theme of "one world, one dream". Plans for the relay were announced on April 26, 2007, in Beijing, China. The relay, also called by the organizers as the "Journey of Harmony", lasted 129 days and carried the torch 137,000 km (85,000 mi) – the longest distance of any Olympic torch relay since the tradition was started ahead of the 1936 Summer Olympics.

After being lit at the birthplace of the Olympic Games in Olympia, Greece on March 24, the torch traveled to the Panathinaiko Stadium in Athens, and then to Beijing, arriving on March 31. From Beijing, the torch was following a route passing through six continents. The torch has visited cities along the Silk Road, symbolizing ancient links between China and the rest of the world. The relay also included an ascent with the flame to the top of Mount Everest on the border of Nepal and Tibet, China from the Chinese side, which was closed specially for the event.

Q: What was the theme
A: "one world, one dream".

Q: What was the length of the race?
A: 137,000 km

Q: Was it larger than previous ones?
A: No

Q: Where did the race begin?
A: Olympia, Greece

Q: Is there anything notable about that place?
A: birthplace of Olympic Games

Q: Where did they go after?
A: Athens

Q: How many days was the race?
A: seven

Q: Did they visit any notable landmarks?
A: Panathinaiko Stadium

Q: And did they climb any mountains?
A:

Model answer: Everest
Turker answers: unknown, yes, Yes, yes

Source: Language Models are Unsupervised Multitask Learners, Radford et al, 2019