deeplearning.ai

# Basics of Neural Network Programming

## Vectorization

# What is vectorization?

$z = w^T x + b$

$w = \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}$ $x = \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}$ $w \in \mathbb{R}^{n_x}$ $x \in \mathbb{R}^{n_x}$

Non-vectorized:

```
z = 0
for i in range(n-x):
    z += w[i] * x[i]
z += b
```

Vectorized

```
z = np.dot(w,x) + b
```

$w^T x$

→ GPU
→ CPU

} SIMD - single instruction multiple data.

deeplearning.ai

# Basics of Neural Network Programming

## More vectorization examples

# Neural network programming guideline

Whenever possible, avoid explicit for-loops.

# Neural network programming guideline

Whenever possible, avoid explicit for-loops.

$u = Av$

$u_i = \sum_j A_{ij} v_j$

```
u = np.zeros((n,1))
for i ...
    for j ...
        u[i] += A[i][j] * v[j]
```

```
u = np.dot(A,v)
```

# Vectors and matrix valued functions

Say you need to apply the exponential operation on every element of a matrix/vector.

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} \rightarrow u = \begin{bmatrix} e^{v_1} \\ e^{v_2} \\ \vdots \\ e^{v_n} \end{bmatrix}$$

```
u = np.zeros((n,1))
for i in range(n):
    u[i]=math.exp(v[i])
```

```
import numpy as np
u = np.exp(v)
```

```
np.log(v)
np.abs(v)
np.maximum(v,0)
v**2
1/v
```

# Logistic regression derivatives

$J = 0$, ~~$dw_1 = 0$, $dw_2 = 0$~~, $db = 0$

$dw = np.zeros((n_x, 1))$

for $i = 1$ to $n$:

$z^{(i)} = w^T x^{(i)} + b$

$a^{(i)} = \sigma(z^{(i)})$

$J \mathrel{+}= -\left[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})\right]$

$dz^{(i)} = a^{(i)}(1 - a^{(i)})$

for $j = 1 \ldots n_x$:

~~$dw_1 \mathrel{+}= x_1^{(i)} z^{(i)}$~~

~~$dw_2 \mathrel{+}= x_2^{(i)} z^{(i)}$~~

$n_x = 2$

$dw \mathrel{+}= x^{(i)} dz^{(i)}$

$db \mathrel{+}= dz^{(i)}$

$J = J/m$, ~~$dw_1 = dw_1/m$, $dw_2 = dw_2/m$~~, $db = db/m$

$dw /= m$.

deeplearning.ai

# Basics of Neural Network Programming

---

# Broadcasting in Python

# Broadcasting example

Calories from Carbs, Proteins, Fats in 100g of different foods:

| | Apples | Beef | Eggs | Potatoes |
|---|---|---|---|---|
| Carb | 56.0 | 0.0 | 4.4 | 68.0 |
| Protein | 1.2 | 104.0 | 52.0 | 8.0 |
| Fat | 1.8 | 135.0 | 99.0 | 0.9 |

$= A$ (3,4)

59 cal

$\frac{56}{59} \approx 94.9\%$

Calculate % of calories from Carb, Protein, Fat. Can you do this without explicit for-loop?

```
cal = A.sum(axis = 0)
percentage = 100*A/(cal.reshape(1,4))
```

(3,4) / (1,4)

# Broadcasting example

$$\begin{bmatrix}1\\2\\3\\4\end{bmatrix} + \begin{bmatrix}100\\100\\100\\100\end{bmatrix} \quad 100$$

$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix} + \begin{bmatrix}100 & 200 & 300\\100 & 200 & 300\end{bmatrix}$$

(m, n) (2,3) (1, n) ⇝ (m, n) (2,3)

$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix} + \begin{bmatrix}100 & 100 & 100\\200 & 200 & 200\end{bmatrix} =$$

(m, n) (m, 1) ⇝ (m, n)

# General Principle

$(m, n)$ matrix

$+ \; - \; * \; /$

$(1, n) \rightsquigarrow (m, n)$

$(m, 1) \rightsquigarrow (m, n)$

$(m, 1) + \mathbb{R}$

$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + 100 = \begin{bmatrix} 101 \\ 102 \\ 103 \end{bmatrix}$

$[1 \; 2 \; 3] + 100 = [101 \; 102 \; 103]$

Matlab/Octave: bsxfun

deeplearning.ai

# Basics of Neural Network Programming

---

# A note on python/ numpy vectors

# Python Demo

# Python / numpy vectors

```
import numpy as np

a = np.random.randn(5)

a = np.random.randn((5,1))

a = np.random.randn((1,5))

assert(a.shape = (5,1))
```