deeplearning.ai

Basics of Neural Network Programming

Vectorization

What is vectorization?

np.array([1,2,3,4,5])

$z = w^T x + b$

$w^T = [\ldots\ldots]$ $x = \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}$

$w, x \in \mathbb{R}^{n_x}$

```
z = np.dot(w,x) + b  → fast
```

v/s

```
for i range(n_x)
    z += w[i] * x[i]
z = z + b
```

O(n) slow

Almost 300x time diff

Why?

SIMD is provided by the CPU/GPU

↳ Single Instruction Multiple data

parallelization instructions provided by the CPU/GPU

np.dot uses SIMD more efficiently than a for loop

deeplearning.ai

Basics of Neural Network Programming

More vectorization examples

Neural network programming guideline

Whenever possible, avoid explicit for-loops.

Suppose the task is to vectorise

$u = A \cdot v$

$A \in \mathbb{R}^{m \times n}$

$v \in \mathbb{R}^{n \times 1}$

$\Rightarrow u \in \mathbb{R}^{m \times 1}$

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

$O(n^2)$

V/S

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

Neural network programming guideline

Whenever possible, avoid explicit for-loops.

Vectors and matrix valued functions

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

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

```
u = np.exp(v)
    np.log(v)
    np.abs(v)
    np.max(v,0)
    v**2
      ↳ [v_1^2 ; ... ; v_n^2]
```

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

Andrew Ng

Logistic regression derivatives

No need for this

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

```
J = 0, dw1 = 0, dw2 = 0, db = 0
for i = 1 to n:m
```

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

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

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

$$dz^{(i)} = \cancel{a^{(i)}(1 - a^{(i)})} \quad a^{(i)} - y^{(i)}$$

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

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

We were updating all our weights here for i 1 to nx

w1 w2 ... wn

$$J = J/m, \quad dw_1 = dw_1/m, \quad dw_2 = dw_2/m, \quad db = db/m$$

How to get rid off this? Make w a vector!

↳ No need for this

Instead do $dw /= m$