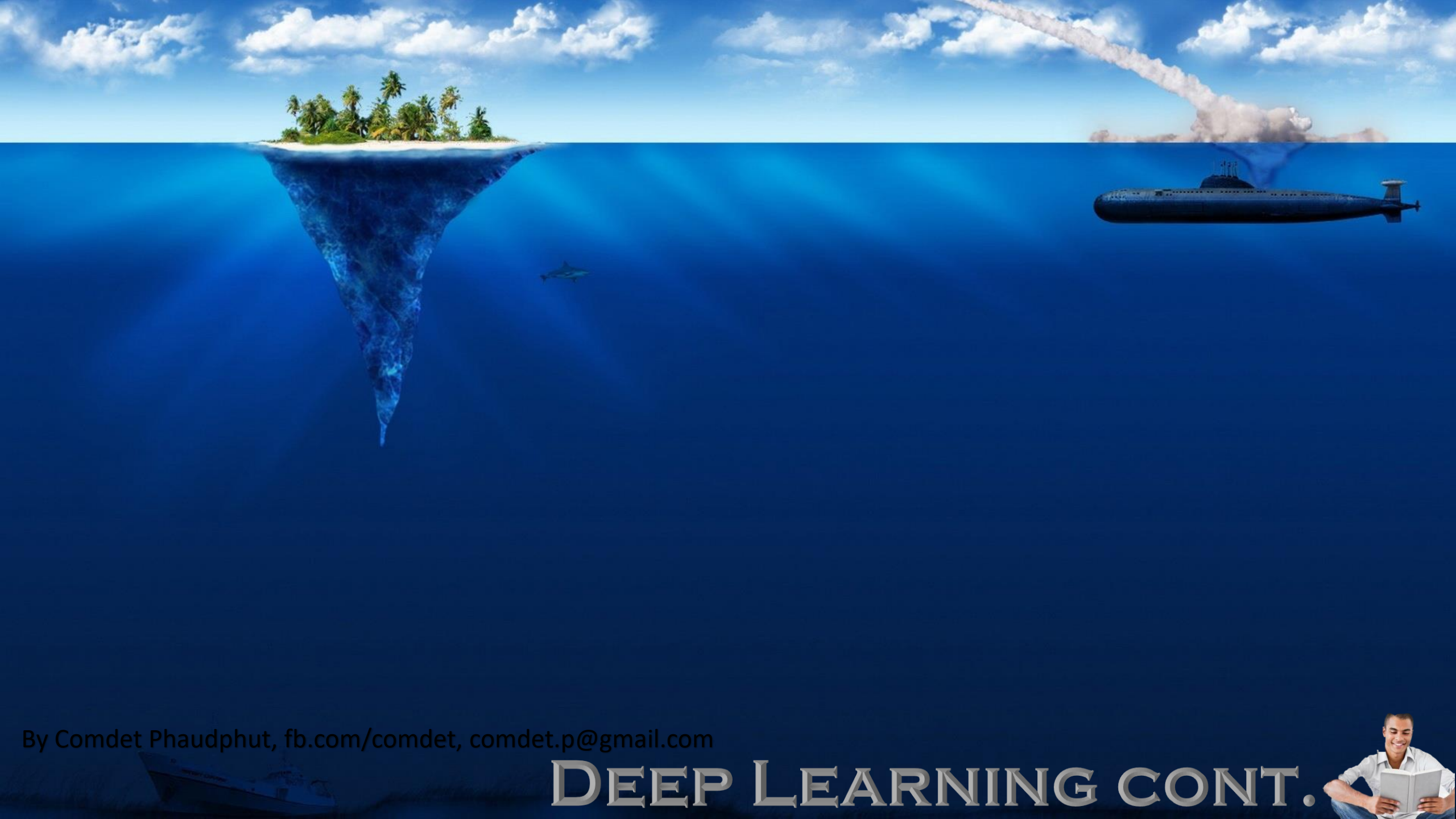
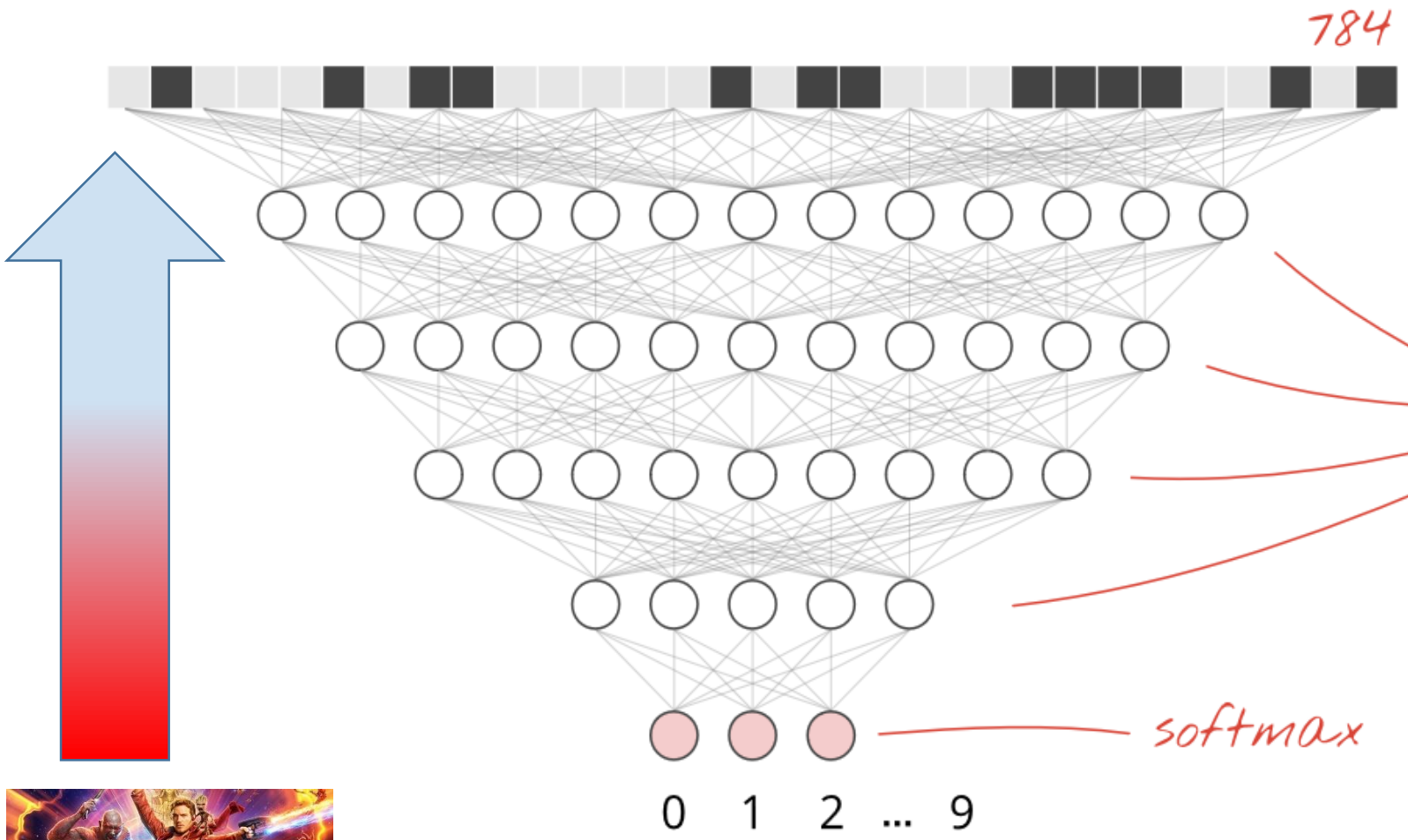


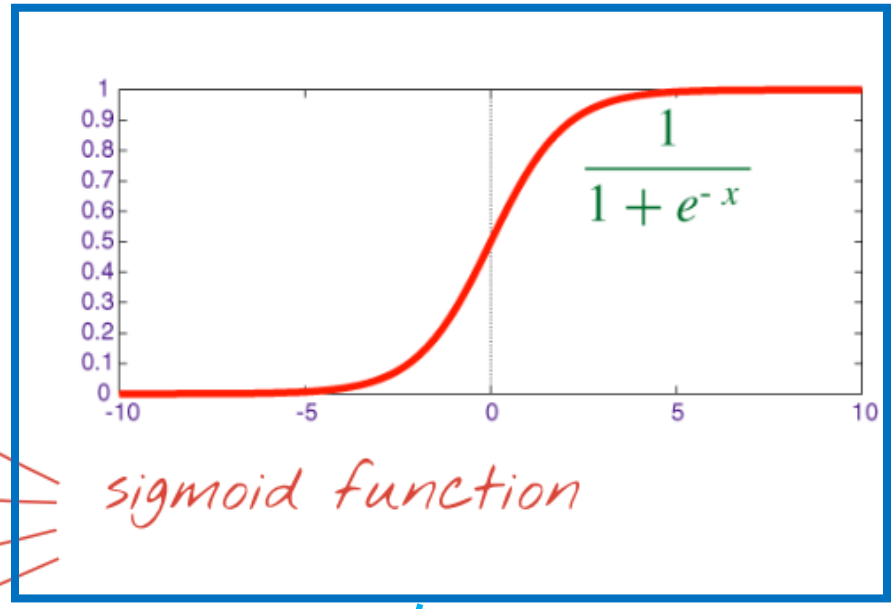
By Comdet Phaudphut, fb.com/comdet, comdet.p@gmail.com

Deep Learning cont.

Review Prev Lab

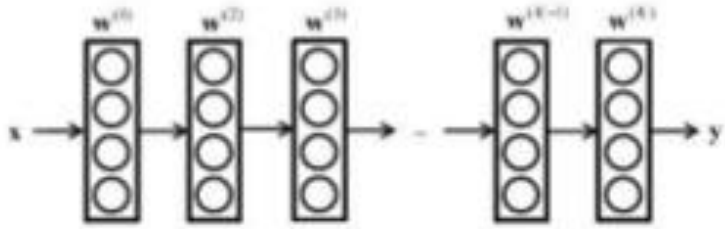
- We add 3 (someone 4) hidden layers.
- But we didn’t gain more accuracy as expected.
- Why ?
  - code errors or bugs?
  - machine mistake?
  - Take a look closer, what happen.

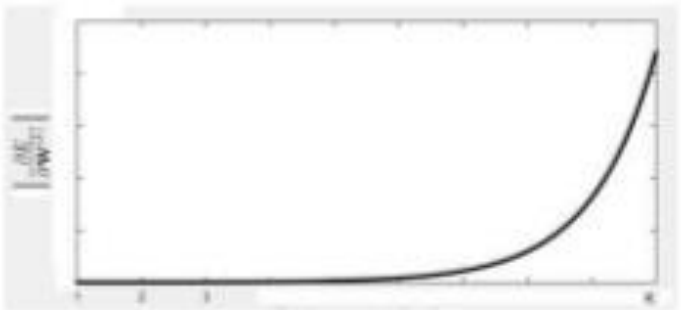


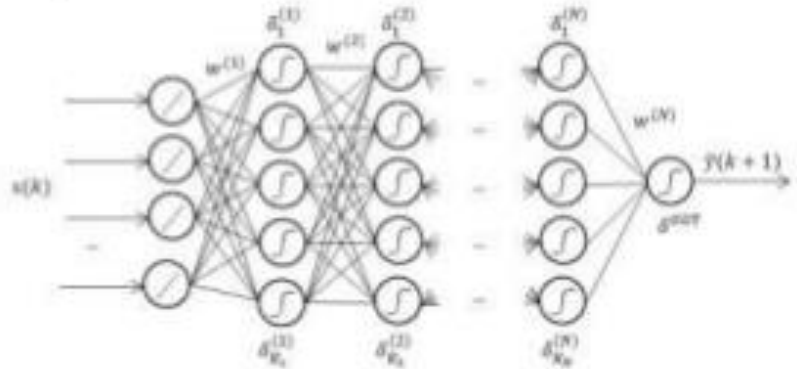


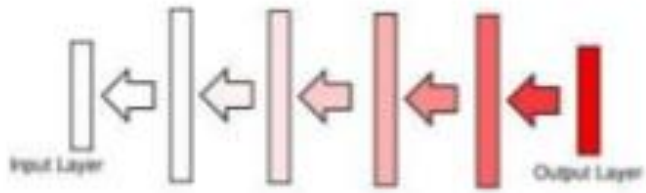
This function make farter layer from output small amplitude
We called “Gradient Vanishing Problem”

Bad effect of vanishing (exploding) gradients: a problem





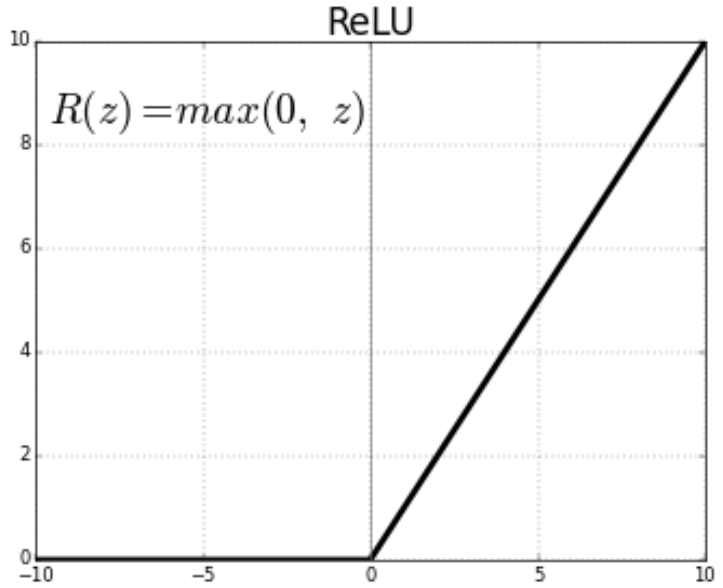




$$\delta_j^{(m)} = f_j^{(m-1)\prime} \sum_i w_{ij}^{(m)} \delta_i^{(m+1)}, \qquad \Rightarrow \qquad \frac{\partial E(k)}{\partial w_{ji}^{(m)}} \to 0 \ \ for\, m \to 1$$

$$\frac{\partial E(k)}{\partial w_{ji}^{(m)}} = \delta_j^{(m)} z_i^{(m-1)},$$

We use new invented activation function “ReLU” (Rectified Linear Units)

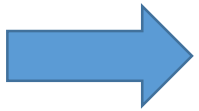


If x > 0 y = x
else y = 0
what it so easy?

- Invented from biological motivation
- But It can describe by proof math … ummm no!
- Let change our code

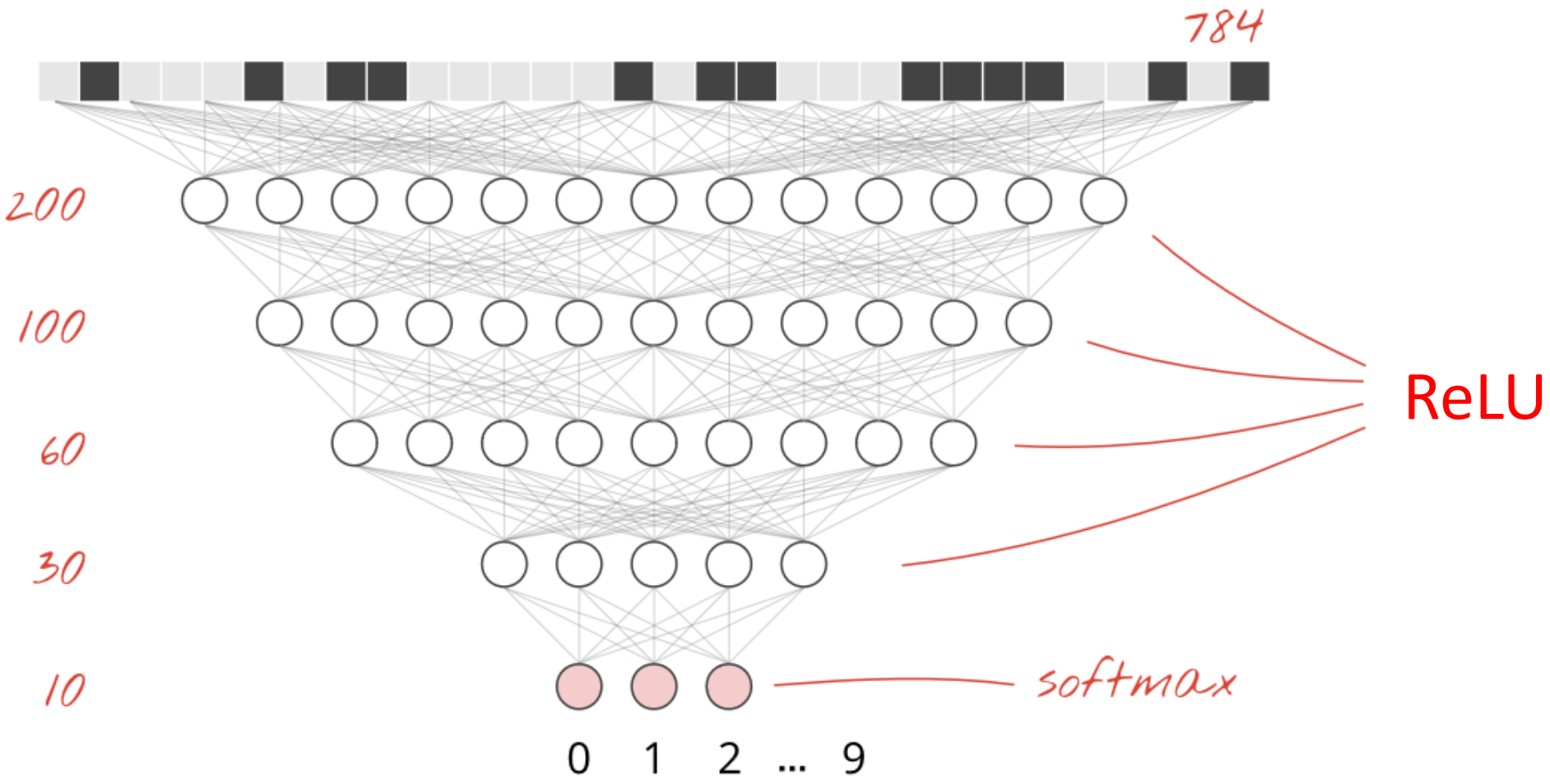
From sigmoid

```
#model
fc1 = tf.nn.sigmoid(tf.matmul(X,   W1) + b1)
fc2 = tf.nn.sigmoid(tf.matmul(fc1, W2) + b2)
fc3 = tf.nn.sigmoid(tf.matmul(fc2, W3) + b3)
fc4 = tf.nn.sigmoid(tf.matmul(fc3, W4) + b4)
Ylogits = tf.matmul(fc4, W5) + b5
```

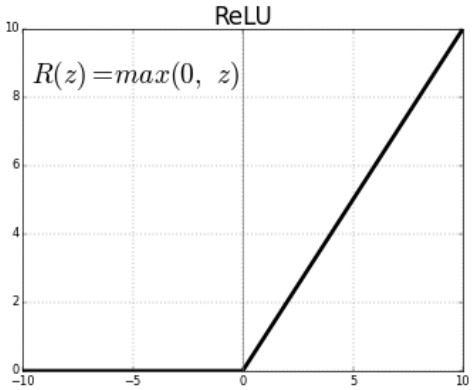


to relu

```
#model
fc1 = tf.nn.relu(tf.matmul(X,   W1) + b1)
fc2 = tf.nn.relu(tf.matmul(fc1, W2) + b2)
fc3 = tf.nn.relu(tf.matmul(fc2, W3) + b3)
fc4 = tf.nn.relu(tf.matmul(fc3, W4) + b4)
Ylogits = tf.matmul(fc4, W5) + b5
```


784
200
100
60
30
10
softmax
0 1 2 ... 9

ReLU


ReLU
$R(z) = max(0,\ z)$
10
8
6
4
2
0
−10
−5
0
5
10

Run & open tensorboard
tensorboard –logdir=“logs”
and see what happen!