Notes on MIT Introduction to Deep Learning

Tianzong Cheng

December 5, 2023

Preface

Thanks to Professor Ban for introducing this lecture to me.

Tianzong Cheng
December 5, 2023

Contents

1 Lecture 1

1.1 Perceptron

$$\hat{y} = g(w_0 + \sum_{i=1}^{m} x_i w_i) \tag{1}$$

x_i is the input, w_i is the weight of each input, w_0 is the bias term, and g is a non-linear activation funcion.

Or we can rewrite the equation in linear algebra language.

$$\hat{y} = g(w_0 + X^T W) \tag{2}$$

where: $X = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$ and $W = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$

An example of the activation function: sigmoid function. It can be intepreted as a continous version of the threshold function.

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

In TensorFlow, we can access this function by:

```
tf.math.sigmoid(z)
```

1.2 Building Neural Networks with Perceptrons

Perceptron: dot product, bias, non-linearity.

Because all inputs are densly connected to all outputs, these layers are called **Dense** layers.

The first layer, which is a hidden layer can be expressed as follows:

$$z_i = w_{0,i}^{(1)} + \sum_{j=1}^{m} x_j w_{j,i}^{(1)} \tag{4}$$

Then, the second layer, which calculates the final outputs, can be expressed as follows:

$$\hat{y}_i = g(w_{0,i}^{(2)} + \sum_{j=1}^{m} g(z_j) w_{j,i}^{(2)}) \tag{5}$$

We can express the whole model using two lines of codes with the help of TensorFlow:

```
import tensorflow as tf
model = tf.keras.Sequential([
    tf.keras.layers.Dense(n),
    tf.keras.layers.Dense(2)
])
```

By stacking these layers on top of each other, we create a sequential model.