Principles of Robot Autonomy II

Neural networks and Tensorflow tutorial
Single layer neural network

Original perceptron: binary inputs, binary output

\[ y_1^i = f(x^i w_1 + b_1) \]
\[ y_2^i = f(x^i w_2 + b_2) \]
\[ y_3^i = f(x^i w_3 + b_3) \]
\[ y_4^i = f(x^i w_4 + b_4) \]

\[ y = f(xW + b) \]
Multi-layer neural network

Also known as the Multilayer Perceptron (MLP)
Also known as the foundations of DEEP LEARNING

$$h_1 = f_1(xW_1 + b_1)$$
$$h_2 = f_2(h_1W_2 + b_2)$$
$$y = f_3(h_2W_3 + b_3)$$

Building blocks: fully-connected, convolutional layers, recurrent layers, ...
Activation functions

Can’t go only linear:

\[ y = ((xW_1 + b_1)W_2 + b_2)W_3 + b_3? \]

\[ \implies y = xW_1W_2W_3 + (b_1W_2W_3 + b_2W_3 + b_3) \]

Sigmoid

\[ \sigma(x) = \frac{1}{1 + e^{-x}} \]

Leaky ReLU

max(0.1x, x)

Secret theme:

All of these functions are super easy to differentiate
Training neural networks

We want to use some variant of gradient descent

How to compute gradients?

1. Sample a batch of data
2. Forward propagate it through the network to compute loss
3. Backpropagate to calculate the gradient of the loss with respect to the weights/biases
4. Update these parameters using SGD

The Chain Rule

\[ \nabla (f \circ g)(x) = (Dg)(x)^T (\nabla f)(g(x)) \]

Leveraging the intermediate results of forward propagation with “easy” to differentiate activation functions

\[ \implies \text{Gradient is a bunch of matrix multiplications} \]
Backpropagation

Consider the function $L(x, y) = g(f(x, y))$
Backpropagation

Consider the function $L(x, y) = g(f(x, y))$
Consider the function \( L(x, y) = g(f(x, y)) \)
Training a simple model

Consider the parametric model \( f(x) = (x + a)(x + b) \)
trained with \( L_2 \) loss \( L_i = (y_i - f(x_i))^2 \)
Training a simple model

Consider the parametric model $f(x) = (x + a)(x + b)$ trained with $L_2$ loss $L_i = (y_i - f(x_i))^2$
Training a simple model

How many operations in the forward and backward passes?

5 in forward
4 in backward
Training neural networks

Lots of regularization tricks:

**Dropout:**
(randomly zero out some neurons each pass)

Transform input data to artificially expand training set:
Tensorflow tutorial

• Tensorflow: software library that provides tools to train and evaluate deep learning models.

• Keras: high-level API written on top of Tensorflow that offers user-friendly interfaces to create neural networks.

• Link to Colab: https://colab.research.google.com/drive/1ZnjSyQRRrTU4TUENxewogTCKvKxQFPwL?usp=sharing