EE 628
Deep Learning
Fall 2019

Lecture 4

02/13/2019

Assistant Prof. Sergul Aydore

Department of Electrical and Computer Engineering

STEVENS INSTITUTE of TECHNOLOGY

THE INNOVATION UNIVERSITY®

Overview

- Last lecture we covered
  - Softmax Regression
- Today, we will cover
  - Multilayer perceptron
  - Overfitting/underfitting

Multilayer Perceptrons

- The simplest deep networks

Multilayer Perceptrons

- The simplest deep networks
- They consist of many layers of neurons

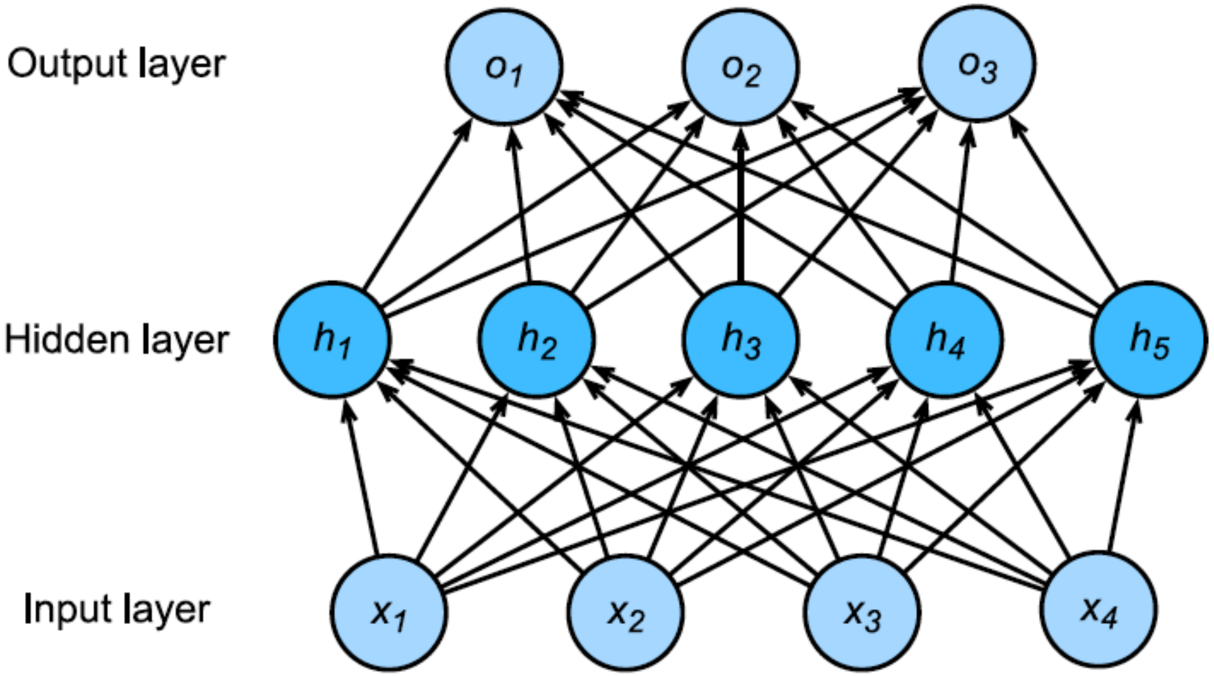

Fig. 6.1.2: Multilayer perceptron with hidden layers. This example contains a hidden layer with 5 hidden units in it.

From Linear to Nonlinear

- In order to get a benefit from multilayer architectures, we need another key ingredient– a nonlinearity σ to be applied to each of the hidden units after each layer's linear transformation

From Linear to Nonlinear

- In order to get a benefit from multilayer architectures, we need another key ingredient– a nonlinearity σ to be applied to each of the hidden units after each layer's linear transformation
- The most popular choice for the nonlinearity these days is the rectified linear unit (ReLU) $\max(x, 0)$.

From Linear to Nonlinear

- In order to get a benefit from multilayer architectures, we need another key ingredient– a nonlinearity σ to be applied to each of the hidden units after each layer's linear transformation
- The most popular choice for the nonlinearity these days is the rectified linear unit (ReLU) $\max(x, 0)$.
- After incorporating these no-linearities it becomes possible to merge layers

$$\mathbf{h} = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{o} = \mathbf{W}_2\mathbf{h} + \mathbf{b}_2$$
$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o})$$

From Linear to Nonlinear

- In order to get a benefit from multilayer architectures, we need another key ingredient– a nonlinearity σ to be applied to each of the hidden units after each layer's linear transformation
- The most popular choice for the nonlinearity these days is the rectified linear unit (ReLU) $\max(x, 0)$.
- After incorporating these no-linearities it becomes possible to merge layers

$$\mathbf{h} = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$$
$$\mathbf{o} = \mathbf{W}_2\mathbf{h} + \mathbf{b}_2$$
$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o})$$

- Clearly, we can continue stacking such hidden layers.

Vectorization and mini-batch

- As before, by the matrix $\mathbf{X}$, we denote a mini-batch of inputs.

Vectorization and mini-batch

- As before, by the matrix $\mathbf{X}$, we denote a mini-batch of inputs.
- The calculations to produce outputs from an MLP with two hidden layers can thus be expressed:

$$\mathbf{H}_1 = \sigma(\mathbf{X}\mathbf{W}_1 + \mathbf{b}_1)$$
$$\mathbf{H}_2 = \sigma(\mathbf{H}_1\mathbf{W}_2 + \mathbf{b}_2)$$
$$\hat{\mathbf{Y}} = \mathrm{softmax}(\mathbf{H}_2\mathbf{W}_3 + \mathbf{b}_3)$$

Activation Functions

- notebook

Concise Implementation of MLP

- notebook