ISTA 421/521
Introduction to Machine Learning

Lecture 25:
Neural Networks and Deep Learning

Leon Palafox

leonp@lpl.arizona.edu

2 December 2014

Introduction to Artificial Neural Networks

(NNs or ANN)

Objective

- By the end of the lecture, you'll be able to implement and use state of the art Machine Learning techniques in your own work.
- You'll have a working understanding of what is the current motor behind many industry search and classification engines.

Background

- NNs have gone through a heavy rebranding thorough the years.
- In 1943, McCulloch and Pitts created the first model of an artificial neuron.
- By 1958, Rosenblatt had come up with the Perceptron, the cornerstone of modern NN.
- In 1986, Rumelhart started the connectionism euphoria.

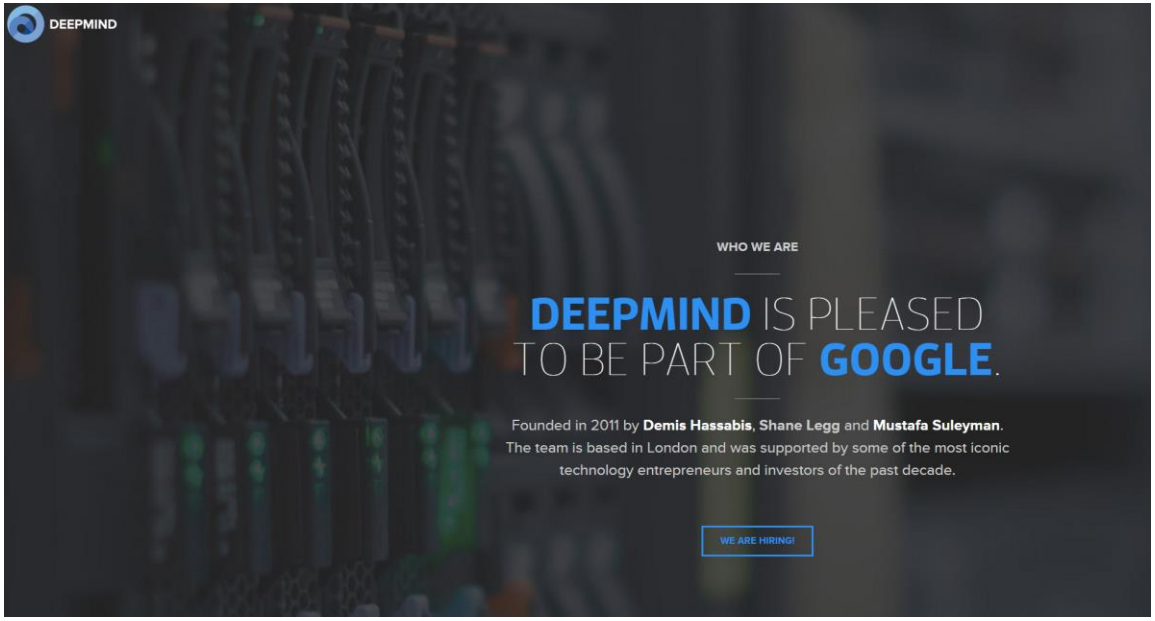
Background

- Processing power was still an issue and until 2006, common NNs were researched by only small clusters of people.
- Training was expensive, and the results only marginally better (or worse) than SVMs or Logistic Regression.
- In 2006, Hinton and Bengio made huge discoveries on how to train NNs and they rebranded them as Deep Nets.
- During this time, Convolutional Neural Networks (CNN) had been a great tool for image pattern recognition.

Motivation

- Deep Nets and CNNs, are by today standards the best algorithm for Image Pattern Recognition.
- The three Big Kahunas of NNs and Deep Nets, Geoffrey Hinton, Yann LeCun and Yoshua Bengio are working actively with Google, Facebook and University of Toronto, respectively.

Motivation



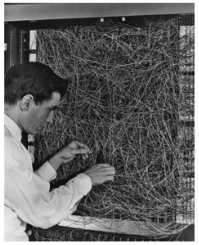
- In January Google bought DeepMind, a startup with no WebPage, no Product, a single NIPS (AI conference) Demo.
- They bought it for $500 million.
- Facebook was deeply interested as well.

NOVEMBER 25, 2012

IS "DEEP LEARNING" A REVOLUTION IN ARTIFICIAL INTELLIGENCE?

BY GARY MARCUS

NEWS DESK



Can a new technique known as deep learning revolutionize artificial intelligence, as yesterday's front-page article at the New York *Times* suggests? There is good reason to be excited about deep learning, a sophisticated "machine learning" algorithm that far exceeds many of its predecessors in its abilities to recognize syllables and images. But there's also good reason to be skeptical. While the *Times* reports that "advances in an artificial intelligence technology that can recognize patterns offer the possibility of machines that perform human activities like seeing, listening and thinking," deep learning takes us, at best, only a small step toward the creation of truly intelligent machines. Deep learning is important work, with immediate practical applications. But it's not as breathtaking as the front-

日本リアルタイム
JAPANREALTIME

FUKUSHIMA WATCH | AUTOS | ECONOMY & BUSINESS | TECHNOLOGY | POLITICS & POLICY | LIFESTYLE & CULTURE

HOT TOPICS: BANK OF JAPAN SHINZO ABE DISPUTED TERRITORY NUCLEAR FUKUSHIMA DAIICHI AGING POPULATION CHANGE LANGUAGE: ENGLISH

8:47 pm JST Oct 1, 2014 JAPAN

NTT, Toyota Seek 'Deep Learning' Expertise

PREVIOUS Big Win for Japanese Banks in Myanmar

NEXT New Commemorative Coins Celebrate Shinkansen's 50th Anniversary

ARTICLE COMMENTS

DEEP LEARNING NTT PREFERRED NETWORKS TOYOTA MOTOR CORP

By TAKASHI MOCHIZUKI CONNECT

About Japan Real Time

Japan Real Time is a newsy, concise guide to what works, what doesn't and why in the one-time poster child for Asian

Scientists See Promise in Deep-Learning Programs

Hao Zhang/The New York Times

A voice recognition program translated a speech given by Richard F. Rashid, Microsoft's top scientist, into Mandarin Chinese.

By JOHN MARKOFF
Published: November 23, 2012

Using an artificial intelligence technique inspired by theories about how the brain recognizes patterns, technology companies are reporting startling gains in fields as diverse as computer vision, speech recognition and the identification of promising new molecules for designing drugs.

点击查看本文中文版。

Keith Penner

A student team led by the computer scientist Geoffrey E. Hinton used

The advances have led to widespread enthusiasm among researchers who design software to perform human activities like seeing, listening and thinking. They offer the promise of machines that converse with humans and perform tasks like driving cars and working in factories, raising the specter of automated robots that could replace human workers.

The technology, called deep learning, has already been put to use in services like Apple's Siri virtual personal assistant, which is based on Nuance Communications' speech recognition service, and in Google's Street View, which uses machine vision to identify specific addresses.

But what is new in recent months is the growing speed and accuracy of deep-learning programs, often called artificial neural networks or just "neural nets" for their resemblance to the neural connections in the brain.

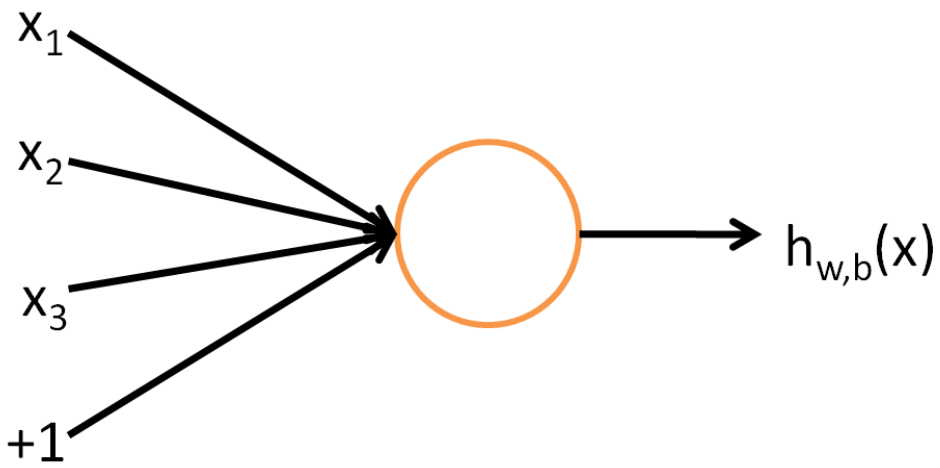
"There has been a number of stunning new results with deep-learning methods," said Yann LeCun, a computer scientist at New York University who did pioneering

MOST EMAILED RECOMMENDED FOR YOU

1. DEALBOOK Saks Flagship Store Is Appraised for Mortgage at $3.7 Billion
2. DEALBOOK U.S.-Backed Mortgages Put to Test in an Innovative Lawsuit
3. TODAY'S EDITORIALS Homeownership and Wealth Creation
4. ASK REAL ESTATE When Vermin Come to Visit
5. THE WORKING LIFE A Store Closes, but the Business Survives
6. DEALBOOK British Firm Starts Hedge Fund for Social Services
7. In Moscow, a Financial District in Name Only
8. On the Market in New York City

Perceptron

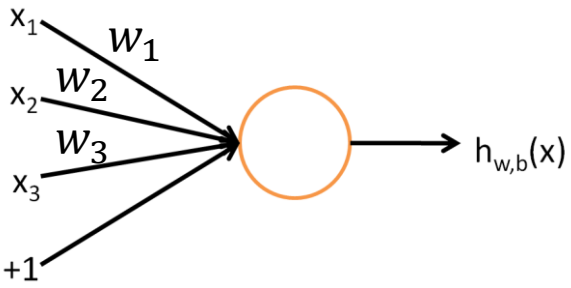
- Tries to mimic a real NN, since it has a nucleus that processes some inputs and give an output.



- $h_{w,b}(x)$ is a function of all the inputs, and is composed of two terms.

Perceptron

$$h_{w,b}(x) = f\left(\sum_{i=1}^{3} W_i x_i + b\right)$$



f is called the activation function, and it works as a way to discretize the outputs of the perceptron.

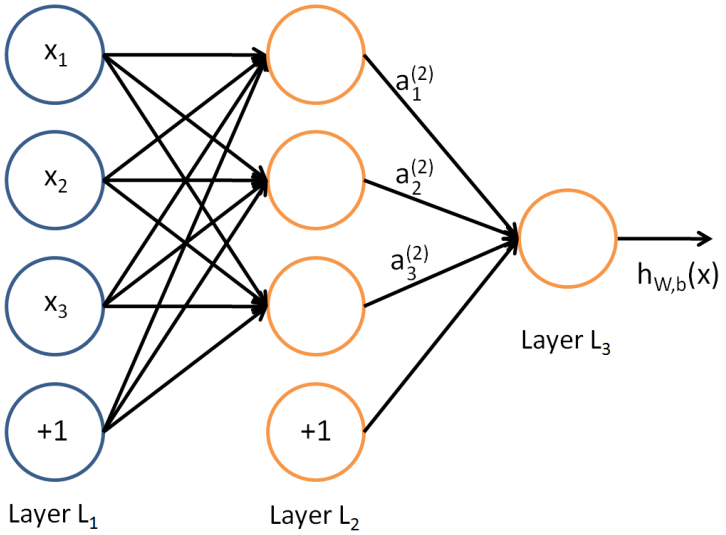
One of the most common activations functions is the sigmoid function:

$$f(z) = \frac{1}{1 + \exp(z)}$$

This looks very familiar

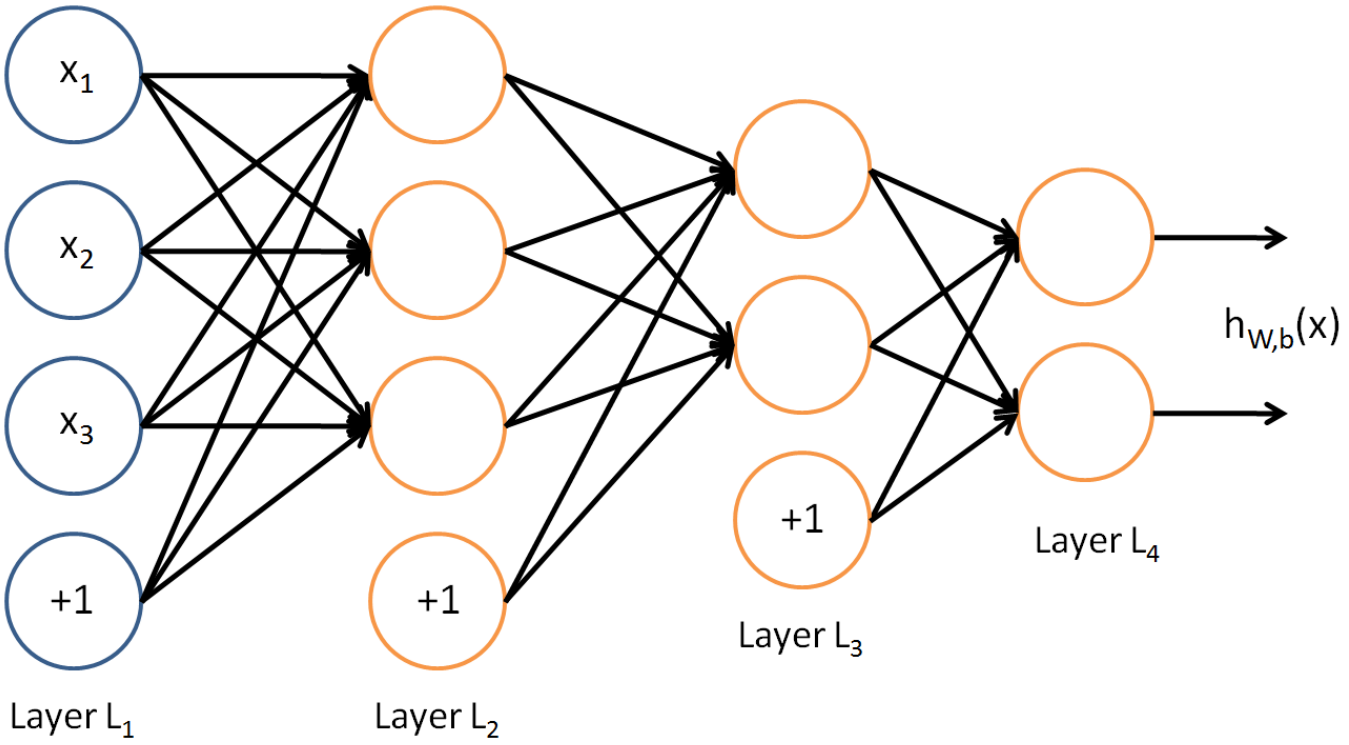
Neural Network

- Naturally, a NN is going to be a set of perceptrons interconnected within each other.



$$
\begin{aligned}
a_1^{(2)} &= f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \\
a_2^{(2)} &= f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \\
a_3^{(2)} &= f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)}) \\
h_{W,b}(x) &= a_1^{(3)} = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})
\end{aligned}
$$

Neural Network



- We can add as many layers and outputs as we want, for example a two binary output allows us to classify in four classes.
- We also regularize NNs, since they can be also prone to overfitting.

Training of Neural Networks

- The basic principle to train Neural Networks is Error Backpropagation.
  - We can find an error for a given input using the equations in the slide 11.
  - Next we go backwards in the network, and find the “share” of error each individual neuron has.
  - We calculate the derivative of this error to use Gradient Based techniques, like Gradient Descent.

Gradient Descent

- Given a cost function

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

- m is the number of examples and h some function of parameter θ.
- Gradient descent updates the parameter in steps:

$$\theta_j = \theta_j - \alpha\frac{\delta}{\delta\theta_j}J(\theta)$$

Gradient Descent for NNs

- The cost function for the overall network is:

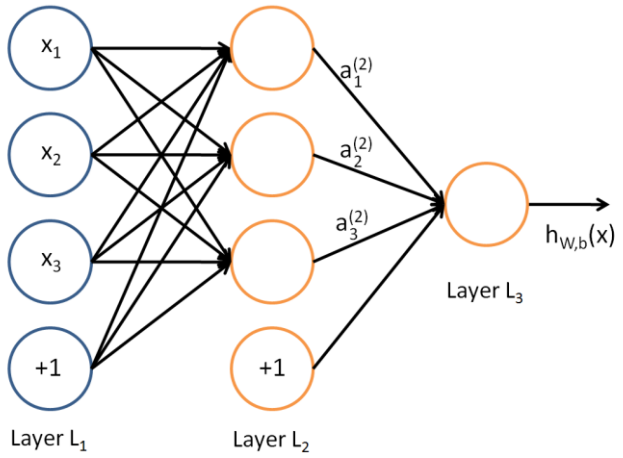
$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2$$

- Given the compact representation of the network:

$$\begin{aligned} z^{(2)} &= W^{(1)}x + b^{(1)} \\ a^{(2)} &= f(z^{(2)}) \\ z^{(3)} &= W^{(2)}a^{(2)} + b^{(2)} \\ h_{W,b}(x) &= a^{(3)} = f(z^{(3)}) \end{aligned}$$

In General

$$\begin{aligned} z^{(l+1)} &= W^{(l)}a^{(l)} + b^{(l)} \\ a^{(l+1)} &= f(z^{(l+1)}) \end{aligned}$$



Gradient Descent for NNs

- Gradient Descent: (Per Layer)

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b)$$

$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W, b)$$

With:

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b) = \left[\frac{1}{m} \sum_{i=1}^{m} \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x^{(i)}, y^{(i)}) \right] + \lambda W_{ij}^{(l)}$$

$$\frac{\partial}{\partial b_i^{(l)}} J(W, b) = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial}{\partial b_i^{(l)}} J(W, b; x^{(i)}, y^{(i)})$$

Gradient Descent for NNs

- We want to compute an “error term” δ, that will measure the error of a node I in layer ‘l’.

For the output layer:

$$\delta_i^{(n_l)} = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)})$$

For the middle layers

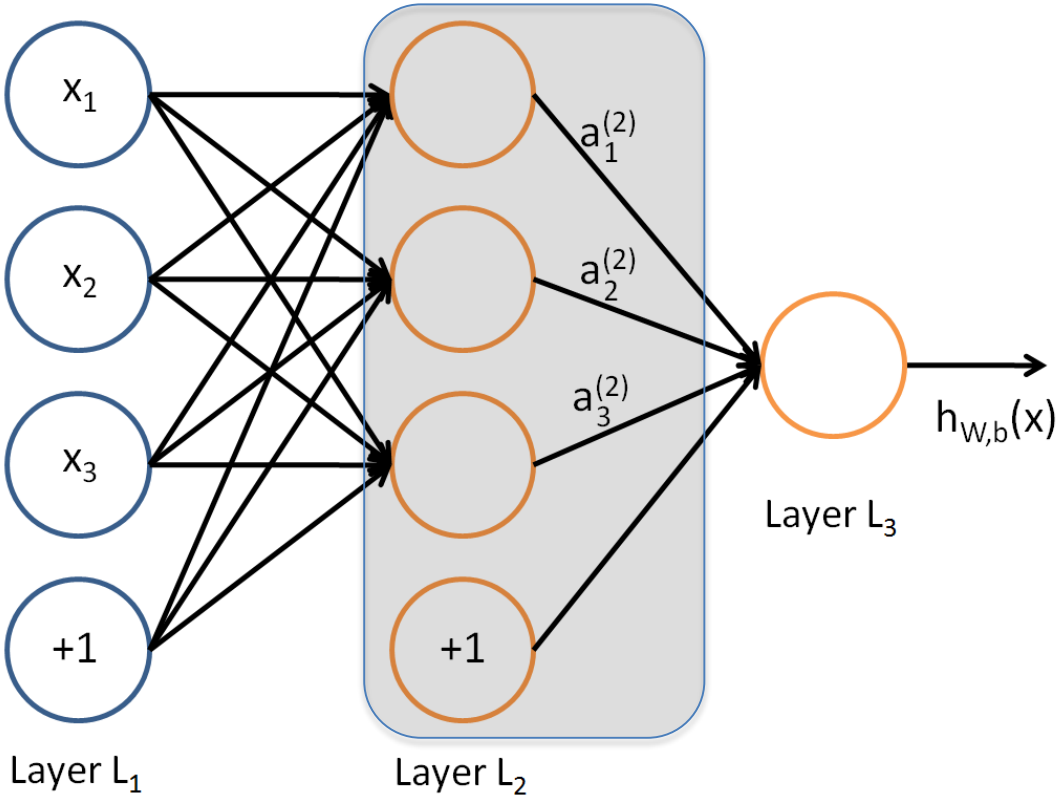
$$\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)})$$

Is a weighted average of all the errors related to this node

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b; x, y) = a_j^{(l)} \delta_i^{(l+1)}$$

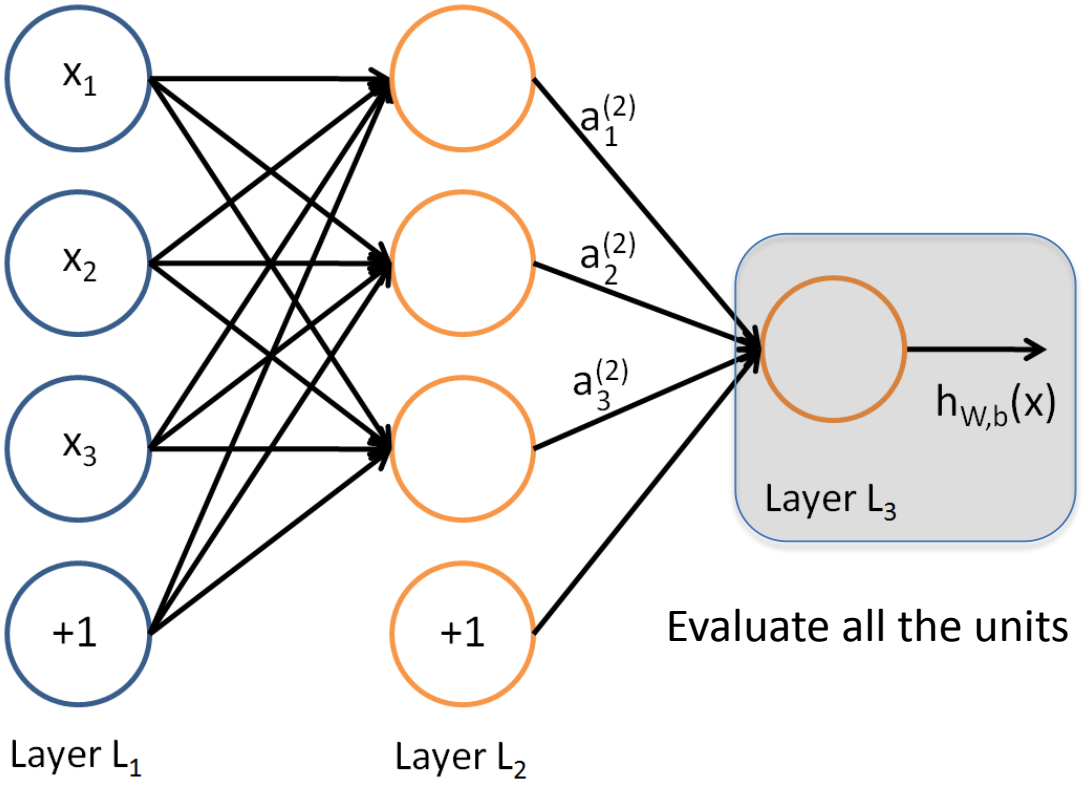
$$\frac{\partial}{\partial b_i^{(l)}} J(W, b; x, y) = \delta_i^{(l+1)}.$$

Feedforward-Backpropagation



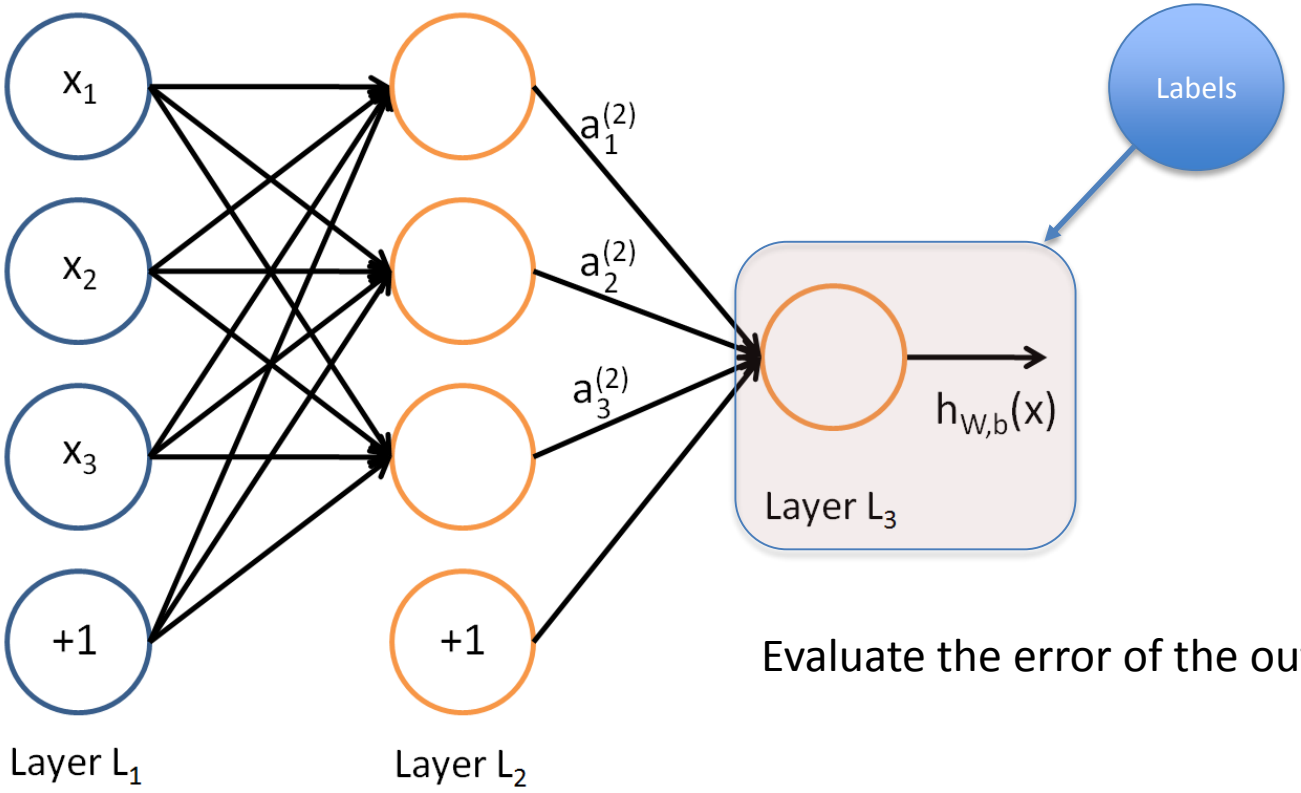
Evaluate all the units in Layer L2

Feedforward-Backpropagation



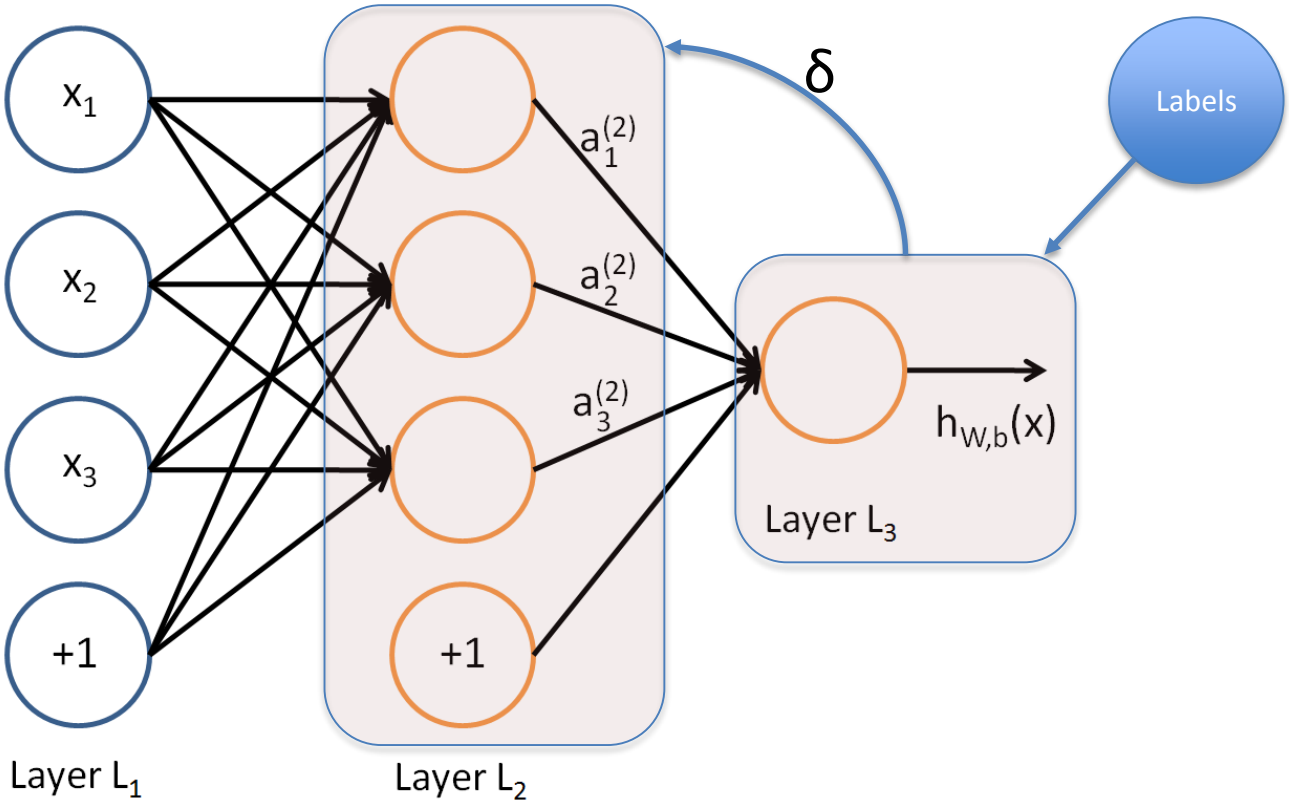
Evaluate all the units in Layer L3

Feedforward-Backpropagation



Evaluate the error of the output node (δ)

Feedforward-Backpropagation



Evaluate the errors of the middle layer nodes (δ)

Basic Intuitions behind NNs

- Instead of using a single classifier, we are using a bundle of them.
- Each classifier, via different weights will emphasize different parts of the input signal.
- For example, in an image, that could be shadows, or shapes.

Problems of NNs

- We need to answer two questions:
  - How many layers are enough to solve a problem?
  - How many hidden units should we use per layer?
- As you can imagine, training complexity increases as we increase hidden units.
  - This can be reduced by avoiding a full interconnection.
- The elephant in the room is called “Vanishing Gradient”

Vanishing Gradient

- A problem of NNs, is that our small δs, will become even smaller as we go back in our layers.
- If we have many layers, we are going to end up with really small gradients.
- This will show as negligible updates in the gradient descent equations.
- For many years, before 2006, this was the main reason few people used classic NNs.
  - What is the point of having the power that comes from many layers, if we cannot train it properly anyways?
  - The solution: Pre-training of the individual layers.

Autoencoders

- An