

---

# Chapter 4 (Part 3): Artificial Neural Networks

CS 536: Machine Learning  
Littman (Wu, TA)

## Artificial Neural Networks

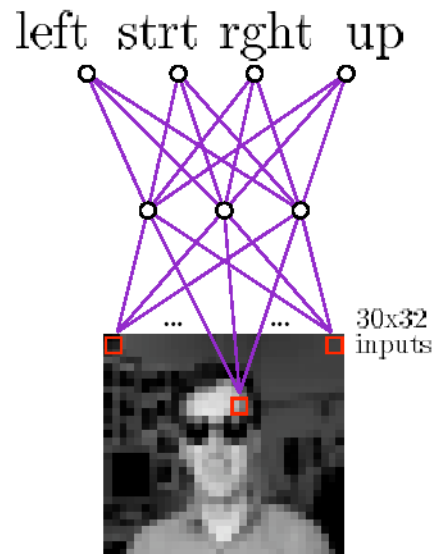
[Read Ch. 4]

[Review exercises 4.1, 4.2, 4.5, 4.9, 4.11]

- Threshold units [ok]
- Gradient descent [ok]
- Multilayer networks [ok]
- Backpropagation [ok]
- Hidden layer representations [ok]
- Example: Face Recognition [today]
- Advanced topics [today]

## Face Recognition

---



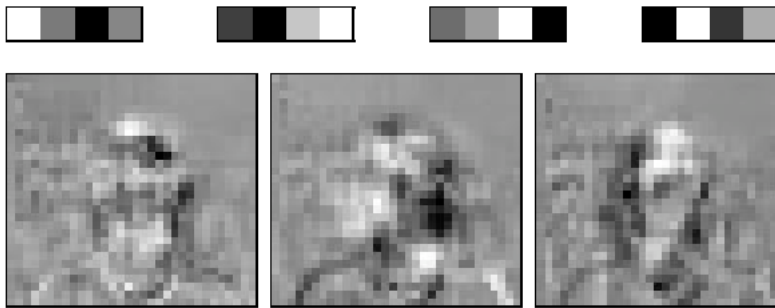
## Typical input images

---



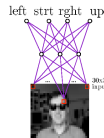
- 90% accurate learning head pose, and recognizing 1-of-20 faces

## Learned Weights



<http://www.cs.cmu.edu/tom/faces.html>

$w_0 w_1 w_2 w_3$   
For left, strtr, right, up



## Alternative Error Functions

Penalize large weights:

$$E(\mathbf{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in \text{outputs}} (t_{kd} - o_{kd})^2 + \lambda \sum_{ij} w_{ji}^2$$

Train on target slopes as well as values:

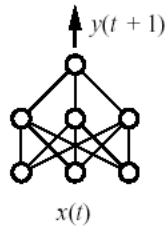
$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in \text{outputs}} \left[ (t_{kd} - o_{kd})^2 + \mu \sum_{j \in \text{inputs}} \left( \frac{\partial t_{kd}}{\partial x_d^j} - \frac{\partial o_{kd}}{\partial x_d^j} \right)^2 \right]$$

Tie together weights:

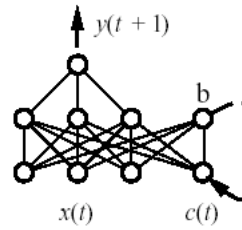
- e.g., in phoneme recognition network

# Recurrent Networks

---



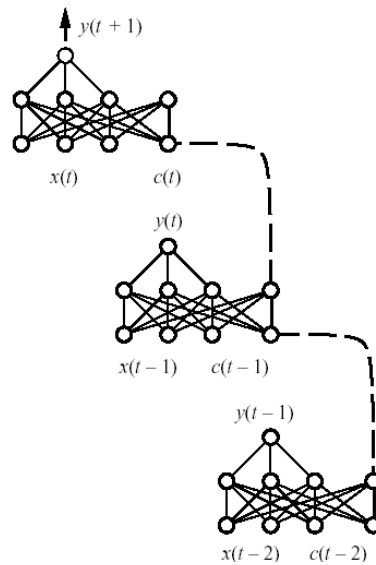
(a) Feedforward network



(b) Recurrent network

# Unfolding: BPTT

---



## Another Good Question

---

Ex. 4.6

## Project Methodology

---

Lots of good ideas for algorithms and domains.

The hard question is: “How will you evaluate it?”

Ultimately, you need to present more than one algorithm (and perhaps more than one problem) and you’ll need some way of saying what worked better.

***What’s the gold standard?***

## Simple Project Idea

Take an interesting dataset

Compare several learning approaches for prediction

- decision trees
- ANNs
- instance-based methods
- SVMs
- boosting

## Training/Testing Data

Getting labeled data is hard. Ideas:

- association: use some features to predict others. Example: spelling correction.
- prediction: use past features to predict future.
- existing data: UCI repository
- natural labels: newswire categories
- automatic checker: answer is tricky, but know if you're right.
- artificial data: generator has a known rule

## Datasets

---

Weather prediction data

Sensor data: <http://www.greatduckisland.net/>

Don Smith has log data on dialup usage  
(use weather, day, time, ...)

Other log data: network usage, CPU usage

GPS data (campus buses, my car)

Vision data (score a goal)

Bio data

## Text Classification

---

Easy to get lots of text: web, TREC  
data, email

Predict topic, authorship, sentiment,  
style, affect, attitude.

## Reinforcement Learning

---

Personal favorite

Can be challenging to do well: generate data or direct control

- Optimize gas well production
- Object tracking
- “Tag” grid world
- Rover control
- animal behavior experiments

Compare approaches (direct policy search, value function learning, model-based, ...)

## Active Learning

---

Take a classic dataset.

Explore the tradeoff between size of training set and generalization.

Devise schemes for choosing items to “pay” for labels to maximize accuracy with minimum cost.

## Theory Stuff

---

Any interest? I'm inclined to introduce instance-based methods next (Ch. 8).