

## Principles & Methods of Cognitive and Neural Modeling II Syllabus and Reading List, Spring 2009

**Instructor:** **Norbert Kopčo** **Office Hours:** Wednesdays, 2-3pm  
Room 304, 677 Beacon St. or by appointment  
(617) 353 8693  
[kopco@bu.edu](mailto:kopco@bu.edu) (best way to contact me)

**TF:** **Alex Storer** **Office Hours:** Mondays, 2-3pm  
Room 110, 677 Beacon St. or by appointment  
(617) 353 6426  
[storer@cns.bu.edu](mailto:storer@cns.bu.edu)

This is the CN520 syllabus, containing requirements and policies for the course as well as a detailed, week-by-week reading list and lecture summary. A schedule outline, below, is provided for your reference (subject to any changes announced in class).

Lectures will be held in room B03 at 677 Beacon St, Tuesdays: 5:00 p.m. – 8:00 p.m.

In the past this course was given by Prof. Barbara Shinn-Cunningham and Piers Howe who have generously allowed me to reuse their course materials.

---

### Weekly Schedule

<b>Date</b>	<b>Topic</b>
1/20	Introduction, McCulloch-Pitts neuron, Linear Discriminant Functions
1/27	Optimal Classifiers
2/3	Adaline and Perceptron ( <b>Proj 1 due: Optimal Classifiers</b> )
2/10	Basic backpropagation
2/17	No class (Substitute Monday schedule of classes)
2/24	Extensions of backpropagation ( <b>Proj 2 due: ADALINE</b> )
3/3	Radial basis function models ( <b>Proj 3 due: Backprop</b> )
3/10	No classes: Spring break
3/17	<b>Mid-Term Exam</b>
3/24	Introduction to associative learning ( <b>Proj 4 due: RBF</b> )
3/31	Competitive learning and Self-Organizing Feature Maps
4/7	Adaptive Resonance Theory ( <b>Proj 5 due: SOFM</b> )
4/14	The Hopfield autoassociator
4/21	Boltzmann Machine ( <b>Proj 6 due: Hopfield</b> )

4/28	Genetic Algorithms
5/5	<b>Final exam</b>

## Overview

This course introduces and analyzes ideas from various traditions in neural modeling, with an emphasis on the general topics of learning and pattern recognition. The course will provide a historical and comparative overview of several architectures, beginning with the pioneering work of McCulloch and Pitts (1943). We will cover models of supervised learning, unsupervised learning, content-addressable memory, associative learning, gradient-descent procedures, self-organizing feature maps and other related topics.

## Prerequisites

Projects require students to program mathematical algorithms and plot results using C, MATLAB, or other computer tools. Students lacking these skills must obtain the permission of the instructor to enroll.

## Grading scale and policy

All students must complete seven **projects** and two in-class **exams**.

**Projects** are designed to exercise and develop some of the skills that you will need as a researcher in Cognitive and Neural Modeling, and will include both computer programming and writing components. Projects will be graded on a 10-point scale based on clarity and content. Make sure to begin projects early. Most projects are due 2 weeks after the topic of the project is covered in class. *Factor these time constraints into your planning!*

**Exams** are meant to test your ability to think critically and apply what you have learned in class. Both the exams are given equal weight in determining your final grade.

## Final course grade

The final course grade will depend on all aspects of the coursework:

**60%** Six homework projects (each project is **10%** of the final grade)

**20%** Mid-term exam

**20%** The final exam

## Late projects and make-up policies

All homework projects are due no later than 5:00 p.m. (start of class) on the date indicated above, or the modified date, if changes to the due dates are announced later. *Projects turned in more than 15 minutes after the start of class are considered late.*

**Projects** turned in up to 2 weeks after their due date are worth at most 60% credit. No projects will be accepted for credit more than two weeks after their due date. All late projects must be turned in by the *second-to-last* week of classes. Any late projects turned in beyond

that date will not be graded until after all regular projects and final exams have been graded. In most cases, this will mean that the student may receive an “I” grade in the course, which will be changed to a regular grade sometime after the end of the semester. Any project due during the last two weeks of classes can be turned in by the date of the final exam and it will be graded for inclusion in the final course grade.

**Exams** cannot be rescheduled except in the event of an *unforeseeable, documented emergency*. If you miss an exam or oral presentation because of an emergency, it is your responsibility to inform me about the problem as *soon as can be reasonably expected*.

### **Policy on incomplete grades**

In the event that you are unable to complete the course requirements, *you must contact me no later than the day of the last lecture* to discuss the *possibility* of taking an “I” (incomplete) grade in the course. In general I am very reluctant to issue “incompletes”. If I have not received a formal, written request (e-mail is adequate *if you receive an acknowledgement*), your final course grade will be based on the work available to me. If an “I” is granted, any work turned in late will receive 90% of its “on-time” grade. *If you take an incomplete in the course, you will not be guaranteed an “A” in the course, even if you do a good job on all the missing assignments.*

When requesting that an “I” grade be issued, please indicate (1) what work remains to be completed, (2) why you cannot complete it on time and (3) a firm timetable for completion of this work. An “I” will be granted only in exceptional circumstances and only if the timetable is shorter than 3 months. In general, the “I” grade will be turned into a final grade by the final date on the accepted timetable, based on all the work received up to that date.

### **Lecture notes and readings**

PDFs of all lecture notes will be available on the web. It is your responsibility to print out the lecture notes and bring them to lecture if you wish to take notes on them. All readings will also be available on the web. The CNS department printers should NOT be used to print out your lecture notes to make copies of readings. The BU Bookstore sells the required textbook and many of the recommended textbooks. The CNS library also has copies of most of these texts.

#### **Required book:**

Duda, Hart, and Stork (2001). *Pattern Classification*, 2<sup>nd</sup> Edition, New York: Wiley Interscience. Referred to in the notes as **DHS**.

#### **Supplementary books:**

Anderson, James A. and Edward Rosenfeld (1988). *Neurocomputing: Foundations of Research*, Cambridge, MA: MIT Press. Referred to in the notes as **NFR**.

Hertz, John, Anders Krogh, and Richard G. Palmer (1991). *Introduction to the Theory of Neural Computation*, New York: Addison-Wesley. Referred to in the notes as **HKP**.

### **CN520 web site**

The official CNS520 web site can be accessed at <http://cns.bu.edu/~kopco/cn520>. Portions of the website are password protected and should only be accessed for your personal

use (for copyright reasons). To obtain access, contact the instructor or the TF (or, if you have an account on the CNS department linux cluster, login to any of the machines, e.g., to kenmore.bu.edu, and type `cat ~kopco/cn520` to obtain the password)

## **Policy on the use of email**

On the first day of class, a signup sheet will be passed around to get your email address. If you miss this signup, you should email your email address to the instructor ([kopco@bu.edu](mailto:kopco@bu.edu)). It is your responsibility to ensure that your email address is correctly registered and that you are receiving all CN520 notices.

Although the use of email is encouraged for contacting either the instructor or the TF, it is always a good idea to double-check that your mail was received, especially if the message is important and/or you do not receive a reply within 24 hours (excluding weekends). Also, try to avoid sending lengthy or sensitive messages by email, as this means of communication often fails to convey subtleties apparent in face-to-face or even voice communications and can lead to misunderstandings.

## **Policy on attendance**

Students are expected to attend lectures and exams. In the case of an emergency or other extreme circumstances, it is the students responsibility to contact another student, determine what material / announcements they missed, and take appropriate action to catch up, as necessary. Any student who misses more than three of the class meetings must meet with the instructor to explain why; if there are no appropriate extenuating circumstances, that student's final grade will drop by one mark (e.g., from a B+ to a B).

## **Policy on collaboration**

Students may discuss lectures, readings etc. but must not collaborate or otherwise help each other with the projects. These projects must be entirely the student's own work. In particular, under no circumstances may students share any computer code. Each student must turn in project results that they generated using their own code. Any case of suspected academic misconduct will be referred to the dean's office. Any work deemed by the dean to be plagiarized will be assigned a failing grade. If a student requires clarification of the requirements of a project he/she should not contact the other students but instead should contact either the TF or the instructor.

## **Lectures**

### **January 20            McCulloch-Pitts neuron, Linear discriminant functions**

*This lecture provides an overview of the course and begins to trace the historical development of some classical neural network (NNet) models. We begin with the McCulloch-Pitts (MP) neuron model (1943) which is a precursor to almost all modern NNets. We analyze what a McCulloch-Pitts neuron can compute, which leads to a description of linear discriminant functions.*

**Readings:**

**HKP** Introduction chapter.

This chapter provides a nice historic overview of the field of neural modeling. Reading it is highly recommended.

**NFR**, Chapt 2. McCulloch, W., and W. Pitts (1943). A logical calculus of the ideas immanent in nervous system activity.

This article introduces the famous McCulloch-Pitts Neuron. You will not be required to know the contents of this article other than what you are taught in class (it requires knowledge of formal generative languages); however, it may amuse you to leaf through it. This is a good article with which to be familiar if only for historical interest.

**January 27                      Optimal decision theory, Gaussian classifiers**

*In this lecture, we discuss some of the basics of traditional pattern recognition theory and optimal classification in order to develop better insight into the performance of neural networks and how they perform similar tasks. We then discuss linear discriminant functions and their abilities and limitations.*

**Essential Readings:**

**DHS**, Chapt 1 and 2

**February 3                      Adaline and Perceptron**

*This lecture examines the Perceptron and the Adaline (LMS algorithm). The two models are compared and contrasted as two different approaches to statistical pattern recognition. Proofs of convergence will be offered for both methods. We then examine the work of Minsky and Papert in which properties and limitations of the simple Perceptron are analyzed.*

**Essential Readings:**

**DHS**, Chapt 5.

**Optional readings:**

**NFR**, Chapt 8. Rosenblatt, F. (1958). The Perceptron: A probabilistic model for information storage and organization in the brain. Read through page 98.

The first two sections give a nice qualitative overview of the Perceptron.

**NFR**, Chapt 13. Minsky, M. and S. Papert (1988). Introduction, in *Perceptrons (2nd Edition)*, Cambridge, MA: MIT Press.

This book has been accused of bringing NNet research to a standstill when first published in 1969. However subsequent advances addressed the issues raised in the original edition.

## **February 10            Basic Backpropagation**

*The generalized delta rule solves the problem of credit assignment for a multi-layer Perceptron, allowing the creation of a network that can handle non-linearly-separable categorization tasks. Because of its impact on the present-day NNet community and because of its widespread use, we will look at this architecture in some detail. This lecture focuses on the theory behind backpropagation (backprop).*

### **Essential Readings:**

**DHS**, Chapt 6, Sections 6.1 – 6.9.

### **Optional Readings:**

**NFR**, Chapt 41. D. Rumelhart, G. Hinton, and R. Williams (1986). (Optional) Learning internal representations by error propagation, in *Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Volume 1*, Rumelhart and McClelland (eds), Cambridge, MA: MIT Press.

This is the “classic” backprop paper. It is long, but the most important stuff is in the first half. Go over the derivation and read the first few examples carefully (XOR and parity problems).

## **February 17            No class (substitute Monday schedule of classes)**

## **February 24            Extensions of Backpropagation**

*Backpropagation is both powerful and ubiquitous. Today’s lecture extends the basic backprop approach to encompass temporal processing problems. These approaches are useful in the development of control applications and for prediction of temporal sequences.*

### **Readings:**

**DHS**, Chapt 6, Sects. 6.10.3 – end

Wan, E. A.(1990). Temporal backpropagation for FIR neural networks. *IEEE International Joint Conference on Neural Networks*, Volume 1, 575-580. San Diego, CA.

A description of extending backprop to FIR neurons.

**NFR**, Chapt 40. Sejnowski, T. and C. Rosenberg. (1986). NETtalk: A parallel network that learns to read aloud.

This is probably the most famous application of backprop. It is greatly responsible for the wave of enthusiasm for neural networks that developed in the eighties.

### **March 3                      Radial Basis Function Models**

*Radial basis function (RBF) models are very similar to networks we have already studied. The main difference is that they explicitly break down a non-linearly-separable problem into two components-- one nonlinear and one linear. These two components are implemented in series, so that the network has a slightly more formalized structure. RBF networks are used both as classifiers and as a way of fitting arbitrary functions. We examine both types of RBF networks and their use in real applications.*

#### **Essential Readings:**

DHS, Chapt 6, Sects 6.10.1 – 6.10.2

#### **Optional Readings:**

Moody, J., and C. J. Darken (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, **1**, 281-294.

One of the original papers on RBFs.

Wetterschereck, D. and T. Dietterich (1992). Improving the performance of radial basis function networks by learning center locations, in *Advances in Neural Information Processing Systems 4*, (J. E. Moody, S. J. Hanson, and R. P. Lippman, eds)., 1133-1140, San Mateo, CA: Morgan Kaufmann.

A comparison of an RBF network and the original multilayer Perceptron approach to the NETtalk experiment.

Jenison, R. L. and Fissell, K. (1996). A spherical basis function neural network for modeling auditory space, *Neural Computation*, **8**(1), 115-128.

A beautiful example using RBF networks to fit complex curves that represent the spatial information reaching a listener as a function of the position of a sound source.

**March 10                      No classes: Spring break**

**March 17                      Mid-Term Exam**

**March 24                      Introduction to Associative Learning**

*After an introduction to associative memory and a description of Kosko's Bidirectional Associative Memory model, we analyze associative learning and memory, examining Kohonen's Optimal Associative Memories. The readings outline a mainstream approach to*

associative memory emphasizing linear hetero- and auto-associators. The lecture concludes by relating correlation- and error-based models.

**Readings:**

Kosko, B. (1987). Constructing an associative memory. *Byte*, 12(10), 137-144.

This article contains easily implemented pseudocode for constructing a bidirectional associative memory (BAM), one of the simplest examples of associative memory models.

Kohonen, Teuvo (1989). *Self-organization and Associative Memory*, New York: Springer-Verlag. Read Chapters 1 - 6 (through section 6.5).

This reading develops the ideas of associative memory from a system theoretic viewpoint.

**March 31                      Competitive Learning, Self-Organizing Feature Maps**

*We introduce a form of learning known as competitive learning. After a discussion of the relationship of competitive learning and Principal Components Analysis (PCA), we introduce a famous model of competitive learning known as the self-organizing feature map (SOFM).*

**Readings:**

**DHS**, Chapt 10, Sections 10.13 - end.

**NFR**, Chapt 30. Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, **43**, 59-69.

This reading introduces Kohonen's SOFM model, which you must implement.

Angeniol, B., G. de La Croix Vaubois, and J. Le Texier (1988). Self-organizing feature maps and the traveling salesman problem. *Neural Networks*, **1**, 289-293.

This is a short and clear article that illustrates that the traveling salesman problem can be solved with SOFMs better than with the continuous symmetric autoassociator.

**April 7                              Clustering and Adaptive Resonance Theory**

*We examine various methods for clustering data and end with an overview of Adaptive Resonance Theory (ART).*

**Readings:**

**DHS**, Chapt 10, Sections 10.1 -10.12.

Carpenter, G.A. (2001). Neural-network models of learning and memory: Leading questions and an emerging framework, *Trends in Cognitive Science*, **5**(3), 114-118.

Carpenter, G.A., Grossberg, S., & Rosen, D.B. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4, 759-771.

## **April 14                      The Hopfield Autoassociator**

*After introducing associative learning in the last lecture, we now turn to mathematical analyses of associative learning and memory with an emphasis on Hopfield's autoassociator model. We also look at Hopfield and Tank's solution to the Traveling Salesman Problem.*

### **Readings:**

Hopfield, J. and D. Tank (1985). "Neural" computation in decisions in optimization problems. *Biological Cybernetics*, **52**, 141-152.

One of the classical applications of the symmetrical autoassociator.

## **April 21                      Boltzmann Machine**

*We now look at another approach to optimization within the framework of simulated annealing. We also study an application of simulated annealing to neural networks in the Boltzmann machine.*

### **Readings:**

**DHS**, Chapt 7, Sections 7.1-7.4.

**NFR**, Chapt 33. Kirkpatrick, S., C. Gelatt, and M. Vecchi (1983). Optimization by simulated annealing. *Science*, **220**, 671-680.

This paper introduces the concept of *simulated annealing*. Concentrate primarily on the first three sections (through **NFR** page 559) and on the last two sections (**NFR** pages 564-566).

**NFR**, Chapt 38. Ackley, D., G. Hinton, and T. Sejnowski (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, **9**, 147-169.

This article makes use of a Hopfield-like network along with simulated annealing, a technique from statistical mechanics, to achieve supervised learning in a multi-layer Perceptron.

## **April 28                      Genetic algorithms**

*We begin this lecture by examining genetic algorithms (GAs) as a global search technique. We also look at an application of GAs to the Traveling Salesman Problem.*

### **Required Readings (to be done before the beginning of the lecture):**

DHS, Chapt 7, Section 7.5.

Houssoun, M. H. (1995). *Fundamentals of Artificial Neural Networks*, Cambridge, MA: MIT Press. Read sections 8.5 - 8.7.

Whitley, D. T. Starkweather, and D. Fuquay (1987). Scheduling problems and traveling salesmen: The genetic edge recombination operator. *Proc 2nd Int Conf Genetic Alg*, 133-140. Read through page 137.

A discussion of the traveling salesman problem using genetic algorithms.

## **May 5                              Final Exam**