Contents

List of Figures xi
List of Tables xix
Preface xxi

1 Natural Computation 1
1.1 Introduction 1
1.2 The Brain 2
1.2.1 Subsystems 2
1.2.2 Maps 4
1.2.3 Neurons 4
1.3 Computational Theory 6
1.4 Elements of Natural Computation 9
1.4.1 Minimum Description Length 10
1.4.2 Learning 14
1.4.3 Architectures 16
1.5 Overview 18
1.5.1 Core Concepts 19
1.5.2 Learning to React: Memories 19
1.5.3 Learning During a Lifetime: Programs 20
1.5.4 Learning Across Generations: Architectures 20
1.6 The Grand Challenge 21
Notes 21
Exercises 22

I Core Concepts 25

2 Fitness 29
2.1 Introduction 29
2.2 Bayes' Rule 31
2.3 Probability Distributions 33
2.3.1 Discrete Distributions 33
2.3.2 Continuous Distributions 35
2.4 Information Theory 37
2.4.1 Information Content and Channel Capacity 38
2.4.2 Entropy 39
2.4.3 Reversible Codes 41
2.5 Classification 44
2.6 Minimum Description Length 46

Appendix: Laws of Probability 48
Notes 50
Exercises 51

3 Programs 55
3.1 Introduction 55
3.2 Heuristic Search 60
3.2.1 The Eight-Puzzle 60
3.3 Two-Person Games 63
3.3.1 Minimax 64
3.3.2 Alpha and Beta Cutoffs 66
3.4 Biological State Spaces 67
Notes 68
Exercises 68

4 Data 71
4.1 Data Compression 71
4.2 Coordinate Systems 72
4.3 Eigenvalues and Eigenvectors 75
4.3.1 Eigenvalues of Positive Matrices 78
4.4 Random Vectors 80
4.4.1 Normal Distribution 82
4.4.2 Eigenvalues and Eigenvectors of the Covariance Matrix 82
4.5 High-Dimensional Spaces 84
4.6 Clustering 87
Appendix: Linear Algebra Review 88
Notes 92
Exercises 92

5 Dynamics 95
5.1 Overview 95
5.2 Linear Systems 98
5.2.1 The General Case 101
5.2.2 Intuitive Meaning of Eigenvalues and Eigenvectors 103
5.3 Nonlinear Systems 104
5.3.1 Linearizing a Nonlinear System 106
5.3.2 Lyapunov Stability 108
Appendix: Taylor Series 109
Notes 110
Exercises 110

6 Optimization 113
6.1 Introduction 113
6.2 Minimization Algorithms 115
6.3 The Method of Lagrange Multipliers 118
6.4 Optimal Control 121
6.4.1 The Euler-Lagrange Method 121
6.4.2 Dynamic Programming 127
Notes 130
Exercises 131

II Memories 135

7 Content-Addressable Memory 143
7.1 Introduction 143
7.2 Hopfield Memories 145
7.2.1 Stability 146
7.2.2 Lyapunov Stability 149
7.3 Kanerva Memories 151
7.3.1 Implementation 153
7.3.2 Performance of Kanerva Memories 155
7.3.3 Implementations of Kanerva Memories 157
7.4 Radial Basis Functions 159
7.5 Kalman Filtering 159
Notes 160
Exercises 161

8 Supervised Learning 163
8.1 Introduction 163
8.2 Perceptrons 164
8.3 Continuous Activation Functions 167
8.3.1 Unpacking the Notation 170
8.3.2 Generating the Solution 170
8.4 Recurrent Networks 174
8.5 Minimum Description Length 178
8.6 The Activation Function 179
8.6.1 Maximum Likelihood with Gaussian Errors 179
8.6.2 Error Functions 180
Notes 181
Exercises 182

9 Unsupervised Learning 185
9.1 Introduction 185
9.2 Principal Components 186
9.3 Competitive Learning 188
9.4 Topological Constraints 190
9.4.1 The Traveling Salesman Example 191
9.4.2 Natural Topologies 191
9.5 Supervised Competitive Learning 194
9.6 Multimodal Data 196
9.6.1 Initial Labeling Algorithm 197
9.6.2 Minimizing Disagreement 197
9.7 Independent Components 201
Notes 203
Exercises 204

III Programs 205

10 Markov Models 215
10.1 Introduction 215
10.2 Markov Models 216
10.2.1 Regular Chains 217
10.2.2 Nonregular Chains 218

10.3 Hidden Markov Models 218
10.3.1 Formal Definitions 219
10.3.2 Three Principal Problems 221
10.3.3 The Probability of an Observation Sequence 222
10.3.4 Most Probable States 224
10.3.5 Improving the Model 225
Note 226
Exercises 226

11 Reinforcement Learning 229
11.1 Introduction 229
11.2 Markov Decision Process 231
11.3 The Core Idea: Policy Improvement 233
11.4 Q-Learning 235
11.5 Temporal-Difference Learning 236
11.6 Learning with a Teacher 241
11.7 Partially Observable MDPs 243
11.7.1 Avoiding Bad States 244
11.7.2 Learning State Information from Temporal Sequences 247
11.7.3 Distinguishing the Value of States 249
11.8 Summary 252
Notes 253
Exercises 254

IV Systems **255**

12 Genetic Algorithms 263
12.1 Introduction 263
12.1.1 Genetic Operators 265
12.1.2 An Example 266
12.2 Schemata 267
12.2.1 Schemata Theorem 267
12.2.2 The Bandit Problem 268
12.3 Determining Fitness 270
12.3.1 Racing for Fitness 270
12.3.2 Coevolution of Parasites 271
Notes 273
Exercises 274

13 Genetic Programming 277
13.1 Introduction 277
13.2 Genetic Operators for Programs 278
13.3 Genetic Programming 280
13.4 Analysis 286
13.5 Modules 287
13.5.1 Testing for a Module Function 289
13.5.2 When to Diversify 290
13.6 Summary 293
Notes 293
Exercises 293

14 Summary 295
14.1 Learning to React: Memories 295
14.2 Learning During a Lifetime: Programs 296
14.3 Learning Across Generations: Systems 296
14.4 The Grand Challenge Revisited 297
Note 297

Index 299

Figures

1.1	The basic function units of the human brain	3
1.2	The brain's main memory system	5
1.3	The basic features of a neuron showing how a neuron communicates	6
1.4	A Turing machine	7
1.5	A hypothetical experiment that will not work	8
1.6	Easy and difficult traveling salesman problems	10
1.7	How a neuron can represent a code for images	13
1.8	Spatial scaling in hierarchical systems	16
1.9	Temporal scaling in hierarchical systems	17
2.1	The binary tree maze (constructed from matches)	30
2.2	Probability of an event using the Venn diagram and Bayes' rule	32
2.3	A setting for the use of Bayes' rule. A robot picks up boxes aided by data from a camera	32
2.4	The probability of real-valued random variables is defined by intervals denoting area under the density function	35
2.5	Stages in applying the Huffman algorithm	42
2.6	The final tree generated by applying the Huffman algorithm	43
2.7	An irreversible coding strategy	44
2.8	The situation when coding prototypes in terms of very long vectors results in having to classify an input vector according to the prototype that is nearest	45
2.9	An ingenious encoding of the human iris illustrates the virtues of redundant features in decision making	45
2.10	The MDL principle applied to image coding	48
2.11	The fundamental ideas of probability theory are captured by the Venn diagram, which uses a square denoted U to symbolize the universe of all possible events	49
2.12	A single roll of two dice	50

2.13	Different coding strategies for tactile signaling	52
3.1	Two examples of state spaces	56
3.2	Calculating the position of a rat based on direct recordings of neural firing	57
3.3	Two examples of operators	57
3.4	Partial specification of the state space for the cannibals and missionaries problem	59
3.5	Steps in the solution of the cannibals and missionaries problem	59
3.6	An eight-puzzle problem	60
3.7	Thinking of moving the blank provides a description of the possible operators: LEFT, RIGHT, UP, and DOWN	61
3.8	Enumerating the possibilities in a search results in a tree	61
3.9	The example problem solved by heuristic search	63
3.10	A single ply of a hypothetical search tree for a two-person game to illustrate the enumeration of moves	64
3.11	The single ply of the hypothetical search tree for a two-person game to illustrate the minimax principle	64
3.12	A half ply of the search tree for Othello	66
3.13	Topographic map	69
3.14	Minimax tree	69
3.15	Three-person game tree	70
4.1	The two main classes of techniques for compressing numerical data	72
4.2	The columns of A are linearly independent in three dimensions if the three column vectors are not coplanar	74
4.3	The basic matrix multiplication used in a linear coordinate transformation can be implemented in a linear neural network	75
4.4	Choosing coordinates to maximize the variance can simplify decision making	81
4.5	Using eigenvector transformations for encoding	83
4.6	A database of 12 figures used to calculate eigenvectors	85
4.7	The average face (I_{ave})	86
4.8	The first seven eigenvectors calculated	86
4.9	An example of a vector	89
5.1	Dynamics used to generate an exemplar of the letter A	96
5.2	The dynamic function has a straightforward interpretation as the instantaneous direction of motion of the system in the state space	97

5.3	The classical second-order system: an undamped harmonic oscillator consisting of a mass connected to a spring	99
5.4	A nonlinear system may be understood by characterizing the behavior of trajectories of the system linearized near equilibrium points, as in this second-order system	105
5.5	Different kinds of stability for state space trajectories	107
5.6	The idea behind Lyapunov stability. The state space trajectory always crosses level contours of V	108
6.1	A simple case for optimization: a function of two variables has a single minimum at the point x^*	115
6.2	A simple case for optimization: a function of one variable minimizes error	117
6.3	Interpretation of the constraint obtained by using Lagrange multipliers	120
6.4	The maximum condition for $u(t)$ can be developed by studying a small perturbation about a trajectory that is assumed optimal	122
6.5	A cart on a one-dimensional track	125
6.6	The basic computations in dynamic programming can be seen as a stage-wise process that starts at the terminal time and proceeds back to the initial time	129
6.7	The optimal solution found by dynamic programming: acceleration profile; velocity profile; distance profile	130
6.8	A cube with a cylindrical hole	131
II.1	The hierarchies composing visual cortex	137
II.2	The basic neural network model consists of neurons that represent a state	139
II.3	Different activation functions lend themselves to different kinds of models	139
II.4	A layered feedforward network contains no cycles	140
II.5	Content-addressable memory fills in missing details	141
7.1	The fundamental abstraction of a neuron represents the state of each neuron i as a firing rate x_i and the synapse connecting neurons i and j as a real numbered weight w_{ij}	144
7.2	A standard activation function for neural units	145
7.3	The Hopfield network can be visualized as a two-layered network where the output at one time becomes the input at the successive time	146
7.4	The discrete states of a CAM can be represented as the vertices of a hypercube	147
7.5	The correlation between patterns is binomially distributed	149

7.6 The database of telephone book entries for the content-addressable memory 150
7.7 Results of using the CAM. The basic feature of content-addressable memory is illustrated 151
7.8 Results of starting from a point in state space that is not near any of the memories 151
7.9 An abstraction of the Kanerva memory scheme illustrates the basic idea. Memory locations are sparsely distributed within the address space 152
7.10 Conventional computer memory uses a dense address space 153
7.11 Kanerva memory uses a sparse address space of locations that have considerably more bits than conventional memory 154
7.12 Kanerva memory can be implemented in a three-layered network 155
7.13 The black and white dots each represent a component of the memory and are encoded as ± 1 157
7.14 Using the heteroassociative feature for the storage of sequences 158
8.1 A basin of attraction for a given memory state vector. The motivating problem: points that are notionally the same prototype are not all included in the basin of attraction 164
8.2 The perceptron 165
8.3 The classification problem with linearly separable patterns 166
8.4 Adding a threshold allows the separation surface to go through the origin in the higher-dimensional space 167
8.5 The perceptron learning rule: If a pattern is misclassified, its projection onto the weight vector is proportional to the weight vector correction 167
8.6 The problem for multilayered networks is to adjust the internal weights 168
8.7 The nonlinear differentiable activation function used in multilayered networks 169
8.8 Two isomorphic family trees for an English and an Italian family that are used as a source of relationships for a feedforward network 172
8.9 A four-layered network used to learn family trees 173
8.10 The activation pattern in the network after it has been trained 173
8.11 The weights for two hidden units in the second layer of the network that are connected to the input person units 174

8.12 A very simple recurrent network with two units and four weights 175
8.13 The feedforward model for the network in Figure 8.12. Shown is just one time step from t to $t + 1$ 176
8.14 Unrolling the XOR network for five time steps 178
8.15 The results of learning XOR with recurrent networks 179
8.16 A particular situation to be handled by the perceptron learning rule 182
9.1 Data points; prototype points; Delaunay triangulation; Voronoi diagram 186
9.2 Two-layered network used to extract principal components 187
9.3 Principal components for natural images found by using Equation 9.3 189
9.4 In competitive learning, data points compete for prototypes. Here a cluster of points attracts a prototype by producing a common correction for the prototype's weight vector 190
9.5 The Kohonen algorithm applied to the traveling salesman problem 192
9.6 Performance of the topology discovery algorithm on an artificial input space with three-, two-, and one-dimensional correlation neighborhoods 193
9.7 Unsupervised versus supervised learning. (*a*) Unsupervised competitive learning performs well when the desired classes are obvious from the data clusters, but makes mistakes otherwise. (*b*) The mistakes are corrected with supervision 194
9.8 The supervised competitive learning algorithm works by moving prototypes to improve classification performance 196
9.9 Multimodal architecture 197
9.10 A simple case of multimodal pattern discrimination 198
9.11 How the minimum-disagreement rule works 200
9.12 Examples of the utterances /ba/ and /va/ 200
9.13 Results on the single-speaker cross-modal data set 201
9.14 The architecture behind the independent components 202
III.1 Evidence of basal ganglia programming 208
III.2 The basal ganglia are centrally located to interconnect motor subsystems with the cortex 209
III.3 The location and shape of the hippocampus, shown by the darkened area, facilitate interconnecting cortical areas 209

III.4	Data that have very different rewards, such as low (gray) or high (white), can have similar sensorimotor descriptions. Adding reward bits changes the metric so that situations with similar reward are near each other	210
10.1	The use of Markov models in speech recognition	216
10.2	The urn model illustrates the basis of the hidden Markov model	220
10.3	A specific urn model with two states that is used for three time steps	221
10.4	The recursive definition of the αs	223
11.1	The cart and inverted pendulum problem	230
11.2	(*top*) A simple maze illustrates the mechanics of reinforcement learning. (*middle*) The transition function. (*bottom*) The optimal policy for the maze problem	232
11.3	The basic vocabulary for reinforcement learning algorithms	234
11.4	The backgammon board in its initial configuration	238
11.5	(*top*) Player 1, black, has just rolled (6, 3). (*bottom*) There are several options for playing this move, one of which is shown	239
11.6	The architecture of the network used to estimate reward for backgammon	240
11.7	The weights from two hidden units after training. Black squares represent negative values; white squares positive values	240
11.8	Learning with an external critic. P_{critic} is the probability that the critic will tell the agent the correct value of the action taken at each step	242
11.9	Learning by watching. In this paradigm the agent sees the correct actions taken, but has to learn their value	242
11.10	A small bias in a random walk has huge effects. Plotted is the expected time to get to the goal as a function of the distance to the goal	243
11.11	(*a*) A linear state space. (*b*) Its internal representation	244
11.12	A graphical display from the output of a program that has learned the "pick up the green block" task	245
11.13	A state in the blocks world and its internal representation	245
11.14	Perceptual aliasing in the blocks world example	246
11.15	Using *k*-nearest sequence memory for state identification	248
11.16	The *k*-nearest sequence algorithm can lead to significant speedup. Here it is compared to the Chrisman algorithm for resolving aliasing by adding perceptual bits	248

11.17 The tree data structure for indexing into temporal sequences 249
11.18 A hallway navigation task with limited sensors 251
11.19 A tree for navigating through the environment in the maze task 252
11.20 A network used by reinforcement learning 254
IV.1 A chromosome is a linear structure with genes at specific loci 257
IV.2 Scanning electron microscope image of a fly after gene transplant showing ectopic eyes under the wing and on the antennae 258
12.1 Genetic operations on a three-letter alphabet of {a, b, c} 265
12.2 An eight-element sorting network 272
12.3 The translation of genotype to phenotype used by Hillis demonstrated on a four-input sorting network 273
13.1 Functions in LISP can be interpreted as trees. Two examples are (* x (+ x y)) and (+ (* z y)(/ y x)) 278
13.2 The genetic programming crossover operator works by picking fracture points in each of two programs 279
13.3 The genetic programming inversion operator works by picking two fracture points in a single program 280
13.4 The genetic programming mutation operator works by picking a fracture point in a single program 280
13.5 The board for the Pac-Man game, together with an example trace of a program found by genetic programming 283
13.6 Program code that plays Pac-Man shows the ad hoc nature of the solution generated by GP 285
13.7 The dynamics of GP are illustrated by tracking the fitness of individuals as a function of generations 286
13.8 Call graph for the extended function set in the even-8 parity example showing the generation when each function was discovered 289
13.9 The dynamics of GP with subroutines are illustrated by tracking the fitness of individuals as a function of generations 291
13.10 Entropy used as a measure to evaluate the GP search process 292

Tables

1.1	The organization of the brain into a layered anatomical structure with increasing refinement of function culminating with the site of complex programs and memory in the cerebral hemispheres	3
1.2	Time to solve a problem on a computer that takes 10^{-7} seconds per unit input as a function of the size of the input and the complexity of the algorithm	9
1.3	Learning times for human brain processes	15
1.4	The organization of human computation into temporal bands	18
1.5	The correspondence between learning algorithms and learning systems in the brain	18
2.1	Data showing the communication times for paths in mazes	30
2.2	The integral under the Gaussian probability density function	36
2.3	Binary encoding for a database of nine names	39
2.4	Huffman encoding for the example	43
2.5	Code used by the Willy Wonka Chocolate Factory	53
II.1	The function of different cortical areas	138
8.1	The patterns that represent OR	166
8.2	Table for XOR	177
9.1	Parameters used in topology-finding algorithm	194
11.1	Categories of Markov models	231
11.2	The transition function for the maze problem	232
12.1	Initial condition for the genetic algorithm example	266
12.2	Mating process	266
12.3	The genetic algorithm example after one step	267
13.1	The even-3 parity function	282
13.2	The GP Pac-Man nonterminals (control functions)	284
13.3	The GP Pac-Man terminals	284

13.4 Comparison of GP and modular GP for parity problems of different sizes 289
13.5 Important steps in the evolutionary trace for a run of even-8 parity 290
13.6 Comparison between solutions obtained with standard GP, subroutines, and a carefully hand-designed program 292