```
Archive-name: ai-faq/neural-nets/part7
Last-modified: 2002-04-09
URL: ftp://ftp.sas.com/pub/neural/FAQ7.html
Maintainer: saswss@unx.sas.com (Warren S. Sarle)
```

This is part 7 (of 7) of a monthly posting to the Usenet newsgroup comp.ai.neural-nets. See the part 1 of this posting for full information what it is all about.

========== Questions ==========

Subject: Neural Network hardware?

Overview articles:

- Ienne, Paolo and Kuhn, Gary (1995), "Digital Systems for Neural Networks", in Papamichalis, P. and Kerwin, R., eds., *Digital Signal Processing Technology,* Critical Reviews Series CR57 Orlando, FL: SPIE Optical Engineering, pp 314-45, ftp://mantraftp.epfl.ch/mantra/ienne.spie95.A4.ps.gz or ftp://mantraftp.epfl.ch/mantra/ienne.spie95.US.ps.gz
- ftp://ftp.mrc-apu.cam.ac.uk/pub/nn/murre/neurhard.ps (1995)
- ftp://ftp.urc.tue.nl/pub/neural/hardware_general.ps.gz (1993)

Various NN HW information can be found in the Web site http://www1.cern.ch/NeuralNets/nnwInHepHard.html (from people who really use such stuff!). Several applications are described in http://www1.cern.ch/NeuralNets/nnwInHepExpt.html

More information on NN chips can be obtained from the Electronic Engineers Toolbox web page. Go to http://www.eg3.com/ebox.htm, type "neural" in the quick search box, click on "chip co's" and then on "search".

Further WWW pointers to NN Hardware:

- http://msia02.msi.se/~lindsey/nnwAtm.html
- http://www.genobyte.com/article.html

Here is a short list of companies:

1. **HNC, INC.**

```
HNC Inc.
5930 Cornerstone Court West
San Diego, CA 92121-3728

619-546-8877  Phone
619-452-6524  Fax
```

   HNC markets:
   - Database Mining Workstation (DMW), a PC based system that builds models of relationships and patterns in data.
   - The SIMD Numerical Array Processor (SNAP). It is an attached parallel array processor in a VME chassis with between 16 and 64 parallel floating point processors. It provides between 640 MFLOPS and 2.56 GFLOPS for neural network and signal processing applications. A Sun SPARCstation serves as the host. The SNAP won the IEEE 1993 Gordon Bell Prize for best price/performance for supercomputer class systems.

2. **SAIC (Sience Application International Corporation)**

```
10260 Campus Point Drive
MS 71, San Diego
CA 92121
(619) 546 6148
Fax: (619) 546 6736
```

3. **Micro Devices**

```
30 Skyline Drive
```

```
Lake Mary
FL 32746-6201
(407) 333-4379
```

MicroDevices makes MD1220 - 'Neural Bit Slice'. Each of the products mentioned sofar have very different usages. Although this sounds similar to Intel's product, the architectures are not.

4. **Intel Corp**

```
2250 Mission College Blvd
Santa Clara, Ca 95052-8125
Attn ETANN, Mail Stop SC9-40
(408) 765-9235
```

Intel was making an experimental chip (which is no longer produced): 80170NW - Electrically trainable Analog Neural Network (ETANN) It has 64 'neurons' on it - almost fully internally connectted and the chip can be put in an hierarchial architecture to do 2 Billion interconnects per second. Support software by

```
  California Scientific Software
  10141 Evening Star Dr #6
  Grass Valley, CA 95945-9051
  (916) 477-7481
```

Their product is called 'BrainMaker'.

5. **Tubb Research Limited**

```
7a Lavant Street
Peterfield
Hampshire
GU32 2EL
United Kingdom
Tel: +44 730 60256
```

6. **Adaptive Solutions Inc**

```
1400 NW Compton Drive
Suite 340
Beaverton, OR 97006
U. S. A.
Tel: 503-690-1236;   FAX: 503-690-1249
```

7. **NeuroDynamX, Inc.**

```
P.O. Box 14
Marion, OH  43301-0014
Voice (740) 387-5074
Fax: (740) 382-4533
Internet:  jwrogers@on-ramp.net
http://www.neurodynamx.com
```

InfoTech Software Engineering purchased the software and trademarks from NeuroDynamX, Inc. and, using the NeuroDynamX tradename, continues to publish the DynaMind, DynaMind Developer Pro and iDynaMind software packages.

8. **NeuroClassifier**

URL: http://www.ice.el.utwente.nl/Finished/Neuro/

Email: peter.masa@csemne.ch

9. **NeuriCam, S.r.l.**

```
Via S. Maria Maddalena, 38100 Trento, Italy
Tel: +39 0461 260 552
Fax: +39 0461 260 617
Email: info@neuricam.com
```

NC3001 TOTEM - Digital Processor for Neural Networks
TOTEM is a digital VLSI parallel processor for fast learning and recognition with artificial neural networks. Its high processing power, low power dissipation and limited chip size make it ideally suited for embedded applications. The architecture is optimised for the implementation of the Reactive Tabu Search learning algorithm, a competitive alternative to back-propagation which leads to a very compact VLSI implementation.

And here is an incomplete overview of known Neural Computers with their newest known reference.

- Special Computers

  1. AAP-2 Takumi Watanabe, Yoshi Sugiyama, Toshio Kondo, and Yoshihiro Kitamura.
     "Neural network simulation on a massively parallel cellular array processor: AAP-2." In International Joint Conference on Neural Networks, 1989.
  2. ANNA B.E.Boser, E.Sackinger, J.Bromley, Y.leChun, and L.D.Jackel.
     "Hardware Requirements for Neural Network Pattern Classifiers." In *IEEE Micro*, 12(1), pages 32-40, February 1992.
  3. Analog Neural Computer Paul Mueller et al.
     "Design and performance of a prototype analog neural computer." In Neurocomputing, 4(6):311-323, 1992.
  4. APx -- Array Processor Accelerator F.Pazienti.
     "Neural networks simulation with array processors." In *Advanced Computer Technology, Reliable Systems and Applications; Proceedings of the 5th Annual Computer Conference*, pages 547-551. IEEE Comput. Soc. Press, May 1991. ISBN: 0-8186-2141-9.
  5. ASP -- Associative String Processor A.Krikelis.
     "A novel massively associative processing architecture for the implementation artificial neural networks." In *1991 International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1057-1060. IEEE Comput. Soc. Press, May 1991.
  6. BSP400 Jan N.H. Heemskerk, Jacob M.J. Murre, Jaap Hoekstra, Leon H.J.G. Kemna, and Patrick T.W. Hudson.
     "The bsp400: A modular neurocomputer assembled from 400 low-cost microprocessors." In International Conference on Artificial Neural Networks. Elsevier Science, 1991.
  7. BLAST J.G.Elias, M.D.Fisher, and C.M.Monemi.
     "A multiprocessor machine for large-scale neural network simulation." In *IJCNN91-Seattle: International Joint Conference on Neural Networks*, volume 1, pages 469-474. IEEE Comput. Soc. Press, July 1991. ISBN: 0-7883-0164-1.
  8. CNAPS Neurocomputer H.McCartor
     "Back Propagation Implementation on the Adaptive Solutions CNAPS Neurocomputer." In *Advances in Neural Information Processing Systems*, 3, 1991.
  9. GENES~IV and MANTRA~I Paolo Ienne and Marc A. Viredaz
     "GENES~IV: A Bit-Serial Processing Element for a Multi-Model Neural-Network Accelerator." Journal of VLSI Signal Processing, volume 9, no. 3, pages 257--273, 1995.
  10. MA16 -- Neural Signal Processor U.Ramacher, J.Beichter, and N.Bruls.
      "Architecture of a general-purpose neural signal processor." In *IJCNN91-Seattle:*

International Joint Conference on Neural Networks, volume 1, pages 443-446. IEEE Comput. Soc. Press, July 1991. ISBN: 0-7083-0164-1.

11. Mindshape Jan N.H. Heemskerk, Jacob M.J. Murre Arend Melissant, Mirko Pelgrom, and Patrick T.W. Hudson.
"Mindshape: a neurocomputer concept based on a fractal architecture." In International Conference on Artificial Neural Networks. Elsevier Science, 1992.
12. mod 2 Michael L. Mumford, David K. Andes, and Lynn R. Kern.
"The mod 2 neurocomputer system design." In IEEE Transactions on Neural Networks, 3(3):423-433, 1992.
13. NERV R.Hauser, H.Horner, R. Maenner, and M.Makhaniok.
"Architectural Considerations for NERV - a General Purpose Neural Network Simulation System." In *Workshop on Parallel Processing: Logic, Organization and Technology -- WOPPLOT 89*, pages 183-195. Springer Verlag, Mars 1989. ISBN: 3-5405-5027-5.
14. NP -- Neural Processor D.A.Orrey, D.J.Myers, and J.M.Vincent.
"A high performance digital processor for implementing large artificial neural networks." In *Proceedings of of the IEEE 1991 Custom Integrated Circuits Conference*, pages 16.3/1-4. IEEE Comput. Soc. Press, May 1991. ISBN: 0-7883-0015-7.
15. RAP -- Ring Array Processor N.Morgan, J.Beck, P.Kohn, J.Bilmes, E.Allman, and J.Beer.
"The ring array processor: A multiprocessing peripheral for connectionist applications." In *Journal of Parallel and Distributed Computing*, pages 248-259, April 1992.
16. RENNS -- REconfigurable Neural Networks Server O.Landsverk, J.Greipsland, J.A.Mathisen, J.G.Solheim, and L.Utne.
"RENNS - a Reconfigurable Computer System for Simulating Artificial Neural Network Algorithms." In *Parallel and Distributed Computing Systems, Proceedings of the ISMM 5th International Conference*, pages 251-256. The International Society for Mini and Microcomputers - ISMM, October 1992. ISBN: 1-8808-4302-1.
17. SMART -- Sparse Matrix Adaptive and Recursive Transforms P.Bessiere, A.Chams, A.Guerin, J.Herault, C.Jutten, and J.C.Lawson.
"From Hardware to Software: Designing a `Neurostation'." In *VLSI design of Neural Networks*, pages 311-335, June 1990.
18. SNAP -- Scalable Neurocomputer Array Processor E.Wojciechowski.
"SNAP: A parallel processor for implementing real time neural networks." In *Proceedings of the IEEE 1991 National Aerospace and Electronics Conference; NAECON-91*, volume 2, pages 736-742. IEEE Comput.Soc.Press, May 1991.
19. Toroidal Neural Network Processor S.Jones, K.Sammut, C.Nielsen, and J.Staunstrup.
"Toroidal Neural Network: Architecture and Processor Granularity Issues." In *VLSI design of Neural Networks*, pages 229-254, June 1990.
20. SMART and SuperNode P. Bessi`ere, A. Chams, and P. Chol.
"MENTAL : A virtual machine approach to artificial neural networks programming." In NERVES, ESPRIT B.R.A. project no 3049, 1991.

- Standard Computers

1. EMMA-2 R.Battiti, L.M.Briano, R.Cecinati, A.M.Colla, and P.Guido.
"An application oriented development environment for Neural Net models on multiprocessor Emma-2." In *Silicon Architectures for Neural Nets; Proceedings for the IFIP WG.10.5 Workshop*, pages 31-43. North Holland, November 1991. ISBN: 0-4448-9113-7.
2. iPSC/860 Hypercube D.Jackson, and D.Hammerstrom
"Distributing Back Propagation Networks Over the Intel iPSC/860 Hypercube" In *IJCNN91-Seattle: International Joint Conference on Neural Networks*, volume 1, pages 569-574. IEEE Comput. Soc. Press, July 1991. ISBN: 0-7083-0164-1.

3. SCAP -- Systolic/Cellular Array Processor Wei-Ling L., V.K.Prasanna, and K.W.Przytula. "Algorithmic Mapping of Neural Network Models onto Parallel SIMD Machines." In *IEEE Transactions on Computers*, 40(12), pages 1390-1401, December 1991. ISSN: 0018-9340.

Subject: What are some applications of NNs?

There are vast numbers of published neural network applications. If you don't find something from your field of interest below, try a web search. Here are some useful search engines:
http://www.google.com/
http://search.yahoo.com/
http://www.altavista.com/
http://www.deja.com/

General

- The Pacific Northwest National Laboratory: http://www.emsl.pnl.gov:2080/proj/neuron/neural/ including a list of commercial applications at http://www.emsl.pnl.gov:2080/proj/neuron/neural/products/
- The Stimulation Initiative for European Neural Applications: http://www.mbfys.kun.nl/snn/siena/cases/
- The DTI NeuroComputing Web's Applications Portfolio: http://www.globalweb.co.uk/nctt/portfolo/
- The Applications Corner, NeuroDimension, Inc.: http://www.nd.com/appcornr/purpose.htm
- The BioComp Systems, Inc. Solutions page: http://www.bio-comp.com
- Chen, C.H., ed. (1996) *Fuzzy Logic and Neural Network Handbook,* NY: McGraw-Hill, ISBN 0-07-011189-8.
- The series *Advances in Neural Information Processing Systems* containing proceedings of the conference of the same name, published yearly by Morgan Kauffman starting in 1989 and by The MIT Press in 1995.

Agriculture

- P.H. Heinemann, Automated Grading of Produce: http://server.age.psu.edu/dept/fac/Heinemann/phhdocs/visionres.html
- Deck, S., C.T. Morrow, P.H. Heinemann, and H.J. Sommer, III. 1995. Comparison of a neural network and traditional classifier for machine vision inspection. Applied Engineering in Agriculture. 11(2):319-326.
- Tao, Y., P.H. Heinemann, Z. Varghese, C.T. Morrow, and H.J. Sommer III. 1995. Machine vision for color inspection of potatoes and apples. Transactions of the American Society of Agricultural Engineers. 38(5):1555-1561.

Automotive

- "No Hands Across America Journal" - steering a car: http://cart.frc.ri.cmu.edu/users/hpm/project.archive/reference.file/Journal.html
  Photos: http://www.techfak.uni-bielefeld.de/ags/ti/personen/zhang/seminar/intelligente-autos/tour.html

Chemistry

- PNNL, General Applications of Neural Networks in Chemistry and Chemical Engineering: http://www.emsl.pnl.gov:2080/proj/neuron/neural/bib/chemistry.html.
- Prof. Dr. Johann Gasteiger, Neural Networks and Genetic Algorithms in Chemistry: http://www2.ccc.uni-erlangen.de/publications/publ_topics/publ_topics-12.html
- Roy Goodacre, pyrolysis mass spectrometry: http://gepasi.dbs.aber.ac.uk/roy/pymshome.htm and Fourier transform infrared (FT-IR) spectroscopy: http://gepasi.dbs.aber.ac.uk/roy/ftir/ftirhome.htm contain applications of a variety of NNs as well as PLS (partial least squares) and other statistical methods.
- Situs, a program package for the docking of protein crystal structures to single-molecule, low-resolution maps from electron microscopy or small angle X-ray scattering: http://chemcca10.ucsd.edu/~situs/
- An on-line application of a Kohonen network with a 2-dimensional output layer for prediction of protein secondary structure percentages from UV circular dichroism spectra: http://www.embl-heidelberg.de/~andrade/k2d/.

Criminology

- Computer Aided Tracking and Characterization of Homicides and Sexual Assaults (CATCH): http://lancair.emsl.pnl.gov:2080/proj/neuron/papers/kangas.spie99.abs.html

Face recognition

- Face Recognition Home Page: http://www.cs.rug.nl/~peterkr/FACE/face.html
- Konen, W., "Neural information processing in real-world face-recognition applications," http://www.computer.muni.cz/pubs/expert/1996/trends/x4004/konen.htm
- Jiang, Q., "Principal Component Analysis and Neural Network Based Face Recognition," http://people.cs.uchicago.edu/~qingj/ThesisHtml/
- Lawrence, S., Giles, C.L., Tsoi, A.C., Back, A.D. (1997), "Face Recognition: A Convolutional Neural Network Approach," IEEE Transactions on Neural Networks, 8, 98-113, http://www.neci.nec.com/~lawrence/papers/face-tnn97/latex.html

Finance and economics

- Athanasios Episcopos, References on Neural Net Applications to Finance and Economics: http://www.compulink.gr/users/episcopo/neurofin.html
- Franco Busetti, Heuristics and artificial intelligence in finance and investment: http://www.geocities.com/francorbusetti/
- Trippi, R.R. & Turban, E. (1993), *Neural Networks in Finance and Investing,* Chicago: Probus.
- Zirilli, J.S. (1996), *Financial Prediction Using Neural Networks,* International Thomson Publishing, ISBN 1850322341, http://www6.bcity.com/mjfutures/
- Andreas S. Weigend, Yaser Abu-Mostafa, A. Paul N. Refenes (eds.) (1997) *Decision Technologies for Financial Engineering: Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets* (Nncm '96) Publisher: World Scientific Publishing Company, ISBN: 9810231245

Games, sports, gambling

- General:

  Jay Scott, Machine Learning in Games: http://satirist.org/learn-game/index.html

METAGAME Game-Playing Workbench: ftp://ftp.cl.cam.ac.uk/users/bdp/METAGAME

R.S. Sutton, "Learning to predict by the methods of temporal differences", Machine Learning 3, p. 9-44 (1988).

David E. Moriarty and Risto Miikkulainen (1994). "Evolving Neural Networks to Focus Minimax Search," In Proceedings of Twelfth National Conference on Artificial Intelligence (AAAI-94, Seattle, WA), 1371-1377. Cambridge, MA: MIT Press, http://www.cs.utexas.edu/users/nn/pages/publications/neuro-evolution.html

Games World '99 at http://gamesworld99.free.fr/menuframe.htm

- Backgammon:

  G. Tesauro and T.J. Sejnowski (1989), "A Parallel Network that learns to play Backgammon," Artificial Intelligence, vol 39, pp. 357-390.

  G. Tesauro and T.J. Sejnowski (1990), "Neurogammon: A Neural Network Backgammon Program," IJCNN Proceedings, vol 3, pp. 33-39, 1990.

  G. Tesauro (1995), "Temporal Difference Learning and TD-Gammon," Communications of the ACM, 38, 58-68, http://www.research.ibm.com/massive/tdl.html

  Pollack, J.P. and Blair, A.D. (1997), "Co-Evolution in the Successful Learning of Backgammon Strategy," Brandeis University Computer Science Technical Report CS-97-193, http://www.demo.cs.brandeis.edu/papers/long.html#hcgam97

- Bridge:

  METAGAME: ftp://ftp.cl.cam.ac.uk/users/bdp/bridge.ps.Z

  He Yo, Zhen Xianjun, Ye Yizheng, Li Zhongrong (19??), "Knowledge acquisition and reasoning based on neural networks - the research of a bridge bidding system," INNC '90, Paris, vol 1, pp. 416-423.

  M. Kohle and F. Schonbauer (19??), "Experience gained with a neural network that learns to play bridge," Proc. of the 5th Austrian Artificial Intelligence meeting, pp. 224-229.

- Checkers/Draughts:

  Mark Lynch (1997), "NeuroDraughts: an application of temporal difference learning to draughts," http://www.ai.univie.ac.at/~juffi/lig/Papers/lynch-thesis.ps.gz Software available at http://satirist.org/learn-game/archive/NeuroDraughts-1.00.zip

  K. Chellapilla and D. B. Fogel, "Co-Evolving Checkers Playing Programs using Only Win, Lose, or Draw," SPIE's AeroSense'99: Applications and Science of Computational Intelligence II, Apr. 5-9, 1999, Orlando, Florida, USA, http://vision.ucsd.edu/~kchellap/Publications.html

  David Fogel (1999), *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence* (2nd edition), IEEE, ISBN: 078035379X

  David Fogel (2001), *Blondie24: Playing at the Edge of AI,* Morgan Kaufmann Publishers, ISBN: 1558607838
  According to the publisher, this is:

> ... the first book to bring together the most advanced work in the general use of evolutionary computation for creative results. It is well suited for the general computer science audience.
>
> Here's the story of a computer that taught itself to play checkers far better than its creators ever could. Blondie24 uses a program that emulates the basic principles of Darwin evolution to discover on its own how to excel at the game. Through this entertaining story, the book provides the reader some of the history of AI and explores its future.
>
> Unlike Deep Blue, the celebrated chess machine that beat Garry Kasparov, the former world champion chess player, this evolutionary program didn't have access to other games played by human grand masters, or databases of moves for the endgame. It created its own means for evaluating the patterns of pieces that it experienced by evolving artificial neural networks--mathematical models that loosely describe how a brain works.

See http://www.natural-selection.com/NSIPublicationsOnline.htm for a variety of online papers by Fogel.

Not NNs, but classic papers:

A.L. Samuel (1959), "Some studies in machine learning using the game of checkers," IBM journal of Research and Development, vol 3, nr. 3, pp. 210-229.

A.L. Samuel (1967), "Some studies in machine learning using the game of checkers 2 - recent progress," IBM journal of Research and Development, vol 11, nr. 6, pp. 601-616.

- Chess:

  Sebastian Thrun, NeuroChess: http://satirist.org/learn-game/systems/neurochess.html

  Luke Pellen, Octavius: http://home.seol.net.au/luke/octavius/

  Louis Savain (AKA Nemesis), Animal, a spiking neural network that the author hopes will learn to play a passable game of chess after he implements the motivation mechanism: http://home1.gte.net/res02khr/AI/Temporal_Intelligence.htm

- Dog racing:

  H. Chen1, P. Buntin, L. She, S. Sutjahjo, C. Sommer, D. Neely (1994), "Expert Prediction, Symbolic Learning, and Neural Networks: An Experiment on Greyhound Racing," IEEE Expert, December 1994, 21-27, http://ai.bpa.arizona.edu/papers/dog93/dog93.html

- Football (Soccer):

  Kuonen Diego, "Statistical Models for Knock-out Soccer Tournaments", http://dmawww.epfl.ch/~kuonen/CALCIO/ (not neural nets, but relevant)

- Go:

  David Stoutamire (19??), "Machine Learning, Game Play, and Go," Center for Automation and Intelligent Systems Research TR 91-128, Case Western Reserve University. http://www.stoutamire.com/david/publications.html

David Stoutamire (1991), *Machine Learning Applied to Go,* M.S. thesis, Case Western Reserve University, ftp://ftp.cl.cam.ac.uk/users/bdp/go.ps.Z

Schraudolph, N., Dayan, P., Sejnowski, T. (1994), "Temporal Difference Learning of Position Evaluation in the Game of Go," In: Neural Information Processing Systems 6, Morgan Kaufmann 1994, ftp://bsdserver.ucsf.edu/Go/comp/td-go.ps.Z

P. Donnelly, P. Corr & D. Crookes (1994), "Evolving Go Playing Strategy in Neural Networks", AISB Workshop on Evolutionary Computing, Leeds, England, ftp://www.joy.ne.jp/welcome/igs/Go/computer/egpsnn.ps.Z or ftp://ftp.cs.cuhk.hk/pub/neuro/GO/techreports/egpsnn.ps.Z

Markus Enzenberger (1996), "The Integration of A Priori Knowledge into a Go Playing Neural Network," http://www.cgl.ucsf.edu/go/Programs/neurogo-html/neurogo.html

Norman Richards, David Moriarty, and Risto Miikkulainen (1998), "Evolving Neural Networks to Play Go," Applied Intelligence, 8, 85-96, http://www.cs.utexas.edu/users/nn/pages/publications/neuro-evolution.html

Dahl, F. A. (1999), "Honte, a Go-playing program using neural nets", http://www.ai.univie.ac.at/icml-99-ws-games/papers/dahl.ps.gz

- Go-Moku:

  Freisleben, B., "Teaching a Neural Network to Play GO-MOKU," in I. Aleksander and J. Taylor, eds, Artificial Neural Networks 2, Proc. of ICANN-92, Brighton UK, vol. 2, pp. 1659-1662, Elsevier Science Publishers, 1992

  Katz, W.T. and Pham, S.P. "Experience-Based Learning Experiments using Go-moku", Proc. of the 1991 IEEE International Conference on Systems, Man, and Cybernetics, 2: 1405-1410, October 1991.

- Olympics:

  E.M.Condon, B.L.Golden, E.A.Wasil (1999), "Predicting the success of nations at the Summer Olympics using neural networks", Computers & Operations Research, 26, 1243-1265.

- Pong:

  http:// www.engin.umd.umich.edu/~watta/MM/pong/pong5.html

- Reversi/Othello:

  David E. Moriarty and Risto Miikkulainen (1995). Discovering Complex Othello Strategies through Evolutionary Neural Networks. Connection Science, 7, 195-209, http://www.cs.utexas.edu/users/nn/pages/publications/neuro-evolution.html

  Yoshioka, T., Ishii, S., and Ito, M., Strategy acquisition for the game ``Othello'' based on reinforcement learning, IEICE Transactions on Information and Systems E82-D 12, 1618-1626, 1999, http://mimi.aist-nara.ac.jp/~taku-y/

- Tic-Tac-Toe/Noughts and Crosses:

  Fogel, David Bb (1993), "Using evolutionary programming to construct neural networks that are capable of playing tic-tac-toe," Intern. Conf. on Neural Networks 1993, IEEE, San Francisco, CA, pp. 875-880.

Richard S. Sutton and Andrew G. Barto (1998), *Reinforcement Learning: An Introduction* The MIT Press, ISBN: 0262193981, http://www-anw.cs.umass.edu/~rich/book/the-book.html

Yongzheng Zhang, Chen Teng, Sitan Wei (2000), "Game playing with Evolutionary Strategies and Modular Neural Networks: Tic-Tac-Toe," http://www.cs.dal.ca/~mheywood/GAPproject/EvolvingGamePlay.html

Rob Ellison, "Neural Os and Xs," http://www.catfood.demon.co.uk/beta/game.html (An online Javascript demo, but you may not live long enough to teach the network to play a mediocre game. I'm not sure what kind of network it uses, but maybe you can figure that out if you read the source.)

Industry

- PNNL, Neural Network Applications in Manufacturing: http://www.emsl.pnl.gov:2080/proj/neuron/neural/bib/manufacturing.html.
- PNNL, Applications in the Electric Power Industry: http://www.emsl.pnl.gov:2080/proj/neuron/neural/bib/power.html.
- PNNL, Process Control: http://www.emsl.pnl.gov:2080/proj/neuron/neural/bib/process.html.
- Raoul Tawel, Ken Marko, and Lee Feldkamp (1998), "Custom VLSI ASIC for Automotive Applications with Recurrent Networks", http://www.jpl.nasa.gov/releases/98/ijcnn98.pdf
- Otsuka, Y. et al. "Neural Networks and Pattern Recognition of Blast Furnace Operation Data" Kobelco Technology Review, Oct. 1992, 12
- Otsuka, Y. et al. "Applications of Neural Network to Iron and Steel Making Processes" 2. International Conference on Fuzzy Logic and Neural Networks, Iizuka, 1992
- Staib, W.E. "Neural Network Control System for Electric Arc Furnaces" M.P.T. International, 2/1995, 58-61
- Portmann, N. et al. "Application of Neural Networks in Rolling Automation" Iron and Steel Engineer, Feb. 1995, 33-36
- Gorni, A.A. (2000), "The modelling of hot rolling processes using neural networks: A bibliographical review", http://www.geocities.com/SiliconValley/5978/neural_1998.html
- Murat, M. E., and Rudman, A. J., 1992, Automated first arrival picking: A neural network approach: Geophysical Prospecting, 40, 587-604.

Materials science

- Phase Transformations Research Group (search for "neural"): http://www.msm.cam.ac.uk/phase-trans/pubs/ptpuball.html

Medicine

- PNNL, Applications in Medicine and Health: http://www.emsl.pnl.gov:2080/proj/neuron/neural/bib/medicine.html.

Music

- Mozer, M. C. (1994), "Neural network music composition by prediction: Exploring the benefits of psychophysical constraints and multiscale processing," Connection Science, 6, 247-280, http://www.cs.colorado.edu/~mozer/papers/music.html.
- Griffith, N., and Todd, P.M., eds. (1999), *Musical Networks: Parallel Distributed Perception and*

Performance, Cambridge, MA: The MIT Press, ISBN 0-262-07181-9.

Robotics

- Institute of Robotics and System Dynamics: http://www.robotic.dlr.de/LEARNING/
- UC Berkeley Robotics and Intelligent Machines Lab: http://robotics.eecs.berkeley.edu/
- Perth Robotics and Automation Laboratory: http://telerobot.mech.uwa.edu.au/
- University of New Hampshire Robot Lab: http://www.ece.unh.edu/robots/rbt_home.htm

Weather forecasting and atmospheric science

- UBC Climate Prediction Group: http://www.ocgy.ubc.ca/projects/clim.pred/index.html
- Artificial Intelligence Research In Environmental Science: http://www.salinas.net/~jpeak/airies/airies.html
- MET-AI, an mailing list for meteorologists and AI researchers: http://www.comp.vuw.ac.nz/Research/met-ai
- Caren Marzban, Ph.D., Research Scientist, National Severe Storms Laboratory: http://www.nhn.ou.edu/~marzban/
- David Myers's references on NNs in atmospheric science: http://terra.msrc.sunysb.edu/~dmyers/ai_refs

Weird

Zaknich, Anthony and Baker, Sue K. (1998), "A real-time system for the characterisation of sheep feeding phases from acoustic signals of jaw sounds," Australian Journal of Intelligent Information Processing Systems (AJIIPS), Vol. 5, No. 2, Winter 1998.

Abstract
This paper describes a four-channel real-time system for the detection and measurement of sheep rumination and mastication time periods by the analysis of jaw sounds transmitted through the skull. The system is implemented using an 80486 personal computer, a proprietary data acquisition card (PC-126) and a custom made variable gain preamplifier and bandpass filter module. Chewing sounds are transduced and transmitted to the system using radio microphones attached to the top of the sheep heads. The system's main functions are to detect and estimate rumination and mastication time periods, to estimate the number of chews during the rumination and mastication periods, and to provide estimates of the number of boli in the rumination sequences and the number of chews per bolus. The individual chews are identified using a special energy threshold detector. The rumination and mastication time periods are determined by neural network classifier using a combination of time and frequency domain features extracted from successive 10 second acoustic signal blocks.

Subject: What to do with missing/incomplete data?

The problem of missing data is very complex.

For unsupervised learning, conventional statistical methods for missing data are often appropriate (Little and Rubin, 1987; Schafer, 1997). There is a concise introduction to these methods in the University of Texas statistics FAQ at http://www.utexas.edu/cc/faqs/stat/general/gen25.html.

For supervised learning, the considerations are somewhat different, as discussed by Sarle (1998). The statistical literature on missing data deals almost exclusively with training rather than prediction (e.g., Little, 1992). For example, if you have only a small proportion of cases with missing data, you can simply throw those cases out for purposes of training; if you want to make predictions for cases with missing inputs, you don't have the option of throwing those cases out! In theory, Bayesian methods take care of everything, but a full Bayesian analysis is practical only with special models (such as multivariate normal distributions) or small sample sizes. The neural net literature contains a few good papers that cover prediction with missing inputs (e.g., Ghahramani and Jordan, 1997; Tresp, Neuneier, and Ahmad 1995), but much research remains to be done.

References:

Donner, A. (1982), "The relative effectiveness of procedures commonly used in multiple regression analysis for dealing with missing values," American Statistician, 36, 378-381.

Ghahramani, Z. and Jordan, M.I. (1994), "Supervised learning from incomplete data via an EM approach," in Cowan, J.D., Tesauro, G., and Alspector, J. (eds.) *Advances in Neural Information Processing Systems 6,* San Mateo, CA: Morgan Kaufman, pp. 120-127.

Ghahramani, Z. and Jordan, M.I. (1997), "Mixture models for Learning from incomplete data," in Greiner, R., Petsche, T., and Hanson, S.J. (eds.) *Computational Learning Theory and Natural Learning Systems, Volume IV: Making Learning Systems Practical,* Cambridge, MA: The MIT Press, pp. 67-85.

Jones, M.P. (1996), "Indicator and stratification methods for missing explanatory variables in multiple linear regression," J. of the American Statistical Association, 91, 222-230.

Little, R.J.A. (1992), "Regression with missing X's: A review," J. of the American Statistical Association, 87, 1227-1237.

Little, R.J.A. and Rubin, D.B. (1987), *Statistical Analysis with Missing Data,* NY: Wiley.

McLachlan, G.J. (1992) *Discriminant Analysis and Statistical Pattern Recognition,* Wiley.

Sarle, W.S. (1998), "Prediction with Missing Inputs," in Wang, P.P. (ed.), JCIS '98 Proceedings, Vol II, Research Triangle Park, NC, 399-402, ftp://ftp.sas.com/pub/neural/JCIS98.ps.

Schafer, J.L. (1997), *Analysis of Incomplete Multivariate Data,* London: Chapman & Hall, ISBN 0 412 04061 1.

Tresp, V., Ahmad, S. and Neuneier, R., (1994), "Training neural networks with deficient data", in Cowan, J.D., Tesauro, G., and Alspector, J. (eds.) *Advances in Neural Information Processing Systems 6,* San Mateo, CA: Morgan Kaufman, pp. 128-135.

Tresp, V., Neuneier, R., and Ahmad, S. (1995), "Efficient methods for dealing with missing data in supervised learning", in Tesauro, G., Touretzky, D.S., and Leen, T.K. (eds.) *Advances in Neural Information Processing Systems 7,* Cambridge, MA: The MIT Press, pp. 689-696.

Subject: How to forecast time series (temporal sequences)?

In most of this FAQ, it is assumed that the training cases are statistically independent. That is, the training cases consist of pairs of input and target vectors, `(X_i,Y_i), i=1,...,N,` such that the conditional distribution of `Y_i` given all the other training data, (`X_j, j=1,...,N`, and `Y_j, j=1,...i-1,i+1,...N`) is equal to the conditional distribution of `Y_i` given `X_i` regardless of the values in the other training cases. Independence of cases is often achieved by random sampling.

The most common violation of the independence assumption occurs when cases are observed in a certain order relating to time or space. That is, case `(X_i,Y_i)` corresponds to time `T_i`, with `T_1 < T_2 < ... < T_N`. It is assumed that the current target `Y_i` may depend not only on `X_i` but also on `(X_i,Y_i)` in the recent past. If the `T_i` are equally spaced, the simplest way to deal with this dependence is to include additional inputs (called lagged variables, shift registers, or a tapped delay line) in the network. Thus, for target `Y_i`, the inputs may include `X_i, Y_{i-1}, X_{i-1}, Y_{i-1}, X_{i-2}`, etc. (In some situations, `X_i` would not be known at the time you are trying to forecast `Y_i` and would therefore be excluded from the inputs.) Then you can train an ordinary feedforward network with these targets and lagged variables. The use of lagged variables has been extensively studied in the statistical and econometric literature (Judge, Griffiths, Hill, Lütkepohl and Lee, 1985). A network in which the only inputs are lagged target values is called an "autoregressive model." The input space that includes all of the lagged variables is called the "embedding space."

If the `T_i` are *not* equally spaced, everything gets much more complicated. One approach is to use a smoothing technique to interpolate points at equally spaced intervals, and then use the interpolated values for training instead of the original data.

Use of lagged variables increases the number of decisions that must be made during training, since you must consider which lags to include in the network, as well as which input variables, how many hidden units, etc. Neural network researchers have therefore attempted to use partially recurrent networks instead of feedforward networks with lags (Weigend and Gershenfeld, 1994). Recurrent networks store information about past values in the network itself. There are many different kinds of recurrent architectures (Hertz, Krogh, and Palmer 1991; Mozer, 1994; Horne and Giles, 1995; Kremer, 199?). For example, in time-delay neural networks (Lang, Waibel, and Hinton 1990), the outputs for predicting target `Y_{i-1}` are used as inputs when processing target `Y_i`. Jordan networks (Jordan, 1986) are similar to time-delay neural networks except that the feedback is an exponential smooth of the sequence of output values. In Elman networks (Elman, 1990), the hidden unit activations that occur when processing target `Y_{i-1}` are used as inputs when processing target `Y_i`.

However, there are some problems that cannot be dealt with via recurrent networks alone. For example, many time series exhibit trend, meaning that the target values tend to go up over time, or that the target values tend to go down over time. For example, stock prices and many other financial variables usually go up. If today's price is higher than all previous prices, and you try to forecast tomorrow's price using today's price as a lagged input, you are extrapolating, and extrapolating is unreliable. The simplest methods for handling trend are:

- First fit a linear regression predicting the target values from the time, `Y_i = a + b T_i + noise`, where `a` and `b` are regression weights. Compute residuals `R_i = Y_i - (a + b T_i)`. Then train the network using `R_i` for the target and lagged values. This method is rather crude but may work for deterministic linear trends. Of course, for nonlinear trends, you would need to fit a nonlinear regression.

- Instead of using `Y_i` as a target, use `D_i = Y_i - Y_{i-1}` for the target and lagged values. This is called differencing and is the standard statistical method for handling nondeterministic (stochastic) trends. Sometimes it is necessary to compute differences of differences.

For an elementary discussion of trend and various other practical problems in forecasting time series with NNs, such as seasonality, see Masters (1993). For a more advanced discussion of NN forecasting of economic series, see Moody (1998).

There are several different ways to compute forecasts. For simplicity, let's assume you have a simple time series, `Y_1, ..., Y_99`, you want to forecast future values `Y_f` for `f > 99`, and you decide to use three lagged values as inputs. The possibilities include:

Single-step, one-step-ahead, or open-loop forecasting:
> Train a network with target `Y_i` and inputs `Y_{i-1}`, `Y_{i-2}`, and `Y_{i-3}`. Let the scalar function computed by the network be designated as `Net(.,.,.)` taking the three input values as arguments and returning the output (predicted) value. Then:
> forecast `Y_100` as `Net(Y_99,Y_98,Y_97)`
> forecast `Y_101` as `Net(Y_100,Y_99,Y_98)`
> forecast `Y_102` as `Net(Y_101,Y_100,Y_99)`
> forecast `Y_103` as `Net(Y_102,Y_101,Y_100)`
> forecast `Y_104` as `Net(Y_103,Y_102,Y_101)`
> and so on.

Multi-step or closed-loop forecasting:
> Train the network as above, but:
> forecast `Y_100` as `P_100 = Net(Y_99,Y_98,Y_97)`
> forecast `Y_101` as `P_101 = Net(P_100,Y_99,Y_98)`
> forecast `Y_102` as `P_102 = Net(P_101,P_100,Y_99)`
> forecast `Y_103` as `P_103 = Net(P_102,P_101,P_100)`
> forecast `Y_104` as `P_104 = Net(P_103,P_102,P_101)`
> and so on.

`N`-step-ahead forecasting:
> For, say, `N=3`, train the network as above, but:
> compute `P_100 = Net(Y_99,Y_98,Y_97)`
> compute `P_101 = Net(P_100,Y_99,Y_98)`
> forecast `Y_102` as `P_102 = Net(P_101,P_100,Y_99)`
> forecast `Y_103` as `P_103 = Net(P_102,P_101,Y_100)`
> forecast `Y_104` as `P_104 = Net(P_103,P_102,Y_101)`
> and so on.

Direct simultaneous long-term forecasting:
> Train a network with multiple targets `Y_i`, `Y_{i+1}`, and `Y_{i+2}` and inputs `Y_{i-1}`, `Y_{i-2}`, and `Y_{i-3}`. Let the vector function computed by the network be designated as `Net3(.,.,.)`, taking the three input values as arguments and returning the output (predicted) vector. Then:
> forecast `(Y_100,Y_101,Y_102)` as `Net3(Y_99,Y_98,Y_97)`

Which method you choose for computing forecasts will obviously depend in part on the requirements of your application. If you have yearly sales figures through 1999 and you need to forecast sales in 2003, you clearly can't use single-step forecasting. If you need to compute forecasts at a thousand different future times, using direct simultaneous long-term forecasting would require an extremely large network.

If a time series is a random walk, a well-trained network will predict `Y_i` by simply outputting `Y_{i-1}`. If you make a plot showing both the target values and the outputs, the two curves will almost coincide, except for being offset by one time step. People often mistakenly intrepret such a plot to indicate good forecasting accuracy, whereas in fact the network is virtually useless. In such situations, it is more enlightening to plot multi-step forecasts or `N`-step-ahead forecasts.

For general information on time-series forecasting, see the following URLs:

- Forecasting FAQs: http://forecasting.cwru.edu/faqs.html
- Forecasting Principles: http://hops.wharton.upenn.edu/forecast/
- Investment forecasts for stocks and mutual funds: http://www.coe.uncc.edu/~hphillip/

References:

Elman, J.L. (1990), "Finding structure in time," Cognitive Science, 14, 179-211.

Hertz, J., Krogh, A., and Palmer, R. (1991). *Introduction to the Theory of Neural Computation.* Addison-Wesley: Redwood City, California.

Horne, B. G. and Giles, C. L. (1995), "An experimental comparison of recurrent neural networks," In Tesauro, G., Touretzky, D., and Leen, T., editors, Advances in Neural Information Processing Systems 7, pp. 697-704. The MIT Press.

Jordan, M. I. (1986), "Attractor dynamics and parallelism in a connectionist sequential machine," In Proceedings of the Eighth Annual conference of the Cognitive Science Society, pages 531-546. Lawrence Erlbaum.

Judge, G.G., Griffiths, W.E., Hill, R.C., Lütkepohl, H., and Lee, T.-C. (1985), *The Theory and Practice of Econometrics,* NY: John Wiley & Sons.

Kremer, S.C. (199?), "Spatio-temporal Connectionist Networks: A Taxonomy and Review," http://hebb.cis.uoguelph.ca/~skremer/Teaching/27642/dynamic2/review.html.

Lang, K. J., Waibel, A. H., and Hinton, G. (1990), "A time-delay neural network architecture for isolated word recognition," Neural Networks, 3, 23-44.

Masters, T. (1993). *Practical Neural Network Recipes in C++,* San Diego: Academic Press.

Moody, J. (1998), "Forecasting the economy with neural nets: A survey of challenges and solutions," in Orr, G,B., and Mueller, K-R, eds., *Neural Networks: Tricks of the Trade,* Berlin: Springer.

Mozer, M.C. (1994), "Neural net architectures for temporal sequence processing," in Weigend, A.S. and Gershenfeld, N.A., eds. (1994) *Time Series Prediction: Forecasting the Future and Understanding the Past*, Reading, MA: Addison-Wesley, 243-264, http://www.cs.colorado.edu/~mozer/papers/timeseries.html.

Weigend, A.S. and Gershenfeld, N.A., eds. (1994) *Time Series Prediction: Forecasting the Future and Understanding the Past*, Reading, MA: Addison-Wesley.

Subject: How to learn an inverse of a function?

Ordinarily, NNs learn a function `Y = f(X)`, where `Y` is a vector of outputs, `X` is a vector of inputs, and `f()` is the function to be learned. Sometimes, however, you may want to learn an inverse of a function `f()`, that is, given `Y`, you want to be able to find an `X` such that `Y = f(X)`. In general, there may be many different `X`s that satisfy the equation `Y = f(X)`.

For example, in robotics (DeMers and Kreutz-Delgado, 1996, 1997), `X` might describe the positions of the joints in a robot's arm, while `Y` would describe the location of the robot's hand. There are simple formulas to compute the location of the hand given the positions of the joints, called the "forward kinematics" problem. But there is no simple formula for the "inverse kinematics" problem to compute positions of the joints that yield a given location for the hand. Furthermore, if the arm has several joints, there will usually be many different positions of the joints that yield the same location of the hand, so the forward kinematics function is many-to-one and has no unique inverse. Picking any `X` such that `Y = f(X)` is OK if the only aim is to position the hand at `Y`. However if the aim is to generate a series of points to move the hand through an arc this may be insufficient. In this case the series of `Xs` need to be in the same "branch" of the function space. Care must be taken to avoid solutions that yield inefficient or impossible movements of the arm.

As another example, consider an industrial process in which `X` represents settings of control variables imposed by an operator, and `Y` represents measurements of the product of the industrial process. The function `Y = f(X)` can be learned by a NN using conventional training methods. But the goal of the analysis may be to find control settings `X` that yield a product with specified measurements `Y`, in which case an inverse of `f(X)` is required. In industrial applications, financial considerations are important, so not just any setting `X` that yields the desired result `Y` may be acceptable. Perhaps a function can be specified that gives the cost of `X` resulting from energy consumption, raw materials, etc., in which case you would want to find the `X` that minimizes the cost function while satisfying the equation `Y = f(X)`.

The obvious way to try to learn an inverse function is to generate a set of training data from a given forward function, but designate `Y` as the input and `X` as the output when training the network. Using a least-squares error function, this approach will fail if `f()` is many-to-one. The problem is that for an input `Y`, the net will not learn any single `X` such that `Y = f(X)`, but will instead learn the arithmetic mean of all the `Xs` in the training set that satisfy the equation (Bishop, 1995, pp. 207-208). One solution to this difficulty is to construct a network that learns a mixture approximation to the conditional distribution of `X` given `Y` (Bishop, 1995, pp. 212-221). However, the mixture method will not work well in general for an `X` vector that is more than one-dimensional, such as `Y = X_1^2 + X_2^2`, since the number of mixture components required may increase exponentially with the dimensionality of `X`. And you are still left with the problem of extracting a single output vector from the mixture distribution, which is nontrivial if the mixture components overlap considerably. Another solution is to use a highly robust error function, such as a redescending M-estimator, that learns a single mode of the conditional distribution instead of learning the mean (Huber, 1981; Rohwer and van der Rest 1996). Additional regularization terms or constraints may be required to persuade the network to choose appropriately among several modes, and there may be severe problems with local optima.

Another approach is to train a network to learn the forward mapping `f()` and then numerically invert the function. Finding `X` such that `Y = f(X)` is simply a matter of solving a nonlinear system of equations, for which many algorithms can be found in the numerical analysis literature (Dennis and Schnabel 1983). One way to solve nonlinear equations is turn the problem into an optimization problem by minimizing `sum(Y_i-f(X_i))^2`. This method fits in nicely with the usual gradient-descent methods for training NNs (Kindermann and Linden 1990). Since the nonlinear equations will generally have multiple solutions, there may be severe problems with local optima, especially if some solutions are considered more desirable than others. You can deal with multiple solutions by inventing some objective function that measures the goodness of different solutions, and optimizing this objective function under the nonlinear constraint `Y = f(X)` using any of numerous algorithms for nonlinear programming (NLP; see Bertsekas, 1995, and other references under "What are conjugate gradients, Levenberg-Marquardt, etc.?") The power and flexibility of the nonlinear programming approach are offset by possibly high computational demands.

If the forward mapping `f()` is obtained by training a network, there will generally be some error in the network's outputs. The magnitude of this error can be difficult to estimate. The process of inverting a network can propagate this error, so the results should be checked carefully for validity and numerical stability. Some training methods can produce not just a point output but also a prediction interval (Bishop, 1995; White, 1992). You can take advantage of prediction intervals when inverting a network by using NLP methods. For example, you could try to find an `X` that minimizes the width of the prediction interval under the constraint that the equation `Y = f(X)` is satisfied. Or instead of requiring `Y = f(X)` be satisfied exactly, you could try to find an `X` such that the prediction interval is contained within some specified interval while minimizing some cost function.

For more mathematics concerning the inverse-function problem, as well as some interesting methods involving self-organizing maps, see DeMers and Kreutz-Delgado (1996, 1997). For NNs that are relatively easy to invert, see the Adaptive Logic Networks described in the software sections of the FAQ.

References:

Bertsekas, D. P. (1995), *Nonlinear Programming,* Belmont, MA: Athena Scientific.

Bishop, C.M. (1995), *Neural Networks for Pattern Recognition*, Oxford: Oxford University Press.

DeMers, D., and Kreutz-Delgado, K. (1996), "Canonical Parameterization of Excess motor degrees of freedom with self organizing maps", IEEE Trans Neural Networks, 7, 43-55.

DeMers, D., and Kreutz-Delgado, K. (1997), "Inverse kinematics of dextrous manipulators," in Omidvar, O., and van der Smagt, P., (eds.) *Neural Systems for Robotics,* San Diego: Academic Press, pp. 75-116.

Dennis, J.E. and Schnabel, R.B. (1983) *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* Prentice-Hall

Huber, P.J. (1981), *Robust Statistics,* NY: Wiley.

Kindermann, J., and Linden, A. (1990), "Inversion of Neural Networks by Gradient Descent," Parallel Computing, 14, 277-286, ftp://icsi.Berkeley.EDU/pub/ai/linden /KindermannLinden.IEEE92.ps.Z

Rohwer, R., and van der Rest, J.C. (1996), "Minimum description length, regularization, and multimodal data," Neural Computation, 8, 595-609.

White, H. (1992), "Nonparametric Estimation of Conditional Quantiles Using Neural Networks," in Page, C. and Le Page, R. (eds.), *Proceedings of the 23rd Sympsium on the Interface: Computing Science and Statistics,* Alexandria, VA: American Statistical Association, pp. 190-199.

Subject: How to get invariant recognition of images under translation, rotation, etc.?

See:

Bishop, C.M. (1995), *Neural Networks for Pattern Recognition*, Oxford: Oxford University Press, section 8.7.

Masters, T. (1994), *Signal and Image Processing with Neural Networks: A C++ Sourcebook*, NY: Wiley.

Soucek, B., and The IRIS Group (1992), *Fast Learning and Invariant Object Recognition,* NY: Wiley.

Squire, D. (1997), *Model-Based Neural Networks for Invariant Pattern Recognition,* http://cuiwww.unige.ch/~squire/publications.html

Laurenz Wiskott, bibliography on "Unsupervised Learning of Invariances in Neural Systems" http://www.cnl.salk.edu/~wiskott/Bibliographies/LearningInvariances.html

----

Subject: How to recognize handwritten characters?

URLS:

- Don Tveter's *The Pattern Recognition Basis of AI* at http://www.dontveter.com/basisofai/char.html
- Andras Kornai's homepage at http://www.cs.rice.edu/~andras/
- Yann LeCun's homepage at http://www.research.att.com/~yann/
  Data sets of handwritten digits can be found at http://www.research.att.com/~yann/exdb/mnist/

Other references:

Hastie, T., and Simard, P.Y. (1998), "Metrics and models for handwritten character recognition," Statistical Science, 13, 54-65.

Jackel, L.D. et al., (1994) "Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition", 1994 International Conference on Pattern Recognition, Jerusalem

LeCun, Y., Jackel, L.D., Bottou, L., Brunot, A., Cortes, C., Denker, J.S., Drucker, H., Guyon, I., Muller, U.A., Sackinger, E., Simard, P., and Vapnik, V. (1995), "Comparison of learning algorithms for handwritten digit recognition," in F. Fogelman and P. Gallinari, eds., International Conference on Artificial Neural Networks, pages 53-60, Paris.

Orr, G.B., and Mueller, K.-R., eds. (1998), *Neural Networks: Tricks of the Trade,* Berlin: Springer, ISBN 3-540-65311-2.

Subject: What about pulsed or spiking NNs?

The standard reference is:

Maass, W., and Bishop, C.M., eds. (1999) *Pulsed Neural Networks,* Cambridge, MA: The MIT Press, ISBN: 0262133504.

For more information on this book, see the section on "Pulsed/Spiking networks" under "Other notable books" in part 4 of the FAQ. Also see Professor Maass's web page at http://www.igi.tugraz.at/maass/.

Some other interesting URLs include:

- Laboratory of Computational Neuroscience (LCN) at the Swiss Federal Institute of Technology Lausanne, http://diwww.epfl.ch/mantra/mantra_bioneuro.html

- The notoriously hyped Berger-Liaw Neural Network Speaker-Independent Speech Recognition System, http://www.usc.edu/ext-relations/news_service/releases/stories/36013.html

----

Subject: What about Genetic Algorithms?

There are a number of definitions of GA (Genetic Algorithm). A possible one is

```
  A GA is an optimization program
  that starts with
  a population of encoded procedures,       (Creation of Life :-> )
  mutates them stochastically,              (Get cancer or so :-> )
  and uses a selection process              (Darwinism)
  to prefer the mutants with high fitness
  and perhaps a recombination process       (Make babies :-> )
  to combine properties of (preferably) the succesful mutants.
```

Genetic algorithms are just a special case of the more general idea of "evolutionary computation". There is a newsgroup that is dedicated to the field of evolutionary computation called comp.ai.genetic. It has a detailed FAQ posting which, for instance, explains the terms "Genetic Algorithm", "Evolutionary Programming", "Evolution Strategy", "Classifier System", and "Genetic Programming". That FAQ also contains lots of pointers to relevant literature, software, other sources of information, et cetera et cetera. Please see the comp.ai.genetic FAQ for further information.

For an entertaining introduction to evolutionary training of neural nets, see:

> David Fogel (2001), *Blondie24: Playing at the Edge of AI,* Morgan Kaufmann Publishers, ISBN: 1558607838

There are other books and papers by Fogel and his colleagues listed under "Checkers/Draughts" in the "Games, sports, gambling" section above.

For an extensive review, see:

> Yao, X. (1999), "Evolving Artificial Neural Networks," Proceedings of the IEEE, 87, 1423-1447, http://www.cs.bham.ac.uk/~xin/journal_papers.html

Here are some other on-line papers about evolutionary training of NNs:

- Backprop+GA: http://geneura.ugr.es/~pedro/G-Prop.htm
- LVQ+GA: http://geneura.ugr.es/g-lvq/g-lvq.html
- Very long chromosomes: ftp://archive.cis.ohio-state.edu/pub/neuroprose/korning.nnga.ps.Z

More URLs on genetic algorithms and NNs:

- Omri Weisman and Ziv Pollack's web page on "Neural Network Using Genetic Algorithms" at http://www.cs.bgu.ac.il/~omri/NNUGA/
- Christoph M. Friedrich's web page on Evolutionary algorithms and Artificial Neural Networks has a bibloigraphy and links to researchers at http://www.tussy.uni-wh.de/~chris/gann/gann.html
- Andrew Gray's Hybrid Systems FAQ at the University of Otago at http://divcom.otago.ac.nz:800/COM/INFOSCI/SMRL/people/andrew/publications/faq/hybrid/hybrid.htm

- Differential Evolution: http://www.icsi.berkeley.edu/~storn/code.html

For general information on GAs, try the links at http://www.shef.ac.uk/~gaipp/galinks.html and http://www.cs.unibo.it/~gaioni

----

Subject: What about Fuzzy Logic?

Fuzzy logic is an area of research based on the work of L.A. Zadeh. It is a departure from classical two-valued sets and logic, that uses "soft" linguistic (e.g. large, hot, tall) system variables and a continuous range of truth values in the interval [0,1], rather than strict binary (True or False) decisions and assignments.

Fuzzy logic is used where a system is difficult to model exactly (but an inexact model is available), is controlled by a human operator or expert, or where ambiguity or vagueness is common. A typical fuzzy system consists of a rule base, membership functions, and an inference procedure.

Most fuzzy logic discussion takes place in the newsgroup comp.ai.fuzzy (where there is a fuzzy logic FAQ) but there is also some work (and discussion) about combining fuzzy logic with neural network approaches in comp.ai.neural-nets.

Early work combining neural nets and fuzzy methods used competitive networks to generate rules for fuzzy systems (Kosko 1992). This approach is sort of a crude version of bidirectional counterpropagation (Hecht-Nielsen 1990) and suffers from the same deficiencies. More recent work (Brown and Harris 1994; Kosko 1997) has been based on the realization that a fuzzy system is a nonlinear mapping from an input space to an output space that can be parameterized in various ways and therefore can be adapted to data using the usual neural training methods (see "What is backprop?") or conventional numerical optimization algorithms (see "What are conjugate gradients, Levenberg-Marquardt, etc.?").

A neural net can incorporate fuzziness in various ways:

- The inputs can be fuzzy. Any garden-variety backprop net is fuzzy in this sense, and it seems rather silly to call a net "fuzzy" solely on this basis, although Fuzzy ART (Carpenter and Grossberg 1996) has no other fuzzy characteristics.
- The outputs can be fuzzy. Again, any garden-variety backprop net is fuzzy in this sense. But competitive learning nets ordinarily produce crisp outputs, so for competitive learning methods, having fuzzy output is a meaningful distinction. For example, fuzzy c-means clustering (Bezdek 1981) is meaningfully different from (crisp) k-means. Fuzzy ART does *not* have fuzzy outputs.
- The net can be interpretable as an adaptive fuzzy system. For example, Gaussian RBF nets and B-spline regression models (Dierckx 1995, van Rijckevorsal 1988) are fuzzy systems with adaptive weights (Brown and Harris 1994) and can legitimately be called neurofuzzy systems.
- The net can be a conventional NN architecture that operates on fuzzy numbers instead of real numbers (Lippe, Feuring and Mischke 1995).
- Fuzzy constraints can provide external knowledge (Lampinen and Selonen 1996).

More information on neurofuzzy systems is available online:

- The Fuzzy Logic and Neurofuzzy Resources page of the Image, Speech and Intelligent Systems (ISIS) research group at the University of Southampton, Southampton, Hampshire, UK: http://www-isis.ecs.soton.ac.uk/research/nfinfo/fuzzy.html.
- The Neuro-Fuzzy Systems Research Group's web page at Tampere University of Technology, Tampere, Finland: http://www.cs.tut.fi/~tpo/group.html and http://dmiwww.cs.tut.fi

/nfs/Welcome_uk.html
- Marcello Chiaberge's Neuro-Fuzzy page at http://polimage.polito.it/~marcello.
- The homepage of the research group on Neural Networks and Fuzzy Systems at the Institute of Knowledge Processing and Language Engineering, Faculty of Computer Science, University of Magdeburg, Germany, at http://www.neuro-fuzzy.de/
- Jyh-Shing Roger Jang's home page at http://www.cs.nthu.edu.tw/~jang/ with information on ANFIS (Adaptive Neuro-Fuzzy Inference Systems), articles on neuro-fuzzy systems, and more links.
- Andrew Gray's Hybrid Systems FAQ at the University of Otago at http://divcom.otago.ac.nz:800/COM/INFOSCI/SMRL/people/andrew/publications/faq/hybrid/hybrid.htm

References:

Bezdek, J.C. (1981), *Pattern Recognition with Fuzzy Objective Function Algorithms,* New York: Plenum Press.

Bezdek, J.C. & Pal, S.K., eds. (1992), *Fuzzy Models for Pattern Recognition,* New York: IEEE Press.

Brown, M., and Harris, C. (1994), *Neurofuzzy Adaptive Modelling and Control,* NY: Prentice Hall.

Carpenter, G.A. and Grossberg, S. (1996), "Learning, Categorization, Rule Formation, and Prediction by Fuzzy Neural Networks," in Chen, C.H. (1996), pp. 1.3-1.45.

Chen, C.H., ed. (1996) *Fuzzy Logic and Neural Network Handbook,* NY: McGraw-Hill, ISBN 0-07-011189-8.

Dierckx, P. (1995), *Curve and Surface Fitting with Splines,* Oxford: Clarendon Press.

Hecht-Nielsen, R. (1990), *Neurocomputing*, Reading, MA: Addison-Wesley.

Klir, G.J. and Folger, T.A.(1988), *Fuzzy Sets, Uncertainty, and Information,* Englewood Cliffs, N.J.: Prentice-Hall.

Kosko, B.(1992), *Neural Networks and Fuzzy Systems,* Englewood Cliffs, N.J.: Prentice-Hall.

Kosko, B. (1997), *Fuzzy Engineering,* NY: Prentice Hall.

Lampinen, J and Selonen, A. (1996), "Using Background Knowledge for Regularization of Multilayer Perceptron Learning", Submitted to International Conference on Artificial Neural Networks, ICANN'96, Bochum, Germany.

Lippe, W.-M., Feuring, Th. and Mischke, L. (1995), "Supervised learning in fuzzy neural networks," Institutsbericht Angewandte Mathematik und Informatik, WWU Muenster, I-12, http://wwwmath.uni-muenster.de/~feuring/WWW_literatur/bericht12_95.ps.gz

Nauck, D., Klawonn, F., and Kruse, R. (1997), *Foundations of Neuro-Fuzzy Systems,* Chichester: Wiley, ISBN 0-471-97151-0.

van Rijckevorsal, J.L.A. (1988), "Fuzzy coding and B-splines," in van Rijckevorsal, J.L.A., and de Leeuw, J., eds., Component and Correspondence Analysis, Chichester: John Wiley & Sons, pp. 33-54.

Subject: Unanswered FAQs

- How many training cases do I need?
- How should I split the data into training and validation sets?
- What error functions can be used?
- How can I select important input variables?
- Should NNs be used in safety-critical applications?

Subject: Other NN links?

- **Search engines**
  - Yahoo: http://www.yahoo.com/Science/Engineering/Electrical_Engineering/Neural_Networks/
  - Neuroscience Web Search: http://www.acsiom.org/nsr/neuro.html

- **Archives of NN articles and software**
  - **Neuroprose ftp archive site**

    ftp://archive.cis.ohio-state.edu/pub/neuroprose This directory contains technical reports as a public service to the connectionist and neural network scientific community.

  - **Finnish University Network archive site**

    A large collection of neural network papers and software at ftp://ftp.funet.fi/pub/sci/neural/ Contains all the public domain software and papers that they have been able to find. All of these files have been transferred from FTP sites in U.S. and are mirrored about every 3 months at fastest. Contact: neural-adm@ftp.funet.fi

  - **SEL-HPC Article Archive**

    http://liinwww.ira.uka.de/bibliography/Misc/SEL-HPC.html

  - **Machine Learning Papers**

    http://gubbio.cs.berkeley.edu/mlpapers/

- **Plain-text Tables of Contents of NN journals**

  Pattern Recognition Group, Department of Applied Physics,
  Faculty of Applied Sciences, Delft University of Technology,
  http://www.ph.tn.tudelft.nl/PRInfo/PRInfo/journals.html

- **The Collection of Computer Science Bibliographies: Bibliographies on Neural Networks** http://liinwww.ira.uka.de/bibliography/Neural/index.html

- **BibTeX data bases of NN journals**

  The Center for Computational Intelligence maintains BibTeX data bases of various NN journals, including IEEE Transactions on Neural Networks, Machine Learning, Neural Computation, and NIPS, at http://www.ci.tuwien.ac.at/docs/ci/bibtex_collection.html or ftp://ftp.ci.tuwien.ac.at/pub/texmf/bibtex/bib/.

- **NN events server**

  There is a WWW page for Announcements of Conferences, Workshops and Other Events on Neural Networks at IDIAP in Switzerland. WWW-Server: http://www.idiap.ch/html/idiap-networks.html.

- **Academic programs list**

  Rutvik Desai <rutvik@c3serve.c3.lanl.gov> has a compilation of acedemic programs offering interdeciplinary studies in computational neuroscience, AI, cognitive psychology etc. at http://www.cs.indiana.edu/hyplan/rudesai/cogsci-prog.html

  Links to neurosci, psychology, linguistics lists are also provided.

- **Neurosciences Internet Resource Guide**

  This document aims to be a guide to existing, free, Internet-accessible resources helpful to neuroscientists of all stripes. An ASCII text version (86K) is available in the Clearinghouse of Subject-Oriented Internet Resource Guides as follows:

  ftp://una.hh.lib.umich.edu/inetdirsstacks/neurosci:cormbonario, gopher://una.hh.lib.umich.edu/00/inetdirsstacks/neurosci:cormbonario, http://http2.sils.umich.edu/Public/nirg/nirg1.html.

- **Other WWW sites**

  In World-Wide-Web (WWW, for example via the xmosaic program) you can read neural network information for instance by opening one of the following uniform resource locators (URLs):
  http://www-xdiv.lanl.gov/XCM/neural/neural_announcements.html Los Alamos neural announcements and general information,
  http://www.ph.kcl.ac.uk/neuronet/ (NEuroNet, King's College, London),
  http://www.eeb.ele.tue.nl (Eindhoven, Netherlands),
  http://www.emsl.pnl.gov:2080/docs/cie/neural/ (Pacific Northwest National Laboratory, Richland, Washington, USA),
  http://www.cosy.sbg.ac.at/~rschwaig/rschwaig/projects.html (Salzburg, Austria),
  http://http2.sils.umich.edu/Public/nirg/nirg1.html (Michigan, USA),
  http://www.lpac.ac.uk/SEL-HPC/Articles/NeuralArchive.html (London),
  http://rtm.science.unitn.it/ Reactive Memory Search (Tabu Search) page (Trento, Italy),
  http://www.wi.leidenuniv.nl/art/ (ART WWW site, Leiden, Netherlands),
  http://nucleus.hut.fi/nnrc/ Helsinki University of Technology.
  http://www.pitt.edu/~mattf/NeuroRing.html links to neuroscience web pages
  http://www.arcade.uiowa.edu/hardin-www/md-neuro.htmlHardin Meta Directory web page for Neurology/Neurosciences.
  Many others are available too; WWW is changing all the time.

```
------------------------------------------------------------------------
```

That's all folks (End of the Neural Network FAQ).

```
Acknowledgements: Thanks to all the people who helped to get the stuff
                  above into the posting. I cannot name them all, because
                  I would make far too many errors then. :->

                  No?  Not good?  You want individual credit?
                  OK, OK. I'll try to name them all. But: no guarantee....

  THANKS FOR HELP TO:
(in alphabetical order of email adresses, I hope)
```

- Steve Ward <71561.2370@CompuServe.COM>
- Allen Bonde <ab04@harvey.gte.com>
- Accel Infotech Spore Pte Ltd <accel@solomon.technet.sg>
- Ales Krajnc <akrajnc@fagg.uni-lj.si>
- Alexander Linden <al@jargon.gmd.de>
- Matthew David Aldous <aldous@mundil.cs.mu.OZ.AU>
- S.Taimi Ames <ames@reed.edu>
- Axel Mulder <amulder@move.kines.sfu.ca>
- anderson@atc.boeing.com
- Andy Gillanders <andy@grace.demon.co.uk>
- Davide Anguita <anguita@ICSI.Berkeley.EDU>
- Avraam Pouliakis <apou@leon.nrcps.ariadne-t.gr>
- Kim L. Blackwell <avrama@helix.nih.gov>
- Mohammad Bahrami <bahrami@cse.unsw.edu.au>
- Paul Bakker <bakker@cs.uq.oz.au>
- Stefan Bergdoll <bergdoll@zxd.basf-ag.de>
- Jamshed Bharucha <bharucha@casbs.Stanford.EDU>
- Carl M. Cook <biocomp@biocomp.seanet.com>
- Yijun Cai <caiy@mercury.cs.uregina.ca>
- L. Leon Campbell <campbell@brahms.udel.edu>
- Cindy Hitchcock <cindyh@vnet.ibm.com>
- Clare G. Gallagher <clare@mikuni2.mikuni.com>
- Craig Watson <craig@magi.ncsl.nist.gov>
- Yaron Danon <danony@goya.its.rpi.edu>
- David Ewing <dave@ndx.com>
- David DeMers <demers@cs.ucsd.edu>
- Denni Rognvaldsson <denni@thep.lu.se>
- Duane Highley <dhighley@ozarks.sgcl.lib.mo.us>
- Dick.Keene@Central.Sun.COM
- DJ Meyer <djm@partek.com>
- Donald Tveter <don@dontveter.com>
- Daniel Tauritz <dtauritz@wi.leidenuniv.nl>
- Wlodzislaw Duch <duch@phys.uni.torun.pl>
- E. Robert Tisdale <edwin@flamingo.cs.ucla.edu>
- Athanasios Episcopos <episcopo@fire.camp.clarkson.edu>
- Frank Schnorrenberg <fs0997@easttexas.tamu.edu>
- Gary Lawrence Murphy <garym@maya.isis.org>
- gaudiano@park.bu.edu
- Lee Giles <giles@research.nj.nec.com>
- Glen Clark <opto!glen@gatech.edu>

- Phil Goodman <goodman@unr.edu>
- guy@minster.york.ac.uk
- Horace A. Vallas, Jr. <hav@neosoft.com>
- Joerg Heitkoetter <heitkoet@lusty.informatik.uni-dortmund.de>
- Ralf Hohenstein <hohenst@math.uni-muenster.de>
- Ian Cresswell <icressw@leopold.win-uk.net>
- Gamze Erten <ictech@mcimail.com>
- Ed Rosenfeld <IER@aol.com>
- Franco Insana <INSANA@asri.edu>
- Janne Sinkkonen <janne@iki.fi>
- Javier Blasco-Alberto <jblasco@ideafix.cps.unizar.es>
- Jean-Denis Muller <jdmuller@vnet.ibm.com>
- Jeff Harpster <uu0979!jeff@uu9.psi.com>
- Jonathan Kamens <jik@MIT.Edu>
- J.J. Merelo <jmerelo@geneura.ugr.es>
- Dr. Jacek Zurada <jmzura02@starbase.spd.louisville.edu>
- Jon Gunnar Solheim <jon@kongle.idt.unit.no>
- Josef Nelissen <jonas@beor.informatik.rwth-aachen.de>
- Joey Rogers <jrogers@buster.eng.ua.edu>
- Subhash Kak <kak@gate.ee.lsu.edu>
- Ken Karnofsky <karnofsky@mathworks.com>
- Kjetil.Noervaag@idt.unit.no
- Luke Koops <koops@gaul.csd.uwo.ca>
- Kurt Hornik <Kurt.Hornik@tuwien.ac.at>
- Thomas Lindblad <lindblad@kth.se>
- Clark Lindsey <lindsey@particle.kth.se>
- Lloyd Lubet <llubet@rt66.com>
- William Mackeown <mackeown@compsci.bristol.ac.uk>
- Maria Dolores Soriano Lopez <maria@vaire.imib.rwth-aachen.de>
- Mark Plumbley <mark@dcs.kcl.ac.uk>
- Peter Marvit <marvit@cattell.psych.upenn.edu>
- masud@worldbank.org
- Miguel A. Carreira-Perpinan<mcarreir@moises.ls.fi.upm.es>
- Yoshiro Miyata <miyata@sccs.chukyo-u.ac.jp>
- Madhav Moganti <mmogati@cs.umr.edu>
- Jyrki Alakuijala <more@ee.oulu.fi>
- Jean-Denis Muller <muller@bruyeres.cea.fr>
- Michael Reiss <m.reiss@kcl.ac.uk>
- mrs@kithrup.com
- Maciek Sitnik <msitnik@plearn.edu.pl>
- R. Steven Rainwater <ncc@ncc.jvnc.net>
- Nigel Dodd <nd@neural.win-uk.net>
- Barry Dunmall <neural@nts.sonnet.co.uk>
- Paolo Ienne <Paolo.Ienne@di.epfl.ch>
- Paul Keller <pe_keller@ccmail.pnl.gov>
- Peter Hamer <P.G.Hamer@nortel.co.uk>
- Pierre v.d. Laar <pierre@mbfys.kun.nl>
- Michael Plonski <plonski@aero.org>
- Lutz Prechelt <prechelt@ira.uka.de> [creator of FAQ]
- Richard Andrew Miles Outerbridge <ramo@uvphys.phys.uvic.ca>
- Rand Dixon <rdixon@passport.ca>

- Robin L. Getz <rgetz@esd.nsc.com>
- Richard Cornelius <richc@rsf.atd.ucar.edu>
- Rob Cunningham <rkc@xn.ll.mit.edu>
- Robert.Kocjancic@IJS.si
- Randall C. O'Reilly <ro2m@crab.psy.cmu.edu>
- Rutvik Desai <rudesai@cs.indiana.edu>
- Robert W. Means <rwmeans@hnc.com>
- Stefan Vogt <s_vogt@cis.umassd.edu>
- Osamu Saito <saito@nttica.ntt.jp>
- Scott Fahlman <sef+@cs.cmu.edu>
- <seibert@ll.mit.edu>
- Sheryl Cormicle <sherylc@umich.edu>
- Ted Stockwell <ted@aps1.spa.umn.edu>
- Stephanie Warrick <S.Warrick@cs.ucl.ac.uk>
- Serge Waterschoot <swater@minf.vub.ac.be>
- Thomas G. Dietterich <tgd@research.cs.orst.edu>
- Thomas.Vogel@cl.cam.ac.uk
- Ulrich Wendl <uli@unido.informatik.uni-dortmund.de>
- M. Verleysen <verleysen@dice.ucl.ac.be>
- VestaServ@aol.com
- Sherif Hashem <vg197@neutrino.pnl.gov>
- Matthew P Wiener <weemba@sagi.wistar.upenn.edu>
- Wesley Elsberry <welsberr@orca.tamu.edu>
- Dr. Steve G. Romaniuk <ZLXX69A@prodigy.com>

Special thanks to Gregory E. Heath <heath@ll.mit.edu> and Will Dwinnell <predictor@delphi.com> for years of stimulating and educational discussions on comp.ai.neurtal-nets.

The FAQ was created in June/July 1991 by Lutz Prechelt; he also maintained the FAQ until November 1995. Warren Sarle maintains the FAQ since December 1995.

```
Bye

  Warren & Lutz
```

Previous part is part 6.

Neural network FAQ / Warren S. Sarle, saswss@unx.sas.com