Types of artificial neural networks

From Wikipedia, the free encyclopedia

There are many **types of artificial neural networks (ANN)**.

An artificial neural network is a computational simulation of a biological neural network. These models mimic the behaviour of neurons and the electrical signals they convey between input (such as from the eyes or nerve endings in the hand), processing, and output from the brain (such as reacting to light, touch, or heat). The way neurons semantically communicate is an area of ongoing research.[1][2][3][4] Most artificial neural networks bear only some resemblance to their more complex biological counterparts, but are very effective at their intended tasks (e.g. classification or segmentation).

Other ANNs are adaptive systems used to model populations and environments.

Neural networks can be hardware- (neurons are represented by physical components) or software-based (computer models), and can use a variety of topologies and learning algorithms.

Contents

Feedforward neural network

The feedforward neural network was the first and arguably most simple type of artificial neural network devised. In this network the information moves in only one direction — forwards: From the input nodes data goes through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network. Feedforward networks can be constructed from different types of units, e.g. binary McCulloch-Pitts neurons, the simplest example being the perceptron. Continuous neurons, frequently with sigmoidal activation, are used in the context of backpropagation of error.

Radial basis function (RBF) network

Radial basis functions are powerful techniques for interpolation in multidimensional space. A RBF is a function which has built into a distance criterion with respect to a center. Radial basis functions have been applied in the area of neural networks where they may be used as a replacement for the sigmoidal hidden layer transfer characteristic in multi-layer perceptrons. RBF networks have two layers of processing: In the first, input is mapped onto each RBF in the 'hidden' layer. The RBF chosen is usually a Gaussian. In regression problems the output layer is then a linear combination of hidden layer values representing mean predicted output. The interpretation of this output layer value is the same as a regression model in statistics. In classification problems the output layer is typically a sigmoid function of a linear combination of hidden layer values, representing a posterior probability. Performance in both cases is often improved by shrinkage techniques, known as ridge regression in classical statistics and known to correspond to a prior belief in small parameter values (and therefore smooth output functions) in a Bayesian framework.

RBF networks have the advantage of not suffering from local minima in the same way as Multi-Layer Perceptrons. This is because the only parameters that are adjusted in the learning process are the linear mapping from hidden layer to output layer. Linearity ensures that the error surface is quadratic and therefore has a s§ingle easily found minimum. In regression problems this can be found in one matrix operation. In classification problems the fixed non-linearity introduced by the sigmoid output function is most efficiently dealt with using iteratively re-weighted least squares.

RBF networks have the disadvantage of requiring good coverage of the input space by radial basis functions. RBF centres are determined with reference to the distribution of the input data, but without reference to the prediction task. As a result, representational resources may be wasted on areas of the input space that are irrelevant to the learning task. A common solution is to associate each data point with its own centre, although this can make the linear system to be solved in the final layer rather large, and requires shrinkage techniques to avoid overfitting.

Associating each input datum with an RBF leads naturally to kernel methods such as support vector machines and Gaussian processes (the RBF is the kernel function). All three approaches use a non-linear kernel function to project the input data into a space where the learning problem can be solved using a linear model. Like Gaussian Processes, and unlike SVMs, RBF networks are typically trained in a Maximum Likelihood framework by maximizing the probability (minimizing the error) of the data under the model. SVMs take a different approach to avoiding overfitting by maximizing instead a margin. RBF networks are outperformed in most classification applications by SVMs. In regression applications they can be competitive when the dimensionality of the input space is relatively small.

Kohonen self-organizing network

The self-organizing map (SOM) invented by Teuvo Kohonen performs a form of unsupervised learning. A set of artificial neurons learn to map points in an input space to coordinates in an output space. The input space can have different dimensions and topology from the output space, and the SOM will attempt to preserve these.

Learning Vector Quantization

Learning Vector Quantization (LVQ) can also be interpreted as a neural network architecture. It was suggested by Teuvo Kohonen, originally. In LVQ, prototypical representatives of the classes parameterize, together with an appropriate distance measure, a distance-based classification scheme.

Recurrent neural network

Contrary to feedforward networks, recurrent neural networks (RNNs) are models with bi-directional data flow. While a feedforward network propagates data linearly from input to output, RNNs also propagate data from later processing stages to earlier stages. RNNs can be used as general sequence processors.

Fully recurrent network

This is the basic architecture developed in the 1980s: a network of neuron-like units, each with a directed connection to every other unit. Each unit has a time-varying real-valued (more than just zero or one) activation (output). Each connection has a modifiable real-valued weight. Some of the nodes are called input nodes, some output nodes, the rest hidden nodes. Most architectures below are special cases.

For supervised learning in discrete time settings, training sequences of real-valued input vectors become sequences of activations of the input nodes, one input vector at a time. At any given time step, each non-input unit computes its current activation as a nonlinear function of the weighted sum of the activations of all units from which it receives connections. There may be teacher-given target activations for some of the output units at certain time steps. For example, if the input sequence is a speech signal corresponding to a spoken digit, the final target output at the end of the sequence may be a label classifying the digit. For each sequence, its error is the sum of the deviations of all activations computed by the network from the corresponding target signals. For a training set of numerous sequences, the total error is the sum of the errors of all individual sequences.

To minimize total error, gradient descent can be used to change each weight in proportion to its derivative with respect to the error, provided the non-linear activation functions are differentiable. Various methods for doing so were developed in the 1980s and early 1990s by Paul Werbos, Ronald J. Williams, Tony Robinson, Jürgen Schmidhuber, Barak Pearlmutter, and others. The standard method is called "backpropagation through time" or BPTT, a generalization of back-propagation for feedforward networks.[5][6] A more computationally expensive online variant is called "Real-Time Recurrent Learning" or RTRL.[7][8] Unlike BPTT this algorithm is *local in time but not local in space*.[9][10] There also is an online hybrid between BPTT and RTRL with intermediate complexity,[11][12] and there are variants for continuous time.[13] A major problem with gradient descent for standard RNN architectures is that error gradients vanish exponentially quickly with the size of the time lag between important events, as first realized by Sepp Hochreiter in 1991.[14][15] The Long short term memory architecture overcomes these problems.[16]

In reinforcement learning settings, there is no teacher providing target signals for the RNN, instead a fitness function or reward function or utility function is occasionally used to evaluate the performance of the RNN, which is influencing its input stream through output units connected to actuators affecting the environment. Variants of evolutionary computation are often used to optimize the weight matrix.

Hopfield network

The Hopfield network (like similar attractor-based networks) is of historic interest although it is not a general RNN, as it is not designed to process sequences of patterns. Instead it requires stationary inputs. It is an RNN in which all connections are symmetric. Invented by John Hopfield in 1982 it guarantees that its dynamics will converge. If the connections are trained using Hebbian learning then the Hopfield network can perform as robust content-addressable memory, resistant to connection alteration.

Boltzmann machine

The Boltzmann machine can be thought of as a noisy Hopfield network. Invented by Geoff Hinton and Terry Sejnowski in 1985, the Boltzmann machine is important because it is one of the first neural networks to demonstrate learning of latent variables (hidden units). Boltzmann machine learning was at first slow to simulate, but the contrastive divergence algorithm of Geoff Hinton (circa 2000) allows models such as Boltzmann machines and Products of Experts to be trained much faster.

Simple recurrent networks

This special case of the basic architecture above was employed by Jeff Elman and Michael I. Jordan. A three-layer network is used, with the addition of a set of "context units" in the input layer. There are connections from the hidden layer (Elman) or from the output layer (Jordan) to these context units fixed with a weight of one.[17] At each time step, the input is propagated in a standard feedforward fashion, and then a simple backprop-like learning rule is applied (this rule is not performing proper gradient descent, however). The fixed back connections result in the context units always maintaining a copy of the previous values of the hidden units (since they propagate over the connections before the learning rule is applied).

Echo state network

The echo state network (ESN) is a recurrent neural network with a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that can change and be trained. ESN are good at reproducing certain time series.[18] A variant for spiking neurons is known as Liquid state machines.[19]

Long short term memory network

The long short term memory (LSTM), developed by Hochreiter and Schmidhuber in 1997,[16] is an artificial neural net structure that unlike traditional RNNs doesn't have the problem of vanishing gradients. It works even when there are long delays, and it can handle signals that have a mix of low and high frequency components. LSTM RNN outperformed other RNN and other sequence learning methods such as HMM in numerous applications such as language learning[20] and connected handwriting recognition.[21]

Bi-directional RNN

Invented by Schuster & Paliwal in 1997[22] bi-directional RNNs, or BRNNs, use a finite sequence to predict or label each element of the sequence based on both the past and the future context of the element. This is done by adding the outputs of two RNNs: one processing the sequence from left to right, the other one from right to left. The combined outputs are the predictions of the teacher-given target signals. This technique proved to be especially useful when combined with LSTM RNNs.[23]

Hierarchical RNN

There are many instances of hierarchical RNN whose elements are connected in various ways to decompose hierarchical behavior into useful subprograms.[24][25]

Stochastic neural networks

A stochastic neural network differs from a typical neural network because it introduces random variations into the network. In a probabilistic view of neural networks, such random variations can be viewed as a form of statistical sampling, such as Monte Carlo sampling.

Modular neural networks

Biological studies have shown that the human brain functions not as a single massive network, but as a collection of small networks. This realization gave birth to the concept of modular neural networks, in which several small networks cooperate or compete to solve problems.

Committee of machines

A committee of machines (CoM) is a collection of different neural networks that together "vote" on a given example. This generally gives a much better result compared to other neural network models. Because neural networks suffer from local minima, starting with the same architecture and training but using different initial random weights often gives vastly different networks. A CoM tends to stabilize the result.

The CoM is similar to the general machine learning *bagging* method, except that the necessary variety of machines in the committee is obtained by training from different random starting weights rather than training on different randomly selected subsets of the training data.

Associative neural network (ASNN)

The ASNN is an extension of the *committee of machines* that goes beyond a simple/weighted average of different models. ASNN (http://cogprints.soton.ac.uk/documents/disk0/00/00/14/41/index.html) represents a combination o an ensemble of feedforward neural networks and the k-nearest neighbor technique (kNN). It uses the correlation between ensemble responses as a measure of **distance** amid the analyzed cases for the kNN. This corrects the bias of the neural network ensemble. An associative neural network has a memory that can coincide with the training set. If new data become available, the network instantly improves its predictive ability and provides data approximation (self-learn the data) without a need to retrain the ensemble. Another important feature of ASNN is the possibility to interpret neural network results by analysis of correlations between data cases in the space of models. The method is demonstrated at www.vcclab.org (http://www.vcclab.org/lab/asnn), where it can be used online or downloaded.

Physical neural network

A physical neural network includes electrically adjustable resistance material to simulate artificial synapses. Examples include the ADALINE neural network developed by Bernard Widrow in the 1960s and the memristor based neural network developed by Greg Snider of HP Labs in 2008.

Other types of networks

These special networks do not fit in any of the previous categories.

Holographic associative memory

Holographic associative memory represents a family of analog, correlation-based, associative, stimulus-response memories, where information is mapped onto the phase orientation of complex numbers operating.

Instantaneously trained networks

Instantaneously trained neural networks (ITNNs) were inspired by the phenomenon of short-term learning that seems to occur instantaneously. In these networks the weights of the hidden and the output layers are mapped directly from the training vector data. Ordinarily, they work on binary data, but versions for continuous data that require small additional processing are also available.

Spiking neural networks

Spiking neural networks (SNNs) are models which explicitly take into account the timing of inputs. The network input and output are usually represented as series of spikes (delta function or more complex shapes). SNNs have an advantage of being able to process information in the time domain (signals that vary over time). They are often implemented as recurrent networks. SNNs are also a form of pulse computer.

Spiking neural networks with axonal conduction delays exhibit polychronization, and hence could have a very large memory capacity.[26]

Networks of spiking neurons — and the temporal correlations of neural assemblies in such networks — have been used to model figure/ground separation and region linking in the visual system (see, for example, Reitboeck et al.in Haken and Stadler: Synergetics of the Brain. Berlin, 1989).

In June 2005 IBM announced construction of a Blue Gene supercomputer dedicated to the simulation of a large recurrent spiking neural network.[27]

Gerstner and Kistler have a freely available online textbook on Spiking Neuron Models (http://icwww.epfl.ch/~gerstner/SPNM/SPNM.html).

Dynamic neural networks

Dynamic neural networks not only deal with nonlinear multivariate behaviour, but also include (learning of) time-dependent behaviour such as various transient phenomena and delay effects. Techniques to estimate a system process from observed data fall under the general category of system identification.

Cascading neural networks

Cascade Correlation is an architecture and supervised learning algorithm developed by Scott Fahlman and Christian Lebiere. Instead of just adjusting the weights in a network of fixed topology, Cascade-Correlation begins with a minimal network, then automatically trains and adds new hidden units one by one, creating a multi-layer structure. Once a new hidden unit has been added to the network, its input-side weights are frozen. This unit then becomes a permanent feature-detector in the network, available for producing outputs or for creating other, more complex feature detectors. The Cascade-Correlation architecture has several advantages over existing algorithms: it learns very quickly, the network determines its own size and topology, it retains the structures it has built even if the training set changes, and it requires no back-propagation of error signals through the connections of the network.

Neuro-fuzzy networks

A neuro-fuzzy network is a fuzzy inference system in the body of an artificial neural network. Depending on the *FIS* type, there are several layers that simulate the processes involved in a *fuzzy inference* like fuzzification, inference, aggregation and defuzzification. Embedding an *FIS* in a general structure of an *ANN* has the benefit of using available *ANN* training methods to find the parameters of a fuzzy system.

Compositional pattern-producing networks

Compositional pattern-producing networks (CPPNs) are a variation of ANNs which differ in their set of activation functions and how they are applied. While typical ANNs often contain only sigmoid functions (and sometimes Gaussian functions), CPPNs can include both types of functions and many others. Furthermore, unlike typical

ANNs, CPPNs are applied across the entire space of possible inputs so that they can represent a complete image. Since they are compositions of functions, CPPNs in effect encode images at infinite resolution and can be sampled for a particular display at whatever resolution is optimal.

One-shot associative memory

This type of network can add new patterns without the need for re-training. It is done by creating a specific memory structure, which assigns each new pattern to an orthogonal plane using adjacently connected hierarchical arrays.[28] The network offers real-time pattern recognition and high scalability, it however requires parallel processing and is thus best suited for platforms such as Wireless sensor networks (WSN), Grid computing, and GPGPUs.

Hierarchical temporal memory

Hierarchical temporal memory (HTM) is an online machine learning model developed by Jeff Hawkins and Dileep George of Numenta, Inc. that models some of the structural and algorithmic properties of the neocortex. HTM is a biomimetic model based on the memory-prediction theory of brain function described by Jeff Hawkins in his book *On Intelligence*. HTM is a method for discovering and inferring the high-level causes of observed input patterns and sequences, thus building an increasingly complex model of the world.

Jeff Hawkins states that HTM does not present any new idea or theory, but combines existing ideas to mimic the neocortex with a simple design that provides a large range of capabilities. HTM combines and extends approaches used in Bayesian networks, spatial and temporal clustering algorithms, while using a tree-shaped hierarchy of nodes that is common in neural networks.

See also

- Adaptive resonance theory
- Artificial life
- Autoassociative memory
- Autoencoder
- Biologically inspired computing
- Blue brain
- Connectionist expert system
- Decision tree
- Expert system
- Genetic algorithm
- In Situ Adaptive Tabulation
- Linear discriminant analysis
- Logistic regression
- Multilayer perceptron
- Neural Gas

- Neuroevolution, NeuroEvolution of Augmented Topologies (NEAT)
- Ni1000 chip
- Optical neural network
- Particle swarm optimization
- Predictive analytics
- Principal components analysis
- Simulated annealing
- Systolic array
- Time delay neural network (TDNN)

References

1. ^ University Of Southern California. (2004, June 16). Gray Matters: New Clues Into How Neurons Process Information. ScienceDaily (http://www.sciencedaily.com/releases/2004/06/040616064016.htm) Quote: "... "It's amazing that after a hundred years of modern neuroscience research, we still don't know the basic information processing functions of a neuron," said Bartlett Mel..."
2. ^ Weizmann Institute of Science. (2007, April 2). It's Only A Game Of Chance: Leading Theory Of Perception Called Into Question. ScienceDaily (http://www.sciencedaily.com/releases/2007/03/070327144225.htm) Quote: "..."Since the 1980s, many neuroscientists believed they possessed the key for finally beginning to understand the workings of the brain. But we have provided strong evidence to suggest that the brain may not encode information using precise patterns of activity."..."
3. ^ University Of California - Los Angeles (2004, December 14). UCLA Neuroscientist Gains Insights Into Human Brain From Study Of Marine Snail. ScienceDaily (http://www.sciencedaily.com/releases/2004/12/041208084855.htm) Quote: "..."Our work implies that the brain mechanisms for forming these kinds of associations might be extremely similar in snails and higher organisms...We don't fully understand even very simple kinds of learning in these animals."..."
4. ^ Yale University. (2006, April 13). Brain Communicates In Analog And Digital Modes Simultaneously. ScienceDaily (http://www.sciencedaily.com/releases/2006/04/060412223937.htm) Quote: "...McCormick said future investigations and models of neuronal operation in the brain will need to take into account the mixed analog-digital nature of communication. Only with a thorough understanding of this mixed mode of signal transmission will a truly in depth understanding of the brain and its disorders be achieved, he said..."
5. ^ P. J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. Neural Networks, 1, 1988.
6. ^ David E. Rumelhart; Geoffrey E. Hinton; Ronald J. Williams. Learning Internal Representations by Error Propagation.
7. ^ A. J. Robinson and F. Fallside. The utility driven dynamic error propagation network. Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, 1987.
8. ^ R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. In Back-propagation: Theory, Architectures and Applications. Hillsdale, NJ: Erlbaum, 1994.

9. ^ J. Schmidhuber. A local learning algorithm for dynamic feedforward and recurrent networks. Connection Science, 1(4):403–412, 1989.
10. ^ Neural and Adaptive Systems: Fundamentals through Simulation. J.C. Principe, N.R. Euliano, W.C. Lefebvre
11. ^ J. Schmidhuber. A fixed size storage O(n3) time complexity learning algorithm for fully recurrent continually running networks. Neural Computation, 4(2):243–248, 1992.
12. ^ R. J. Williams. Complexity of exact gradient computation algorithms for recurrent neural networks. Technical Report Technical Report NU-CCS-89-27, Boston: Northeastern University, College of Computer Science, 1989.
13. ^ B. A. Pearlmutter. Learning state space trajectories in recurrent neural networks. Neural Computation, 1(2):263–269, 1989.
14. ^ S. Hochreiter. Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut f. Informatik, Technische Univ. Munich, 1991.
15. ^ S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In S. C. Kremer and J. F. Kolen, editors, A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press, 2001.
16. ^ ***a*** ***b*** S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735– 1780, 1997.
17. ^ Neural Networks as Cybernetic Systems 2nd and revised edition, Holk Cruse [1] (http://www.brains-minds-media.org/archive/615/bmm615.pdf)
18. ^ H. Jaeger. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science, 304:78–80, 2004.
19. ^ W. Maass, T. Natschläger, and H. Markram. A fresh look at real-time computation in generic recurrent neural circuits. Technical report, Institute for Theoretical Computer Science, TU Graz, 2002.
20. ^ F. A. Gers and J. Schmidhuber. LSTM recurrent networks learn simple context free and context sensitive languages. IEEE Transactions on Neural Networks, 12(6):1333–1340, 2001.
21. ^ A. Graves, J. Schmidhuber. Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. Advances in Neural Information Processing Systems 22, NIPS'22, p 545-552, Vancouver, MIT Press, 2009.
22. ^ Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. IEEE Trans- actions on Signal Processing, 45:2673–2681, November 1997.
23. ^ A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 18:602–610, 2005.
24. ^ J. Schmidhuber. Learning complex, extended sequences using the principle of history compression. Neural Computation, 4(2):234-242, 1992
25. ^ Dynamic Representation of Movement Primitives in an Evolved Recurrent Neural Network (http://www.bdc.brain.riken.go.jp/~rpaine/PaineTaniSAB2004_h.pdf)
26. ^ Izhikevich EM (February 2006). "Polychronization: computation with spikes". *Neural Comput* **18** (2): 245–82. doi:10.1162/089976606775093882 (http://dx.doi.org/10.1162%2F089976606775093882). PMID 16378515 (https://www.ncbi.nlm.nih.gov/pubmed/16378515).
27. ^ "IBM Research | Press Resources | IBM and EPFL Join Forces to Uncover the Secrets of Cognitive Intelligence" (http://domino.research.ibm.com/comm/pr.nsf/pages/news.20050606_CognitiveIntelligence.html). Retrieved 2009-05-02.

28. ^ B.B. Nasution, A.I. Khan, A Hierarchical Graph Neuron Scheme for Real-Time Pattern Recognition (http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4359217), IEEE Transactions on Neural Networks, vol 19(2), 212-229, Feb. 2008

Retrieved from "http://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&oldid=613937781"

Categories: Computational statistics | Artificial neural networks | Classification algorithms | Computational neuroscience

---