Types of artificial neural networks

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#mw-head)[Jump to search](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#searchInput)

|  |  |
| --- | --- |
|  | This article **needs attention from an expert in Computer science**. Please add a *reason* or a *talk* parameter to this template to explain the issue with the article. [WikiProject Computer science](https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Computer_science" \o "Wikipedia:WikiProject Computer science) may be able to help recruit an expert. *(August 2019)* |

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

[Artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_network) are [computational models](https://en.wikipedia.org/wiki/Computational_model) inspired by [biological neural networks](https://en.wikipedia.org/wiki/Biological_neural_network), and are used to [approximate](https://en.wikipedia.org/wiki/Universal_approximation_theorem) [functions](https://en.wikipedia.org/wiki/Function_(mathematics)) that are generally unknown. Particularly, they are inspired by the behaviour of [neurons](https://en.wikipedia.org/wiki/Neuron) 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]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-1)[[2]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-2)[[3]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-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).

Some artificial neural networks are [adaptive systems](https://en.wikipedia.org/wiki/Adaptive_system) and are used for example to [model populations](https://en.wikipedia.org/wiki/Population_model) and environments, which constantly change.

Neural networks can be hardware- (neurons are represented by physical components) or [software-based](https://en.wikipedia.org/wiki/Neural_network_software) (computer models), and can use a variety of topologies and learning algorithms.

Feedforward[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=1)]

*Main article:*[*Feedforward neural network*](https://en.wikipedia.org/wiki/Feedforward_neural_network)

The feedforward neural network was the first and simplest type. In this network the information moves only from the input layer directly through any hidden layers to the output layer without cycles/loops. Feedforward networks can be constructed with various types of units, such as binary [McCulloch–Pitts neurons](https://en.wikipedia.org/wiki/McCulloch%E2%80%93Pitts_neuron), the simplest of which is the [perceptron](https://en.wikipedia.org/wiki/Perceptron). Continuous neurons, frequently with sigmoidal activation, are used in the context of [backpropagation](https://en.wikipedia.org/wiki/Backpropagation).

**Group method of data handling**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=2)]

*Main article:*[*Group method of data handling*](https://en.wikipedia.org/wiki/Group_method_of_data_handling)

The Group Method of Data Handling (GMDH)[[5]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ivak1968-5) features fully automatic structural and parametric model optimization. The node activation functions are [Kolmogorov](https://en.wikipedia.org/wiki/Andrey_Kolmogorov)–Gabor polynomials that permit additions and multiplications. It uses a deep multilayer [perceptron](https://en.wikipedia.org/wiki/Perceptron) with eight layers.[[6]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ivak1971-6) It is a [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) network that grows layer by layer, where each layer is trained by [regression analysis](https://en.wikipedia.org/wiki/Regression_analysis). Useless items are detected using a validation set, and pruned through [regularization](https://en.wikipedia.org/wiki/Regularization_(mathematics)). The size and depth of the resulting network depends on the task.[[7]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-kondo2008-7)

**Autoencoder**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=3)]

*Main article:*[*Autoencoder*](https://en.wikipedia.org/wiki/Autoencoder)

An autoencoder, autoassociator or Diabolo network[[8]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-bengio-8):19 is similar to the [multilayer perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron) (MLP) – with an input layer, an output layer and one or more hidden layers connecting them. However, the output layer has the same number of units as the input layer. Its purpose is to reconstruct its own inputs (instead of emitting a target value). Therefore, autoencoders are [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning) models. An autoencoder is used for [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning) of [efficient codings](https://en.wikipedia.org/wiki/Feature_learning),[[9]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-9)[[10]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-10) typically for the purpose of [dimensionality reduction](https://en.wikipedia.org/wiki/Dimensionality_reduction) and for learning [generative models](https://en.wikipedia.org/wiki/Generative_model) of data.[[11]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-VAE-11)[[12]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-gan_faces-12)

**Probabilistic**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=4)]

*Main article:*[*Probabilistic neural network*](https://en.wikipedia.org/wiki/Probabilistic_neural_network)

A probabilistic neural network (PNN) is a four-layer feedforward neural network. The layers are Input, hidden, pattern/summation and output. In the PNN algorithm, the parent probability distribution function (PDF) of each class is approximated by a [Parzen window](https://en.wikipedia.org/wiki/Kernel_density_estimation" \o "Kernel density estimation) and a non-parametric function. Then, using PDF of each class, the class probability of a new input is estimated and Bayes’ rule is employed to allocate it to the class with the highest posterior probability.[[13]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-13) It was derived from the [Bayesian network](https://en.wikipedia.org/wiki/Bayesian_network)[[14]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-14) and a statistical algorithm called [Kernel Fisher discriminant analysis](https://en.wikipedia.org/wiki/Kernel_Fisher_discriminant_analysis).[[15]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-15) It is used for classification and pattern recognition.

**Time delay**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=5)]

*Main article:*[*Time delay neural network*](https://en.wikipedia.org/wiki/Time_delay_neural_network)

A time delay neural network (TDNN) is a feedforward architecture for sequential data that recognizes features independent of sequence position. In order to achieve time-shift invariance, delays are added to the input so that multiple data points (points in time) are analyzed together.

It usually forms part of a larger pattern recognition system. It has been implemented using a [perceptron](https://en.wikipedia.org/wiki/Perceptron) network whose connection weights were trained with back propagation (supervised learning).[[16]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-16)

**Convolutional**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=6)]

*Main article:*[*Convolutional neural network*](https://en.wikipedia.org/wiki/Convolutional_neural_network)

A convolutional neural network (CNN, or ConvNet or shift invariant or space invariant) is a class of deep network, composed of one or more [convolutional](https://en.wikipedia.org/wiki/Convolution) layers with fully connected layers (matching those in typical ANNs) on top.[[17]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-:1-17)[[18]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-:0-18) It uses tied weights and pooling layers. In particular, max-pooling.[[19]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Weng19932-19) It is often structured via Fukushima's convolutional architecture.[[20]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-FUKU1980-20) They are variations of [multilayer perceptrons](https://en.wikipedia.org/wiki/Multilayer_perceptron) that use minimal [preprocessing](https://en.wikipedia.org/wiki/Data_pre-processing).[[21]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-LeCun-21) This architecture allows CNNs to take advantage of the 2D structure of input data.

Its unit connectivity pattern is inspired by the organization of the visual cortex. Units respond to stimuli in a restricted region of space known as the receptive field. Receptive fields partially overlap, over-covering the entire [visual field](https://en.wikipedia.org/wiki/Visual_field). Unit response can be approximated mathematically by a [convolution](https://en.wikipedia.org/wiki/Convolution) operation.[[22]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-deeplearning-22)

CNNs are suitable for processing visual and other two-dimensional data.[[23]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-LECUN1989-23)[[24]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-lecun2016slides-24) They have shown superior results in both image and speech applications. They can be trained with standard backpropagation. CNNs are easier to train than other regular, deep, feed-forward neural networks and have many fewer parameters to estimate.[[25]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-STANCNN-25)

[Capsule Neural Networks](https://en.wikipedia.org/wiki/Capsule_neural_network) (CapsNet) add structures called capsules to a CNN and reuse output from several capsules to form more stable (with respect to various perturbations) representations.[[26]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-26)

Examples of applications in computer vision include [DeepDream](https://en.wikipedia.org/wiki/DeepDream" \o "DeepDream)[[27]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-deepdream-27) and [robot navigation](https://en.wikipedia.org/wiki/Robot_navigation).[[28]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-28) They have wide applications in [image and video recognition](https://en.wikipedia.org/wiki/Computer_vision), [recommender systems](https://en.wikipedia.org/wiki/Recommender_system)[[29]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-29) and [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing).[[30]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-30)

**Deep stacking network**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=7)]

A deep stacking network (DSN)[[31]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref17-31) (deep convex network) is based on a hierarchy of blocks of simplified neural network modules. It was introduced in 2011 by Deng and Dong.[[32]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref16-32) It formulates the learning as a [convex optimization problem](https://en.wikipedia.org/wiki/Convex_optimization_problem) with a [closed-form solution](https://en.wikipedia.org/wiki/Closed-form_expression), emphasizing the mechanism's similarity to [stacked generalization](https://en.wikipedia.org/wiki/Ensemble_learning).[[33]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref18-33) Each DSN block is a simple module that is easy to train by itself in a [supervised](https://en.wikipedia.org/wiki/Supervised_learning) fashion without backpropagation for the entire blocks.[[34]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-34)

Each block consists of a simplified [multi-layer perceptron](https://en.wikipedia.org/wiki/Multi-layer_perceptron) (MLP) with a single hidden layer. The hidden layer ***h*** has logistic [sigmoidal](https://en.wikipedia.org/wiki/Sigmoid_function) [units](https://en.wikipedia.org/wiki/Artificial_neuron), and the output layer has linear units. Connections between these layers are represented by weight matrix ***U;*** input-to-hidden-layer connections have weight matrix ***W***. Target vectors ***t*** form the columns of matrix ***T***, and the input data vectors ***x*** form the columns of matrix ***X.*** The matrix of hidden units is {\displaystyle {\boldsymbol {H}}=\sigma ({\boldsymbol {W}}^{T}{\boldsymbol {X}})}. Modules are trained in order, so lower-layer weights ***W*** are known at each stage. The function performs the element-wise [logistic sigmoid](https://en.wikipedia.org/wiki/Logistic_function) operation. Each block estimates the same final label class *y*, and its estimate is concatenated with original input ***X*** to form the expanded input for the next block. Thus, the input to the first block contains the original data only, while downstream blocks' input adds the output of preceding blocks. Then learning the upper-layer weight matrix ***U*** given other weights in the network can be formulated as a convex optimization problem:

{\displaystyle \min \_{U^{T}}f=\|{\boldsymbol {U}}^{T}{\boldsymbol {H}}-{\boldsymbol {T}}\|\_{F}^{2},}

which has a closed-form solution.[[31]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref17-31)

Unlike other deep architectures, such as DBNs, the goal is not to discover the transformed [feature](https://en.wikipedia.org/wiki/Feature_(machine_learning)) representation. The structure of the hierarchy of this kind of architecture makes parallel learning straightforward, as a batch-mode optimization problem. In purely [discriminative tasks](https://en.wikipedia.org/wiki/Discriminative_model), DSNs outperform conventional [DBNs](https://en.wikipedia.org/wiki/Deep_belief_network).

**Tensor deep stacking networks**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=8)]

This architecture is a DSN extension. It offers two important improvements: it uses higher-order information from [covariance](https://en.wikipedia.org/wiki/Covariance) statistics, and it transforms the [non-convex problem](https://en.wikipedia.org/wiki/Convex_optimization) of a lower-layer to a convex sub-problem of an upper-layer.[[35]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref19-35) TDSNs use covariance statistics in a [bilinear mapping](https://en.wikipedia.org/wiki/Bilinear_map) from each of two distinct sets of hidden units in the same layer to predictions, via a third-order [tensor](https://en.wikipedia.org/wiki/Tensor).

While parallelization and scalability are not considered seriously in conventional DNNs,[[36]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref26-36)[[37]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref27-37)[[38]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref28-38) all learning for DSNs and TDSNs is done in batch mode, to allow parallelization.[[39]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref162-39)[[40]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref172-40) Parallelization allows scaling the design to larger (deeper) architectures and data sets.

The basic architecture is suitable for diverse tasks such as [classification](https://en.wikipedia.org/wiki/Statistical_classification) and [regression](https://en.wikipedia.org/wiki/Regression_analysis).

Regulatory feedback[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=9)]

Regulatory feedback networks started as a model to explain brain phenomena found during recognition including network-wide [bursting](https://en.wikipedia.org/wiki/Bursting) and [difficulty with similarity](https://en.wikipedia.org/wiki/Visual_search) found universally in sensory recognition. A mechanism to perform optimization during recognition is created using inhibitory feedback connections back to the same inputs that activate them. This reduces requirements during learning and allows learning and updating to be easier while still being able to perform complex recognition.

Radial basis function (RBF)[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=10)]

*Main article:*[*Radial basis function network*](https://en.wikipedia.org/wiki/Radial_basis_function_network)

Radial basis functions are functions that have a distance criterion with respect to a center. Radial basis functions have been applied as a replacement for the sigmoidal hidden layer transfer characteristic in multi-layer perceptrons. RBF networks have two layers: 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 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](https://en.wikipedia.org/wiki/Regression_analysis) in statistics. In classification problems the output layer is typically a [sigmoid function](https://en.wikipedia.org/wiki/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](https://en.wikipedia.org/wiki/Ridge_regression) in classical statistics. This corresponds to a prior belief in small parameter values (and therefore smooth output functions) in a [Bayesian](https://en.wikipedia.org/wiki/Bayesian_statistics) framework.

RBF networks have the advantage of avoiding 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 single 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](https://en.wikipedia.org/wiki/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 task. A common solution is to associate each data point with its own centre, although this can expand the linear system to be solved in the final layer and requires shrinkage techniques to avoid [overfitting](https://en.wikipedia.org/wiki/Overfitting).

Associating each input datum with an RBF leads naturally to kernel methods such as [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machine) (SVM) and Gaussian processes (the RBF is the [kernel function](https://en.wikipedia.org/wiki/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). SVMs avoid overfitting by maximizing instead a margin. SVMs outperform RBF networks in most classification applications. In regression applications they can be competitive when the dimensionality of the input space is relatively small.

**How RBF networks work**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=11)]

RBF neural networks are conceptually similar to [K-Nearest Neighbor](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm) (k-NN) models. The basic idea is that similar inputs produce similar outputs.

In the case in of a training set has two predictor variables, x and y and the target variable has two categories, positive and negative. Given a new case with predictor values x=6, y=5.1, how is the target variable computed?

The nearest neighbor classification performed for this example depends on how many neighboring points are considered. If 1-NN is used and the closest point is negative, then the new point should be classified as negative. Alternatively, if 9-NN classification is used and the closest 9 points are considered, then the effect of the surrounding 8 positive points may outweigh the closest 9 (negative) point.

An RBF network positions neurons in the space described by the predictor variables (x,y in this example). This space has as many dimensions as predictor variables. The Euclidean distance is computed from the new point to the center of each neuron, and a radial basis function (RBF) (also called a kernel function) is applied to the distance to compute the weight (influence) for each neuron. The radial basis function is so named because the radius distance is the argument to the function.

*Weight* = RBF(*distance*)

**Radial Basis Function**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=12)]

The value for the new point is found by summing the output values of the RBF functions multiplied by weights computed for each neuron.

The radial basis function for a neuron has a center and a radius (also called a spread). The radius may be different for each neuron, and, in RBF networks generated by DTREG, the radius may be different in each dimension.

With larger spread, neurons at a distance from a point have a greater influence.

**Architecture**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=13)]

RBF networks have three layers:

* **Input layer:** One neuron appears in the input layer for each predictor variable. In the case of [categorical variables](https://en.wikipedia.org/wiki/Categorical_variable), N-1 neurons are used where N is the number of categories. The input neurons standardizes the value ranges by subtracting the [median](https://en.wikipedia.org/wiki/Median) and dividing by the [interquartile](https://en.wikipedia.org/wiki/Interquartile_range) range. The input neurons then feed the values to each of the neurons in the hidden layer.
* **Hidden layer:** This layer has a variable number of neurons (determined by the training process). Each neuron consists of a radial basis function centered on a point with as many dimensions as predictor variables. The spread (radius) of the RBF function may be different for each dimension. The centers and spreads are determined by training. When presented with the x vector of input values from the input layer, a hidden neuron computes the Euclidean distance of the test case from the neuron’s center point and then applies the RBF kernel function to this distance using the spread values. The resulting value is passed to the summation layer.
* **Summation layer:** The value coming out of a neuron in the hidden layer is multiplied by a weight associated with the neuron and adds to the weighted values of other neurons. This sum becomes the output. For classification problems, one output is produced (with a separate set of weights and summation unit) for each target category. The value output for a category is the probability that the case being evaluated has that category.

**Training**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=14)]

The following parameters are determined by the training process:

* The number of neurons in the hidden layer
* The coordinates of the center of each hidden-layer RBF function
* The radius (spread) of each RBF function in each dimension
* The weights applied to the RBF function outputs as they pass to the summation layer

Various methods have been used to train RBF networks. One approach first uses [K-means clustering](https://en.wikipedia.org/wiki/K-means_clustering) to find cluster centers which are then used as the centers for the RBF functions. However, K-means clustering is computationally intensive and it often does not generate the optimal number of centers. Another approach is to use a random subset of the training points as the centers.

DTREG uses a training algorithm that uses an evolutionary approach to determine the optimal center points and spreads for each neuron. It determines when to stop adding neurons to the network by monitoring the estimated leave-one-out (LOO) error and terminating when the LOO error begins to increase because of overfitting.

The computation of the optimal weights between the neurons in the hidden layer and the summation layer is done using ridge regression. An iterative procedure computes the optimal regularization Lambda parameter that minimizes the generalized cross-validation (GCV) error.

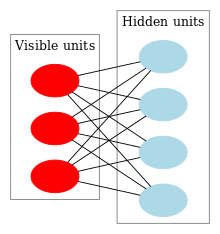
**General regression neural network**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=15)]

*Main article:*[*General regression neural network*](https://en.wikipedia.org/wiki/General_regression_neural_network)

A GRNN is an associative memory neural network that is similar to the [probabilistic neural network](https://en.wikipedia.org/wiki/Probabilistic_neural_network) but it is used for regression and approximation rather than classification.

**Deep belief network**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=16)]

*Main article:*[*Deep belief network*](https://en.wikipedia.org/wiki/Deep_belief_network)

[](https://en.wikipedia.org/wiki/File:Restricted_Boltzmann_machine.svg)

A [restricted Boltzmann machine](https://en.wikipedia.org/wiki/Restricted_Boltzmann_machine) (RBM) with fully connected visible and hidden units. Note there are no hidden-hidden or visible-visible connections.

A deep belief network (DBN) is a probabilistic, [generative model](https://en.wikipedia.org/wiki/Generative_model) made up of multiple hidden layers. It can be considered a [composition](https://en.wikipedia.org/wiki/Function_composition) of simple learning modules.[[41]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-SCHOLARDBNS-41)

A DBN can be used to generatively pre-train a deep neural network (DNN) by using the learned DBN weights as the initial DNN weights. Various discriminative algorithms can then tune these weights. This is particularly helpful when training data are limited, because poorly initialized weights can significantly hinder learning. These pre-trained weights end up in a region of the weight space that is closer to the optimal weights than random choices. This allows for both improved modeling and faster ultimate convergence.[[42]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-42)

Recurrent neural network[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=17)]

*Main article:*[*Recurrent neural network*](https://en.wikipedia.org/wiki/Recurrent_neural_network)

[Recurrent neural networks](https://en.wikipedia.org/wiki/Recurrent_neural_network) (RNN) propagate data forward, but also backwards, from later processing stages to earlier stages. RNN can be used as general sequence processors.

**Fully recurrent**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=18)]

This architecture was developed in the 1980s. Its network creates a directed connection between every pair of units. Each 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 labeled nodes, some output nodes, the rest hidden nodes.

For [supervised learning](https://en.wikipedia.org/wiki/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 each 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. The system can explicitly activate (independent of incoming signals) some 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](https://en.wikipedia.org/wiki/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](https://en.wikipedia.org/wiki/Differentiable_function). The standard method is called "[backpropagation through time](https://en.wikipedia.org/wiki/Backpropagation_through_time)" or BPTT, a generalization of back-propagation for feedforward networks.[[43]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-43)[[44]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-44) A more computationally expensive online variant is called "Real-Time Recurrent Learning" or RTRL.[[45]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-45)[[46]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-46) Unlike BPTT this algorithm is *local in time but not local in space*.[[47]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-47)[[48]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-48) An online hybrid between BPTT and RTRL with intermediate complexity exists,[[49]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-49)[[50]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-50) with variants for continuous time.[[51]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-51) 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.[[52]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-52)[[53]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-53) The [Long short-term memory](https://en.wikipedia.org/wiki/Long_short-term_memory) architecture overcomes these problems.[[54]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-lstm-54)

In [reinforcement learning](https://en.wikipedia.org/wiki/Reinforcement_learning) settings, no teacher provides target signals. Instead a [fitness function](https://en.wikipedia.org/wiki/Fitness_function) or [reward function](https://en.wikipedia.org/wiki/Reward_function) or [utility function](https://en.wikipedia.org/wiki/Utility_function) is occasionally used to evaluate performance, which influences its input stream through output units connected to actuators that affect the environment. Variants of [evolutionary computation](https://en.wikipedia.org/wiki/Evolutionary_computation) are often used to optimize the weight matrix.

**Hopfield**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=19)]

*Main article:*[*Hopfield network*](https://en.wikipedia.org/wiki/Hopfield_network)

The [Hopfield network](https://en.wikipedia.org/wiki/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. It guarantees that it will converge. If the connections are trained using [Hebbian learning](https://en.wikipedia.org/wiki/Hebbian_learning) the Hopfield network can perform as robust [content-addressable memory](https://en.wikipedia.org/wiki/Content-addressable_memory), resistant to connection alteration.

**Boltzmann machine**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=20)]

*Main article:*[*Boltzmann machine*](https://en.wikipedia.org/wiki/Boltzmann_machine)

The [Boltzmann machine](https://en.wikipedia.org/wiki/Boltzmann_machine) can be thought of as a noisy Hopfield network. 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 speeds up training for Boltzmann machines and [Products of Experts](https://en.wikipedia.org/wiki/Product_of_Experts).

**Self-organizing map**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=21)]

*Main article:*[*Self-organizing map*](https://en.wikipedia.org/wiki/Self-organizing_map)

The self-organizing map (SOM) uses [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning). A set of 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 SOM attempts to preserve these.

**Learning vector quantization**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=22)]

*Main article:*[*Learning vector quantization*](https://en.wikipedia.org/wiki/Learning_vector_quantization)

[Learning vector quantization](https://en.wikipedia.org/wiki/Learning_vector_quantization) (LVQ) can be interpreted as a neural network architecture. Prototypical representatives of the classes parameterize, together with an appropriate distance measure, in a distance-based classification scheme.

**Simple recurrent**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=23)]

Simple recurrent networks have three layers, with the addition of a set of "context units" in the input layer. These units connect from the hidden layer or the output layer with a fixed weight of one.[[55]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-55) At each time step, the input is propagated in a standard feedforward fashion, and then a backpropagation-like learning rule is applied (not performing [gradient descent](https://en.wikipedia.org/wiki/Gradient_descent)). The fixed back connections leave a copy of the previous values of the hidden units in the context units (since they propagate over the connections before the learning rule is applied).

**Reservoir computing**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=24)]

*Main article:*[*Reservoir computing*](https://en.wikipedia.org/wiki/Reservoir_computing)

Reservoir computing is a computation framework that may be viewed as an extension of [neural networks](https://en.wikipedia.org/wiki/Neural_network).[[56]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-56) Typically an input signal is fed into a fixed (random) [dynamical system](https://en.wikipedia.org/wiki/Dynamical_system) called a *reservoir* whose dynamics map the input to a higher dimension. A *readout* mechanism is trained to map the reservoir to the desired output. Training is performed only at the readout stage. [Liquid-state machines](https://en.wikipedia.org/wiki/Liquid-state_machine)[[57]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-57) are two major types of reservoir computing.[[58]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-58)

**Echo state**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=25)]

*Main article:*[*Echo state network*](https://en.wikipedia.org/wiki/Echo_state_network)

The echo state network (ESN) employs a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that are trained. ESN are good at reproducing certain time series.[[59]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-59)

**Long short-term memory**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=26)]

*Main article:*[*Long short-term memory*](https://en.wikipedia.org/wiki/Long_short-term_memory)

The [long short-term memory](https://en.wikipedia.org/wiki/Long_short-term_memory) (LSTM)[[54]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-lstm-54) avoids the [vanishing gradient problem](https://en.wikipedia.org/wiki/Vanishing_gradient_problem). It works even when with long delays between inputs and can handle signals that mix low and high frequency components. LSTM RNN outperformed other RNN and other sequence learning methods such as [HMM](https://en.wikipedia.org/wiki/Hidden_Markov_model) in applications such as language learning[[60]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-60) and connected handwriting recognition.[[61]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-61)

**Bi-directional**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=27)]

Bi-directional RNN, or BRNN, use a finite sequence to predict or label each element of a sequence based on both the past and future context of the element.[[62]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-62) 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.[[63]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-63)

**Hierarchical**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=28)]

Hierarchical RNN connects elements in various ways to decompose hierarchical behavior into useful subprograms.[[64]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-64)[[65]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-65)

**Stochastic**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=29)]

*Main article:*[*Stochastic neural network*](https://en.wikipedia.org/wiki/Stochastic_neural_network)

A stochastic neural network introduces random variations into the network. Such random variations can be viewed as a form of [statistical sampling](https://en.wikipedia.org/wiki/Statistical_sampling), such as [Monte Carlo sampling](https://en.wikipedia.org/wiki/Monte_Carlo_sampling).

**Genetic Scale**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=30)]

A RNN (often a LSTM) where a series is decomposed into a number of scales where every scale informs the primary length between two consecutive points. A first order scale consists of a normal RNN, a second order consists of all points separated by two indices and so on. The Nth order RNN connects the first and last node. The outputs from all the various scales are treated as a Committee of Machines and the associated scores are used genetically for the next iteration.

Modular[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=31)]

*Main article:*[*Modular neural network*](https://en.wikipedia.org/wiki/Modular_neural_network)

Biological studies have shown that the human brain operates as a collection of small networks. This realization gave birth to the concept of [modular neural networks](https://en.wikipedia.org/wiki/Modular_neural_networks), in which several small networks cooperate or compete to solve problems.

**Committee of machines**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=32)]

*Main article:*[*Committee machine*](https://en.wikipedia.org/wiki/Committee_machine)

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 than individual networks. Because neural networks suffer from local minima, starting with the same architecture and training but using randomly different initial weights often gives vastly different results.[[*citation needed*](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed)] A CoM tends to stabilize the result.

The CoM is similar to the general [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [*bagging*](https://en.wikipedia.org/wiki/Bootstrap_Aggregating) method, except that the necessary variety of machines in the committee is obtained by training from different starting weights rather than training on different randomly selected subsets of the training data.

**Associative**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=33)]

The associative neural network (ASNN) is an extension of committee of machines that combines multiple feedforward neural networks and the k-nearest neighbor technique. 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-learns) without retraining. 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.[[66]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-66)

Physical[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=34)]

*Main article:*[*Physical neural network*](https://en.wikipedia.org/wiki/Physical_neural_network)

A physical neural network includes electrically adjustable resistance material to simulate artificial synapses. Examples include the [ADALINE](https://en.wikipedia.org/wiki/ADALINE) [memristor](https://en.wikipedia.org/wiki/Memristor)-based neural network.[[67]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-67) An [optical neural network](https://en.wikipedia.org/wiki/Optical_neural_network) is a physical implementation of an [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) with [optical components](https://en.wikipedia.org/wiki/Photonics).

Other types[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=35)]

**Instantaneously trained**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=36)]

[Instantaneously trained neural networks](https://en.wikipedia.org/wiki/Instantaneously_trained_neural_networks) (ITNN) 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 exist.

**Spiking**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=37)]

[Spiking neural networks](https://en.wikipedia.org/wiki/Spiking_neural_network) (SNN) explicitly consider the timing of inputs. The network input and output are usually represented as a series of spikes (delta function or more complex shapes). SNN can process information in the [time domain](https://en.wikipedia.org/wiki/Time_domain) (signals that vary over time). They are often implemented as recurrent networks. SNN are also a form of [pulse computer](https://en.wikipedia.org/wiki/Pulse_computer).[[68]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-68)

Spiking neural networks with axonal conduction delays exhibit polychronization, and hence could have a very large memory capacity.[[69]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-pmid16378515-69)

SNN 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.

**Regulatory feedback**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=38)]

A regulatory feedback network makes inferences using [negative feedback](https://en.wikipedia.org/wiki/Negative_feedback).[[70]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-shedding2-70) The feedback is used to find the optimal activation of units. It is most similar to a [non-parametric method](https://en.wikipedia.org/wiki/Non-parametric_methods) but is different from K-nearest neighbor in that it mathematically emulates feedforward networks.

**Neocognitron**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=39)]

The [neocognitron](https://en.wikipedia.org/wiki/Neocognitron" \o "Neocognitron) is a hierarchical, multilayered network that was modeled after the [visual cortex](https://en.wikipedia.org/wiki/Visual_cortex). It uses multiple types of units, (originally two, called [simple](https://en.wikipedia.org/wiki/Simple_cell) and [complex](https://en.wikipedia.org/wiki/Complex_cell) cells), as a cascading model for use in pattern recognition tasks.[[71]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-71)[[72]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-72)[[73]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-73) Local features are extracted by S-cells whose deformation is tolerated by C-cells. Local features in the input are integrated gradually and classified at higher layers.[[74]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-74) Among the various kinds of neocognitron[[75]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks" \l "cite_note-75) are systems that can detect multiple patterns in the same input by using back propagation to achieve [selective attention](https://en.wikipedia.org/wiki/Selective_attention).[[76]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-76) It has been used for [pattern recognition](https://en.wikipedia.org/wiki/Pattern_recognition) tasks and inspired [convolutional neural networks](https://en.wikipedia.org/wiki/Convolutional_neural_network).[[77]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-77)

**Compound hierarchical-deep models**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=40)]

Compound hierarchical-deep models compose deep networks with non-parametric [Bayesian models](https://en.wikipedia.org/wiki/Bayesian_network). [Features](https://en.wikipedia.org/wiki/Feature_(machine_learning)) can be learned using deep architectures such as DBNs,[[78]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-hinton2006-78) deep Boltzmann machines (DBM),[[79]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref3-79) deep auto encoders,[[80]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref15-80) convolutional variants,[[81]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref39-81)[[82]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref40-82) ssRBMs,[[83]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref32-83) deep coding networks,[[84]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref41-84) DBNs with sparse feature learning,[[85]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref42-85) RNNs,[[86]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref43-86) conditional DBNs,[[87]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref44-87) de-noising auto encoders.[[88]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref45-88) This provides a better representation, allowing faster learning and more accurate classification with high-dimensional data. However, these architectures are poor at learning novel classes with few examples, because all network units are involved in representing the input (a *distributed representation*) and must be adjusted together (high [degree of freedom](https://en.wikipedia.org/wiki/Degree_of_freedom)). Limiting the degree of freedom reduces the number of parameters to learn, facilitating learning of new classes from few examples. [*Hierarchical Bayesian (HB)* models](https://en.wikipedia.org/wiki/Hierarchical_Bayesian_model) allow learning from few examples, for example[[89]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref34-89)[[90]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref37-90)[[91]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref46-91)[[92]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref47-92)[[93]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref48-93) for computer vision, [statistics](https://en.wikipedia.org/wiki/Statistics) and cognitive science.

Compound HD architectures aim to integrate characteristics of both HB and deep networks. The compound HDP-DBM architecture is a [*hierarchical Dirichlet process*](https://en.wikipedia.org/wiki/Hierarchical_Dirichlet_process)*(HDP)* as a hierarchical model, incorporating DBM architecture. It is a full [generative model](https://en.wikipedia.org/wiki/Generative_model), generalized from abstract concepts flowing through the model layers, which is able to synthesize new examples in novel classes that look "reasonably" natural. All the levels are learned jointly by maximizing a joint [log-probability](https://en.wikipedia.org/wiki/Log_probability) [score](https://en.wikipedia.org/wiki/Score_(statistics)).[[94]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref38-94)

In a DBM with three hidden layers, the probability of a visible input ''***ν''*** is:

{\displaystyle p({\boldsymbol {\nu }},\psi )={\frac {1}{Z}}\sum \_{h}\exp \left(\sum \_{ij}W\_{ij}^{(1)}\nu \_{i}h\_{j}^{1}+\sum \_{j\ell }W\_{j\ell }^{(2)}h\_{j}^{1}h\_{\ell }^{2}+\sum \_{\ell m}W\_{\ell m}^{(3)}h\_{\ell }^{2}h\_{m}^{3}\right),}

where {\displaystyle {\boldsymbol {h}}=\{{\boldsymbol {h}}^{(1)},{\boldsymbol {h}}^{(2)},{\boldsymbol {h}}^{(3)}\}} is the set of hidden units, and {\displaystyle \psi =\{{\boldsymbol {W}}^{(1)},{\boldsymbol {W}}^{(2)},{\boldsymbol {W}}^{(3)}\}} are the model parameters, representing visible-hidden and hidden-hidden symmetric interaction terms.

A learned DBM model is an undirected model that defines the joint distribution {\displaystyle P(\nu ,h^{1},h^{2},h^{3})}. One way to express what has been learned is the [conditional model](https://en.wikipedia.org/wiki/Discriminative_model) {\displaystyle P(\nu ,h^{1},h^{2}\mid h^{3})} and a prior term {\displaystyle P(h^{3})}.

Here {\displaystyle P(\nu ,h^{1},h^{2}\mid h^{3})} represents a conditional DBM model, which can be viewed as a two-layer DBM but with bias terms given by the states of {\displaystyle h^{3}}:

{\displaystyle P(\nu ,h^{1},h^{2}\mid h^{3})={\frac {1}{Z(\psi ,h^{3})}}\exp \left(\sum \_{ij}W\_{ij}^{(1)}\nu \_{i}h\_{j}^{1}+\sum \_{j\ell }W\_{j\ell }^{(2)}h\_{j}^{1}h\_{\ell }^{2}+\sum \_{\ell m}W\_{\ell m}^{(3)}h\_{\ell }^{2}h\_{m}^{3}\right).}

**Deep predictive coding networks**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=41)]

A deep predictive coding network (DPCN) is a [predictive](https://en.wikipedia.org/wiki/Predictive_modelling) coding scheme that uses top-down information to empirically adjust the priors needed for a bottom-up [inference](https://en.wikipedia.org/wiki/Inference) procedure by means of a deep, locally connected, [generative model](https://en.wikipedia.org/wiki/Generative_model). This works by extracting sparse [features](https://en.wikipedia.org/wiki/Feature_(machine_learning)) from time-varying observations using a linear dynamical model. Then, a pooling strategy is used to learn invariant feature representations. These units compose to form a deep architecture and are trained by [greedy](https://en.wikipedia.org/wiki/Greedy_algorithm) layer-wise [unsupervised learning](https://en.wikipedia.org/wiki/Unsupervised_learning). The layers constitute a kind of [Markov chain](https://en.wikipedia.org/wiki/Markov_chain) such that the states at any layer depend only on the preceding and succeeding layers.

DPCNs predict the representation of the layer, by using a top-down approach using the information in upper layer and temporal dependencies from previous states.[[95]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref56-95)

DPCNs can be extended to form a [convolutional network](https://en.wikipedia.org/wiki/Convolutional_neural_network).[[95]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref56-95)

**Multilayer kernel machine**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=42)]

Multilayer kernel machines (MKM) are a way of learning highly nonlinear functions by iterative application of weakly nonlinear kernels. They use [kernel principal component analysis](https://en.wikipedia.org/wiki/Kernel_principal_component_analysis) (KPCA),[[96]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref60-96) as a method for the [unsupervised](https://en.wikipedia.org/wiki/Unsupervised_learning) greedy layer-wise pre-training step of deep learning.[[97]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ref59-97)

Layer {\displaystyle \ell +1} learns the representation of the previous layer {\displaystyle \ell }, extracting the {\displaystyle n\_{l}} [principal component](https://en.wikipedia.org/wiki/Principal_component_analysis) (PC) of the projection layer {\displaystyle l} output in the feature domain induced by the kernel. To reduce the [dimensionaliity](https://en.wikipedia.org/wiki/Dimensionality_reduction" \o "Dimensionality reduction) of the updated representation in each layer, a [supervised strategy](https://en.wikipedia.org/wiki/Supervised_learning) selects the best informative features among features extracted by KPCA. The process is:

* rank the {\displaystyle n\_{\ell }} features according to their [mutual information](https://en.wikipedia.org/wiki/Mutual_information) with the class labels;
* for different values of *K* and {\displaystyle m\_{\ell }\in \{1,\ldots ,n\_{\ell }\}}, compute the classification error rate of a [*K-nearest neighbor*](https://en.wikipedia.org/wiki/K-nearest_neighbor)*(K-NN)* classifier using only the {\displaystyle m\_{l}} most informative features on a [validation set](https://en.wikipedia.org/wiki/Validation_set);
* the value of {\displaystyle m\_{\ell }} with which the classifier has reached the lowest error rate determines the number of features to retain.

Some drawbacks accompany the KPCA method for MKMs.

A more straightforward way to use kernel machines for deep learning was developed for spoken language understanding.[[98]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-98) The main idea is to use a kernel machine to approximate a shallow neural net with an infinite number of hidden units, then use [stacking](https://en.wikipedia.org/wiki/Deep_learning#Deep_stacking_networks) to splice the output of the kernel machine and the raw input in building the next, higher level of the kernel machine. The number of levels in the deep convex network is a hyper-parameter of the overall system, to be determined by cross validation.

Dynamic[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=43)]

Dynamic neural networks address nonlinear multivariate behaviour and include (learning of) time-dependent behaviour, such as transient phenomena and delay effects. Techniques to estimate a system process from observed data fall under the general category of system identification.

**Cascading**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=44)]

Cascade correlation is an architecture and [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) [algorithm](https://en.wikipedia.org/wiki/Algorithm). Instead of just adjusting the weights in a network of fixed topology,[[99]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-99) 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: It learns quickly, determines its own size and topology, retains the structures it has built even if the training set changes and requires no [backpropagation](https://en.wikipedia.org/wiki/Back-propagation).

**Neuro-fuzzy**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=45)]

A neuro-fuzzy network is a [fuzzy](https://en.wikipedia.org/wiki/Fuzzy_logic) [inference system](https://en.wikipedia.org/wiki/Inference_system) in the body of an artificial neural network. Depending on the FIS type, several layers 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**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=46)]

*Main article:*[*Compositional pattern-producing network*](https://en.wikipedia.org/wiki/Compositional_pattern-producing_network)

Compositional pattern-producing networks (CPPNs) are a variation of artificial neural networks which differ in their set of [activation functions](https://en.wikipedia.org/wiki/Activation_function) and how they are applied. While typical artificial neural networks often contain only [sigmoid functions](https://en.wikipedia.org/wiki/Sigmoid_function) (and sometimes [Gaussian functions](https://en.wikipedia.org/wiki/Gaussian_function)), CPPNs can include both types of functions and many others. Furthermore, unlike typical artificial neural networks, 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.

Memory networks[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=47)]

Memory networks[[100]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Weston,_Jason_14102-100)[[101]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-101) incorporate [long-term memory](https://en.wikipedia.org/wiki/Long-term_memory). The long-term memory can be read and written to, with the goal of using it for prediction. These models have been applied in the context of [question answering](https://en.wikipedia.org/wiki/Question_answering) (QA) where the long-term memory effectively acts as a (dynamic) knowledge base and the output is a textual response.[[102]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-102)

In [sparse distributed memory](https://en.wikipedia.org/wiki/Sparse_distributed_memory) or [hierarchical temporal memory](https://en.wikipedia.org/wiki/Hierarchical_temporal_memory), the patterns encoded by neural networks are used as addresses for [content-addressable memory](https://en.wikipedia.org/wiki/Content-addressable_memory), with "neurons" essentially serving as address encoders and [decoders](https://en.wikipedia.org/wiki/Binary_decoder). However, the early controllers of such memories were not differentiable.[[103]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Hinton,_Geoffrey_E_19842-103)

**One-shot associative memory**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=48)]

This type of network can add new patterns without 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.[[104]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-104) The network offers real-time pattern recognition and high scalability; this requires parallel processing and is thus best suited for platforms such as [wireless sensor networks](https://en.wikipedia.org/wiki/Wireless_sensor_network), [grid computing](https://en.wikipedia.org/wiki/Grid_computing), and [GPGPUs](https://en.wikipedia.org/wiki/GPGPU).

**Hierarchical temporal memory**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=49)]

*Main article:*[*Hierarchical temporal memory*](https://en.wikipedia.org/wiki/Hierarchical_temporal_memory)

Hierarchical temporal memory (HTM) models some of the structural and [algorithmic](https://en.wikipedia.org/wiki/Algorithm) properties of the [neocortex](https://en.wikipedia.org/wiki/Neocortex). HTM is a [biomimetic](https://en.wikipedia.org/wiki/Bionics) model based on [memory-prediction](https://en.wikipedia.org/wiki/Memory-prediction_framework) theory. 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.

HTM combines existing ideas to mimic the neocortex with a simple design that provides many capabilities. HTM combines and extends approaches used in [Bayesian networks](https://en.wikipedia.org/wiki/Bayesian_networks), spatial and temporal clustering algorithms, while using a tree-shaped hierarchy of nodes that is common in [neural networks](https://en.wikipedia.org/wiki/Neural_networks).

**Holographic associative memory**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=50)]

*Main article:*[*Holographic associative memory*](https://en.wikipedia.org/wiki/Holographic_associative_memory)

Holographic Associative Memory (HAM) is an analog, correlation-based, associative, stimulus-response system. Information is mapped onto the phase orientation of complex numbers. The memory is effective for [associative](https://en.wikipedia.org/wiki/Association_(psychology)) [memory](https://en.wikipedia.org/wiki/Memory) tasks, generalization and pattern recognition with changeable attention. Dynamic search localization is central to biological memory. In visual perception, humans focus on specific objects in a pattern. Humans can change focus from object to object without learning. HAM can mimic this ability by creating explicit representations for focus. It uses a bi-modal representation of pattern and a hologram-like complex spherical weight state-space. HAMs are useful for optical realization because the underlying hyper-spherical computations can be implemented with optical computation.[[105]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-105)

**LSTM-related differentiable memory structures**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=51)]

Apart from [long short-term memory](https://en.wikipedia.org/wiki/Long_short-term_memory) (LSTM), other approaches also added differentiable memory to recurrent functions. For example:

* Differentiable push and pop actions for alternative memory networks called neural stack machines[[106]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-S._Das,_C.L._Giles_p._79-106)[[107]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Mozer,_M._C._1993_pp._863-870-107)
* Memory networks where the control network's external differentiable storage is in the fast weights of another network[[108]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-ReferenceC-108)
* LSTM forget gates[[109]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-F._Gers,_N._Schraudolph_2002-109)
* Self-referential RNNs with special output units for addressing and rapidly manipulating the RNN's own weights in differentiable fashion (internal storage)[[110]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-J._Schmidhuber_pages_191-195-110)[[111]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Hochreiter,_Sepp_2001-111)
* Learning to transduce with unbounded memory[[112]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Grefenstette,_Edward_1506-112)

**Neural Turing machines**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=52)]

*Main article:*[*Neural Turing machine*](https://en.wikipedia.org/wiki/Neural_Turing_machine)

Neural Turing machines[[113]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Graves,_Alex_14102-113) couple LSTM networks to external memory resources, with which they can interact by attentional processes. The combined system is analogous to a [Turing machine](https://en.wikipedia.org/wiki/Turing_machine) but is differentiable end-to-end, allowing it to be efficiently trained by [gradient descent](https://en.wikipedia.org/wiki/Gradient_descent). Preliminary results demonstrate that neural Turing machines can infer simple algorithms such as copying, sorting and associative recall from input and output examples.

[Differentiable neural computers](https://en.wikipedia.org/wiki/Differentiable_neural_computer) (DNC) are an NTM extension. They out-performed Neural turing machines, [long short-term memory](https://en.wikipedia.org/wiki/Long_short-term_memory) systems and memory networks on sequence-processing tasks.[[114]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-:02-114)[[115]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-115)[[116]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-116)[[117]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-117)[[118]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-118)

**Semantic hashing**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=53)]

Approaches that represent previous experiences directly and [use a similar experience to form a local model](https://en.wikipedia.org/wiki/Instance-based_learning) are often called [nearest neighbour](https://en.wikipedia.org/wiki/K-nearest_neighbor_algorithm) or [k-nearest neighbors](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm) methods.[[119]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-119) Deep learning is useful in semantic hashing[[120]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-120) where a deep [graphical model](https://en.wikipedia.org/wiki/Graphical_model) the word-count vectors[[121]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-Le_2014-121) obtained from a large set of documents.[[*clarification needed*](https://en.wikipedia.org/wiki/Wikipedia:Please_clarify)] Documents are mapped to memory addresses in such a way that semantically similar documents are located at nearby addresses. Documents similar to a query document can then be found by accessing all the addresses that differ by only a few bits from the address of the query document. Unlike [sparse distributed memory](https://en.wikipedia.org/wiki/Sparse_distributed_memory) that operates on 1000-bit addresses, semantic hashing works on 32 or 64-bit addresses found in a conventional computer architecture.

**Pointer networks**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=54)]

Deep neural networks can be potentially improved by deepening and parameter reduction, while maintaining trainability. While training extremely deep (e.g., 1 million layers) neural networks might not be practical, [CPU](https://en.wikipedia.org/wiki/CPU)-like architectures such as pointer networks[[122]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-122) and neural random-access machines[[123]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-123) overcome this limitation by using external [random-access memory](https://en.wikipedia.org/wiki/Random-access_memory) and other components that typically belong to a [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture) such as [registers](https://en.wikipedia.org/wiki/Processor_register), [ALU](https://en.wikipedia.org/wiki/Arithmetic_logic_unit) and [pointers](https://en.wikipedia.org/wiki/Pointer_(computer_programming)). Such systems operate on [probability distribution](https://en.wikipedia.org/wiki/Probability_distribution) vectors stored in memory cells and registers. Thus, the model is fully differentiable and trains end-to-end. The key characteristic of these models is that their depth, the size of their short-term memory, and the number of parameters can be altered independently.

Hybrids[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=55)]

**Encoder–decoder networks**[[edit](https://en.wikipedia.org/w/index.php?title=Types_of_artificial_neural_networks&action=edit&section=56)]

Encoder–decoder frameworks are based on neural networks that map highly [structured](https://en.wikipedia.org/wiki/Structured_prediction) input to highly structured output. The approach arose in the context of [machine translation](https://en.wikipedia.org/wiki/Machine_translation),[[124]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-124)[[125]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-125)[[126]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-126) where the input and output are written sentences in two natural languages. In that work, an LSTM RNN or CNN was used as an encoder to summarize a source sentence, and the summary was decoded using a conditional RNN [language model](https://en.wikipedia.org/wiki/Language_model) to produce the translation.[[127]](https://en.wikipedia.org/wiki/Types_of_artificial_neural_networks#cite_note-127) These systems share building blocks: gated RNNs and CNNs and trained attention mechanisms.