Artificial neural networks (ANNs) are widely used to model low-level neural activities and high-level cognitive functions. In this article, we review the applications of statistical inference for learning in ANNs. Statistical inference provides an objective way to derive learning algorithms both for training and for evaluation of the performance of trained ANNs. Solutions to the over-fitting problem by model-selection methods, based on either conventional statistical approaches or on a Bayesian approach, are discussed. The use of supervised and unsupervised learning algorithms for ANNs are reviewed.

Training a multilayer ANN by supervised learning is equivalent to nonlinear regression. The ensemble methods, bagging and arching, described here, can be applied to combine ANNs to form a new predictor with improved performance. Unsupervised learning algorithms that are derived either by the Hebbian law for bottom-up self-organization, or by global objective functions for top-down self-organization are also discussed.

Although the brain is an extremely large and complex system, from the point of view of its organization the hierarchy of the brain can be divided into eight levels: behavior, cortex, neural circuit, neuron, microcircuit, synapse, membrane and molecule. With advanced invasive and noninvasive measurement techniques, the brain can be observed at all these levels and a huge amount of data have been collected. Neural computational theories have been developed to account for complex brain functions based on the accumulated data.

Neural computational theories comprise neural models, neural dynamics and learning theories. Mathematical modeling has been applied to each level in the hierarchy of the brain. The brain can be considered at three functional levels: (1) a cognitive-function level related to behavior and cortex; (2) a neural-activity level related to neural circuit, neuron, microcircuit and synapse; and (3) a subneural level related to the membrane and molecule. In this article, we only consider the first and second functional levels.

To focus on the information processing principles of the brain, we must simplify the neurons and synapses in real neural systems. ANNs are simplified mathematical models for neural systems formed by massively interconnected computational units running in parallel. We discuss the applications of ANNs at the neural-activity level and the cognitive-function level.

Applications of ANNs can model brain functions at the level of either neural activity or cognitive function. The meanings of the units and connections in an ANN are different at these two levels.

At the neural-activity level, the units and connections model the nerve cells and the synapses between the neurons, respectively. This gives a correspondence between an ANN and a neural system. One successful application of ANNs at the activity level is Zipser and Andersen's model, which is a three-layer feed-forward network, trained by back-propagation to perform the vector addition of the retinal and eye positions. After training, the simulated retinal receptive fields and the eye position responses of the hidden units in the trained network closely resembled those found in the posterior parietal cortex of the primate brain, where the absolute spatial location (the position of the object in space, which does not depend on the head direction) is computed.

At the cognitive-function level, ANNs are connectionist models for cognitive processing. The units and connections in the connectionist models are used to represent certain cognitive states or hypotheses, and constraints among these states or hypotheses, respectively. It has been widely believed, and demonstrated by connectionists, that some cognitive functions emerge from the interactions among a large number of computational units. Different cognitive tasks, such as memory retrieval, category formation, speech perception, language acquisition and object recognition, have been modeled by ANNs (Refs 3-5). Some examples are the word pronunciation model, the neural arithmetic model, the English text-to-speech system, and the TD-Gammon model, which is one of the best backgammon players in the world.
Statistics and ANNs
McClendon summarized five principles to characterize the information processing in connectionist models: principles of graded activation, gradual propagation, interactive processing, mutual competition, and intrinsic variability. In ANNs, the first principle is realized by a linear combination of inputs and sigmoid activation function, the second by finite impulse response (FIR) filters with exponentially decaying impulse response functions as models for synapses, the third by a bi-directional structure, the fourth by lateral inhibition, and the fifth by noisy or probabilistic units. The factor of intrinsic variability plays an important role in human information processing and it is intrinsic variability that is the main difference between the brain and digital computers. von Neumann once hinted at the idea of building a brain-like computer based on statistical principles. It is a reasonable hypothesis that the brain incorporates intrinsic variability naturally in its structure so that it can operate in a stochastic environment, receiving noisy and ambiguous inputs. McClendon’s work provided some new directions for neural-network research. As a basic hypothesis for connectionist models, the intrinsic variability principle is appealing, especially to statisticians because it allows them to build stochastic neural-network models and to apply statistical inference to these models. 

Two essential parts of the modern neural-network theory are stochastic models for ANNs and learning algorithms based on statistical inference. White and Ripley give some statistician’s perspectives about ANNs and treat them rigorously using a statistical framework. Amari reviews some important issues about learning and statistical inference and the applications of information geometry in ANNs. In this article, we further review these issues as well as some other issues not covered previously by Amari.

Stochastic models
Many stochastic neural-network models have been proposed in the literature. A good neural-network model should be concise in structure with powerful approximation capability, and be tractable by statistical inference methods. A simple, but useful, model for a stochastic perceptron is:

\[ y = g(x^Tw + b) + \text{noise} \]

where \( x \) is the input, \( y \) the output, and \( g(x^Tw + b) \) a nonlinear activation function parameterized by \( \Theta \). For example, \( g(x^Tw + b) = f(a^x + b) \) for \( \Theta = (a,b) \) or \( g(x^Tw + b) = f(x^T\theta) \) for \( \Theta = (\theta) \) where \( f(a) \) is a single variable function, \( b \) bias, \( a \) a vector linearly combining the inputs, \( \lambda \) a matrix linearly combining the second order inputs, and \( f \) denotes the vector transpose. The function \( f \) should be simple for calculation, but not too simple so that the network can approximate arbitrary nonlinear functions. Some common choices for \( f \) are sigmoid, bell shape and Mexican hat functions. Other stochastic models are random noise, Boltzmann machines, stochastic Helmholtz machine, and the hierarchical mixture of experts model.

Learning algorithms based on statistical inference
Why should we apply statistical inference in the learning of ANNs? A brief answer to this question is that statistical inference will guide us to derive learning algorithms and to analyze their performance in a more systematic way. ANNs have input nodes for taking data, hidden nodes for the internal representation and output nodes for displaying patterns. The goal of learning is to find an input-output relation with a prediction error as small as possible. A learning algorithm is called supervised if the desired outputs are known. Otherwise, it is unsupervised.

Supervised learning
Multilayer neural networks are chosen as useful connectionist models because of their universal approximation capability. A network for knowledge representation can be trained from examples without using verbal rules and hard-wiring. Three typical approaches to train a multilayer neural network are (1) the optimization approach, (2) the conventional statistical approach and (3) the Bayesian approach.

Optimization approach
Training a multilayer neural network is often formulated as an optimization problem. The learning algorithms based on gradient descent are enriched by some optimization techniques such as the momentum and the Newton-Raphson method. However, because the cost functions are subjectively chosen, it is very difficult to address problems such as the efficiency and the accuracy of the learning algorithms within the optimization framework. A further problem is that the trained network might over-fit the data. Thus, when the network architecture is more complex than required, the algorithm might decrease the training error on the training examples, but increase the generalization error on the novel examples that are not shown to the network during training. As a result, the learning process can be driven by the training examples to a wrong solution. Therefore, it is crucial to select a correct model in the learning process based on the performance of the trained network. To examine the performance of the trained network, some examples should be left aside for testing, not training.

In the optimization approach, the model selection is done by trial and error. This is time consuming and the optimal architecture might not be found. We now discuss some model selection methods based on statistical inference.

Conventional statistical approach
Many papers about statistical inference learning have been published in the past three decades. A key link between neural networks and statistics is nonlinear regression. Through this link, many statistical inference methods can be applied to neural networks. Perhaps the earliest work on statistical inference learning was that carried out in 1967 by Amari, in which the error-correction-adjustment method based on the stochastic-gradient-descent method was proposed to train linear or nonlinear classifiers, including single-layer and multilayer neural networks. In the 1970s and 1980s, this method was rediscovered and refined, to become the well-known error-back-propagation method.

Here, we focus on the maximum likelihood method and model selection by minimizing the generalization error. Although other learning algorithms, such as reinforcement learning and the EM-algorithm, are also closely
Box 1. Maximum likelihood method for training ANNs

Consider a $n$-to-$1$ multilayer perceptron:

$$\mathbf{x} = a(p W x + \mathbf{b} + \mathbf{e})$$

where $\mathbf{x}$ denotes the transpose, the entries in the vector $\mathbf{a}$ and the matrix $W$ are weights, the entries in the vector $\mathbf{b}$ are thresholds, $p$ is a differentiable activation function for each hidden neuron, and $\mathbf{e}$ is an additive noise with a probability density function (pdf) $p(e)$. The conditional pdf of $\mathbf{x}$ given $\mathbf{a}$ is $p(x | \mathbf{a}, \mathbf{W}, \mathbf{b} + \mathbf{e}) = q(x)$. The joint pdf of the input and output is

$$p(x, \mathbf{a}, \mathbf{W}, \mathbf{b}, \mathbf{e}) = \prod_i p(x_i | \mathbf{a}, \mathbf{W}, \mathbf{b} + \mathbf{e})$$

or equivalently to minimize a loss function defined by

$$L(D; \mathbf{a}, \mathbf{W}, \mathbf{b}) = -\sum_i \log p(x_i | \mathbf{a}, \mathbf{W}, \mathbf{b} + \mathbf{e})$$

Since $p(x)$ does not depend on $\mathbf{b}$, minimizing $L(D; \mathbf{a}, \mathbf{W}, \mathbf{b})$ is equivalent to minimizing the following loss function

$$\ell(\mathbf{a}, \mathbf{b}; \mathbf{W}) = -\sum_i \log p(x_i | \mathbf{a}, \mathbf{W}, \mathbf{b} + \mathbf{e})$$

With different assumptions for $q(x)$, we have different ways to measure training error. For example, when $q(x)$ is Gaussian,

$$q(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} x^T x}$$

$$\ell(\mathbf{a}, \mathbf{b}; \mathbf{W}) = \frac{1}{2} \sum_i (x_i - a(p(W x_i + b) + e))^2$$

when $x$ is symmetrically exponential, i.e. $q(x) = \frac{1}{z} e^{-\frac{1}{2} x}$, $\ell(\mathbf{a}, \mathbf{b}; \mathbf{W})$ is an absolute error:

$$\ell(\mathbf{a}, \mathbf{b}; \mathbf{W}) = \frac{1}{2} \sum_i (|x_i - a(p(W x_i + b) + e)|)$$

related to statistical inference learning, they are not reviewed here because of space limitations.

From the statistical point of view, training a network is equivalent to estimating its weight parameters by the maximum likelihood method (see Box 1). The following on-line algorithm is usually used to compute the maximum likelihood estimate:

$$\theta_{\text{ML}} = \arg \max_{\theta} \frac{1}{N} \sum_i q(x_i | \theta)$$

A statistically more efficient algorithm is the method of scoring:

$$\theta_{\text{ML}} = \arg \max_{\theta} \frac{1}{N} \sum_i L(D; \theta)$$

where $G(\theta)$ is the Fisher information matrix defined by

$$G(\theta) = \frac{1}{N} \sum_i \frac{1}{q(x_i | \theta)} \frac{\partial^2}{\partial \theta \partial \theta^T} q(x_i | \theta)$$

This is one of the natural-gradient ascent–descent-type algorithms based on Amari’s information geometry theory. It has been shown by Amari12 that natural-gradient learning is Fisher efficient. If $\theta$ is updated by the method of scoring, the asymptotic variance of $\theta$ is

$$\ell(\mathbf{a}, \mathbf{b}; \mathbf{W}) = \frac{1}{2} \sum_i (|x_i - a(p(W x_i + b) + e)|)$$

Although the natural-gradient-descent algorithm is statistically efficient, it is computationally expensive. The bottleneck is the computation of the natural gradient. For a $n$-to-$1$ multilayer network, assuming the input dimensionality, $n$, is much larger than the number of hidden neurons, $m$, and $y$, Yang and Amari12 described a new scheme to represent the Fisher information matrix and a fast algorithm to compute the natural gradient. The complexity of this fast algorithm for computing the natural gradient is $O(n^3)$ flops (a flop is a floating point operation; add or multiply) while a direct method, namely matrix inversion and matrix-vector multiplication, needs $O(n^4)$ flops.

Neural networks trained by the maximum likelihood method are usually statistically efficient and asymptotically unbiased when the model is correct. However, when the model is wrong, the trained networks often have the over-fitting problem. 

Moody26 formulated a stochastic model for a multilayer perceptron and introduced a regularization term in the cost function to penalize a complex network architecture. He found a second-order approximation of the generalization error as a function of the network complexity and the regularization parameter. The estimated generalization error is useful for selecting an optimal network structure. Also addressing the over-fitting problem, Murata et al.13 generalized Akaike’s information criterion and proposed an NIC (network information criterion) under a general loss function with a regularization term. A common theme in these two approaches is to minimize the generalization error by regularization, and this is the theoretical basis for the techniques used in practice, such as early stopping, growing and pruning networks. The bias-variance tradeoff26 and bias-variance-covariance tradeoff28 characterize the overfitting problem from a different perspective. In the light of the bias-variance trade-off, the regularization method balances bias and variance in order to decrease the generalization error.

A well-known theoretical framework for neural-network learning is Valiant’s probably approximately correct (PAC) learning model29 for approximately correct learning. The Vapnik–Chervonenkis (VC) dimension theory was developed to analyze the sample complexity (the required
number of training examples) and the computation complexity of PAC learning. The VC-dimension theory also analyses the difference between the training error and the generalization error in PAC learning. Most of the work on the PAC learning and the VC-dimension models is for binary functions only. Haussler26 extended the PAC model and the VC-dimension theory to more general function classes. The pseudodimension he introduced plays a similar role to the VC-dimension in the case of binary function classes. Further results on the extended PAC model and the pseudodimension are given by Maass33.

Bayesian approach

From a Bayesian point of view, it is better to assign some preference to unknown parameters to include some prior knowledge about the model.

Assume \(q \sim \text{prior} \) with a prior \(p(q)\). By the Bayesian formula, the posterior probability is given by:

\[
P(q|D) = \frac{p(D|q)p(q)}{\int p(D|q)p(q)\,dq}
\]

The Bayesian approach is to maximize the posterior instead of the likelihood function. When the prior is Gaussian, maximizing the posterior is equivalent to minimizing a cost function defined by:

\[
I(q) = -\int \log(p(x|q))\,dx + D(q||p)\]

In particular, when the noise, \(e\), is Gaussian, \(c(\theta) = \frac{1}{2\sigma^2} \int (e(\theta; x) - e_x)^2\,dx\)

Here, the regularization term is naturally included in the cost function to deal with the over-fitting problem. \(\alpha\) and \(\sigma\), are hyperparameters that, in practice, are usually chosen by trial and error. There are several rational ways to choose the hyperparameters. MacKay34 proposed a Bayesian framework for the back-propagation method and gave a Bayesian approach to choose the hyperparameters. The idea is to optimize alternately the maximum posterior estimate of \(\theta\) and the hyperparameters, by maximizing an approximated posterior probability of the hyperparameters. Amari and Murata35 gave both Bayesian and non-Bayesian treatments to find the optimal hyperparameter \(\theta\) by maximizing the posterior probability and minimizing the generalization error, respectively. Other papers on Bayesian learning include those by Neal36, and Bishop and Quarz37.

An important concept in the Bayesian approach is the predictive distribution defined by

\[
p(x|D) = \int p(x|\theta)p(\theta|D)\,d\theta
\]

where \(D = \{(x_1, y_1), \ldots, (x_N, y_N)\}\) is a training set.

If the posterior \(p(\theta|D)\) has several maxima \(\bar{\theta}_1, \ldots, \bar{\theta}_M\), and it is sharply peaked around these maxima, then the predictive distribution

\[
p(x|D) \approx \sum_{\bar{\theta}_i} c_i p(x|\bar{\theta}_i)
\]

where \(c_i \Rightarrow p(\bar{\theta}_i)\). In implementation, when the dimensional \(p(x|\theta)\) is modeled by a neural network with the parameter \(\theta\), the predictive distribution is modeled by a committee machine weighted by \(c_i\).

The conventional statistical and the Bayesian approaches both have advantages and drawbacks. A general superiority of one approach over another does not exist. However, in the context of model selection for training neural networks, Amari and Murata35 have shown that the conventional approach is asymptotically better than the Bayesian approach in terms of reducing the generalization error.

Unsupervised learning

Unsupervised algorithms are designed for the self-organization of ANNs. They can be derived either by the Hebbian law or by optimizing some global objective functions, such as entropy or mutual information.

Bottom-up self-organization

The Hebbian law is a local rule that requires only local signals to update the connections. It has several mathematical forms. Two typical bottom-up self-organization systems are Amari’s self-organizing neural fields38 and Kohonen’s self-organizing map39. In Amari’s model, the connections between presynaptic field and postsynaptic field are updated by the product of the postsynaptic neuron activity and the postsynaptic neuron activity. In Kohonen’s model, the neighborhood and the winner-take-all are two important concepts. The central idea of the Kohonen self-organizing algorithm is to reinforce the weight vectors of the winner node and the nodes in its neighborhood during learning.

Top-down self-organization

The Hebbian law is local and biologically plausible. It is also very flexible with various mathematical forms. However, the global behavior of a learning rule based on the Hebbian law is often difficult to predict from its local mathematical form. Optimizing some global objective functions, which characterize the internal representation of a learning task in an ANN, is an alternative way to derive an unsupervised learning rule to achieve self-organization. Unlike the cost functions for supervised learning, the global objective functions for unsupervised learning are often subjectively chosen to measure the performance of the trained network for a certain task. Generally, an unsupervised algorithm derived from a global objective function cannot be transformed into a local Hebbian learning rule.

Factorial coding and the informax are two important concepts for top-down self-organization. Their relation is discussed in Box 2.

Another phrase for factorial coding is independent component analysis (ICA). A rigorous mathematical theory for ICA has been provided by Comon40. One application of ICA is the blind separation of sources in a linear mixture \(x = \theta x\) where the mixing matrix, \(A\), and sources, \(z = (z_1, \ldots, z_s)\) are both unknown. Bell and Sejnowski41 applied the informax to blind separation and derived the following on-line algorithm to maximize \(H(e)\):

\[
\frac{dW}{dt} = -\psi(W^T - \psi(W^T W))
\]
Box 2. The relation between factorial coding and infomax

The factorial coding and the informax are two strategies to eliminate redundancy in neural coding. To find a factorial coding of a random vector x is to find a mapping g: x → y such that the joint pdf p(x, y) = Πy(y) is factorial.

Nadal and Parga used the following dual model to discuss the relation between the factorial coding and the informax:

\[ u = x + \xi + f(x) + f(Wx) + \xi \]

where \( x \) is the input, \( \xi \) is a white Gaussian noise, and \( f(Wx) \) is a random variable that is generated in an additive noise, \( \text{H} \) does not depend on the channel parameter \( \text{W} \) and the function \( f \). Therefore, minimizing \( H(x) \) is equivalent to maximizing \( H(y) \). In the low noise limit, \( \text{H}(x) \to \text{H}(y) \) and

\[ H(y) = -KL[p(y)||\Pi y(y)] \leq 0 \]

where \( KL[p(y)||\Pi y(y)] \) denotes the Kullback-Leibler divergence between two probability density functions \( p(y) \) and \( \Pi y(y) \).

Assume \( f(y) \) are distribution functions, then \( \Pi y(y) \) is a pdf. When \( H(y) = 0 \), \( f(y) = \Pi y(y) \), i.e. \( y \) is the factorial coding of a linear transform \( y = Wx \), when \( H(x) \) is maximized.

\[ \frac{\partial H(y)}{\partial W} = -W^{-1}f(y)g(y) + W^{-1}f'(y)g'(y) \]

This approach is the same as Pham and Gariani’s Quasi-Maximum Likelihood approach. \(^{32}\) The above learning equation was optimized by using the natural-gradient ascent method. \(^{32}\) The optimized learning equation is

\[ \frac{\partial E}{\partial W} = -W^{-1}f(y)g(y) + W^{-1}f'(y)g'(y) \]

Another approach for ICA is to minimize the mutual information

\[ I(y) = -\log p(y) - \log \prod_{i=1}^{m} p(x_i) \]

The mutual information \( I(y) \) is an ICA contrast function, and is invariant to non-zero scaling, permutation and translation. The details about this approach can be found in Yang and Amari \(^{25}\) where an adaptive algorithm was proposed to estimate the marginal entropy \( H(y) \), and the relation between the informax and minimum mutual information was analyzed.

Although the ICA algorithms are usually not local learning rules, they are very useful in practice. Applications of these algorithms can be found in Makeig et al. \(^{44}\) for EEG data analysis, and Bell and Sejnowski \(^{45}\) for natural image analysis.

Combined learning

Breiman’s bagging method \(^{37}\), and Freund and Schapire’s arching (boosting) method \(^{46}\), are very useful ensemble learning methods. Internal to these ensemble methods is a voting mechanism, and the learning equation is constructed by voting in the following way:

\[ f(x) = \arg \max_{j \in \{1, \ldots, M\}} \sum_{m=1}^{M} \alpha_m I(x_j, x) \]

where \( \alpha_m \) are voting weights to be determined.

Given a training set \( X \equiv \{x_i, y_i\} \), \( i = 1, \ldots, N \) a classifier \( g(x) \) is trained on \( X \). A sequence of training sets \( X_m = \{x_i, y_i\} \), \( m = 2, \ldots, M \), are generated by resampling the \( X \) using probabilities \( p_m \). A classifier \( g(x) \) is trained on each training set \( X_m \).

The sampling probabilities \( p_m \) and the voting weights \( \alpha_m \) are updated by Freund and Schapire’s arching algorithm consisting of the following steps:

(1) Installing \( p_1 = \frac{1}{M} \) and \( m = 1 \)
(2) Let \( d_k = 1 \) if \( g_k(x) \neq y \), else \( d_k = 0 \), for \( (x,y) \in X_m \)
(3) Compute \( \epsilon = \sum_{i=1}^{N} d_k \) \( \beta_k = 1 + \epsilon \log \beta_k \), and update the probabilities

\[ p_{m+1} = p_m \exp \left( \sum_{i=1}^{N} d_k \right) \]

(4) Set \( m = m + 1 \)
(5) Repeat the steps 2–4 until \( m > M \)

Note that \( \epsilon_m \) is the error probability of the classifier \( g_m(x) \). When \( \epsilon_m \geq 0.5 \), the classifier \( g_m(x) \) should not be included in the vote because it will weaken the performance of the combined classifier. So it is reasonable to assume \( \epsilon_m < 0.5 \) for all \( m \). Under this assumption, we have \( \beta > 1 \) which means that the sampling probabilities of those patterns misclassified by \( g_m(x) \) are increased in the step 2 in the arching algorithm.

References


Box 3. Arching algorithm (adaptive boost)

A classifier on a space \( X \) is a function \( g(x) = y(x) \) where \( x \) is a pattern in \( X \). Each is a class label taking values in \( \{1, \ldots, K\} \) respectively. From \( M \) classifiers \( g_m(x) \), a better classifier can be constructed by voting in the following way:

\[ f(x) = \arg \max_{j \in \{1, \ldots, M\}} \sum_{m=1}^{M} \alpha_m I(x_j, x) \]

where \( \alpha_m \) are voting weight to be determined.

Given a training set \( X = \{x_i, y_i\} \), \( i = 1, \ldots, N \) a classifier \( g(x) \) is trained on \( X \). A sequence of training sets \( X_m = \{x_i, y_i\} \), \( m = 2, \ldots, M \), are generated by resampling the \( X \) using probabilities \( p_m \). A classifier \( g(x) \) is trained on each training set \( X_m \).

The sampling probabilities \( p_m \) and the voting weights \( \alpha_m \) are updated by Freund and Schapire’s arching algorithm consisting of the following steps:

(1) Initializing \( p_1 = \frac{1}{M} \) and \( m = 1 \)
(2) Let \( d_k = 1 \) if \( g_k(x) \neq y \), else \( d_k = 0 \), for \( (x,y) \in X_m \)
(3) Compute \( \epsilon = \sum_{i=1}^{N} d_k \) \( \beta_k = 1 + \epsilon \log \beta_k \), and update the probabilities

\[ p_{m+1} = p_m \exp \left( \sum_{i=1}^{N} d_k \right) \]

(4) Set \( m = m + 1 \)
(5) Repeat the steps 2–4 until \( m > M \)

Note that \( \epsilon_m \) is the error probability of the classifier \( g_m(x) \). When \( \epsilon_m \geq 0.5 \), the classifier \( g_m(x) \) should not be included in the vote because it will weaken the performance of the combined classifier. So it is reasonable to assume \( \epsilon_m < 0.5 \) for all \( m \). Under this assumption, we have \( \beta > 1 \) which means that the sampling probabilities of those patterns misclassified by \( g_m(x) \) are increased in the step 2 in the arching algorithm.

References

methods to improve the performance of the existing learning machines, including ANNs. Both methods combine the learning machines by voting. The arching method (see Box 3) uses an adaptive weighting procedure to increase the sampling probabilities of the examples that are difficult to learn for the network. Leisch and Hornik proposed a variant of the arching method to combine ANNs.

It is worth pointing out that animals might also use ensemble methods in learning. The brain might take the advantage of its intrinsic variability to implement bagging and arching in neural systems. Human beings often learn from examples repeatedly and selectively. If some examples are difficult to learn, they are thought to be important and beings learned once more. Without exaggerating, we can say that the brain is a natural bagging and arching machine.

Conclusions
It is a promising hypothesis that a future brain-like computer will be a statistical inference machine with a probabilistic model to deal with a stochastic environment. Statistical inference has offered systematic ways to derive not only the learning algorithms, but also objective measures, such as generalization error and NCC, to evaluate the performance of the algorithms.

The theoretical basis for techniques, such as early stopping, growing and pruning, is model selection. The schemes for model selection can be derived either by the conventional approach or by the Bayesian approach.

Unsupervised learning algorithms can be constructed according to the Hebbian law or can be derived from global objective functions. The Hebbian learning rules are local and biologically plausible, but the global behavior of the networks using Hebbian rules is difficult to predict. On the other hand, unsupervised learning algorithms derived from some global objective function might not have a local form but its global behavior is usually clear. Some non-local unsupervised learning algorithms, such as ICA algorithms, are very useful for data analysis.

Acknowledgment
We would like to thank Kaiyuan Su for proof reading of the manuscript.

References
We readily suppose an object may continue individually the same, though several times absent from and present to the senses; and ascribe to it an identity, notwithstanding the interruption of the perception, whenever we conclude, that if we had kept our eye or hand constantly upon it, it would have conveyed an invariable and uninterrupted perception.


There is a long-standing view that the notion of objecthood is one of the fundamental structures of human thought. Physical objects are a major focus of human attention in the first year of life, and structure visual attention in adults. We present a new theory of the ‘object-concept’ in infancy, drawing inspiration from ideas developed in the study of adult visual attention. According to our framework, a key component of object cognition is an internal representation which functions as an ‘index’ to a physical object in the world. Just as a finger that points at something conveys no information about the nature of what it points at, so too an ‘object index’, in our account, is an entirely abstract representation that conveys no information about the properties of the entity at which it points. However, once an index is pointing to an object, the properties of that object can be examined and featural information can be associated with, or ‘bound’ to, its index. The distinction between indexing and feature binding underwrites the distinction between object individuation and object identification, a distinction that turns out to be crucial in both the adult attention and the infant object-concept literature. By developing the indexing model, we draw together two disparate sets of literature and suggest new ways to study object-based attention in infancy.

Indexing and the object concept: developing ‘what’ and ‘where’ systems

Alan M. Leslie, Fei Xu, Patrice D. Tremoulet and Brian J. Scholl

The study of object cognition over the past 25 years has proceeded in two largely non-interacting camps. One camp has studied object-based visual attention in adults, while the other has studied the object concept in infants. We briefly review both sets of literature and distill from the adult research a theoretical model that we apply to findings from the infant studies. The key notion in our model of object representation is the ‘sticky’ index, a mechanism of selective attention that points at a physical object in a location. An object index does not represent any of the properties of the entity at which it points. However, once an index is pointing to an object, the properties of that object can be examined and featural information can be associated with, or ‘bound’ to, its index. The distinction between indexing and feature binding underwrites the distinction between object individuation and object identification, a distinction that turns out to be crucial in both the adult attention and the infant object-concept literature. By developing the indexing model, we draw together two disparate sets of literature and suggest new ways to study object-based attention in infancy.

A.M. Leslie, P.D. Tremoulet and B.J. Scholl are at the Department of Psychology and Center for Cognitive Science, Rutgers University, Piscataway, NJ 08855, USA. F. Xu is at the Department of Psychology, Northeastern University, Boston, MA 02115, USA. Tel: +1 732 445 6152 Fax: +1 732 445 6280 E-mail: aleslie@ruccs.rutgers.edu