Learning in Neural Networks

- Goal: To improve performance
- Means: interact with environment
- A process by which the adaptable parameters of an ANN are adjusted thru an iterative process of stimulation by the environment in which the ANN is embedded
- Supervised vs. unsupervised

Two-Time Scale Dynamics in Neural Networks

- Faster dynamics in neuron activities represented by *u* and *v*. Also called as short-term memory
- Slower dynamics in connection weight activities represented by *w*. Also called as long-term memory

Categories of Neural Networks

- Deterministic vs. stochastic, in terms of *F*
- Feedforward vs. recurrent, in terms of *G* and *H*
- Semilinear vs. higher-order, in terms of *G*
- Supervised vs. unsupervised, in terms of *L*

Properties of Neural Networks

1. Nonlinearity
2. Input-output mapping
3. Adaptivity
4. Contextual information
5. Fault tolerance
6. hardware implementability
7. Uniformity of analysis and design
8. Neurobiological analogy and plausibility

Threshold Logic Units

Proposition: Any logical function $F: \{0, 1\}^n \to \{0, 1\}$ can be implemented with a two-layer McCulloch-Pitts network.

Proposition: Uninhibited threshold logic units of McCulloch-Pitts type can only implement monotonic logical functions.

单调

Finite Automata

An automaton is an abstract device capable of assuming different states which change according to the received input and previous states.

A finite automaton can take only a finite set of possible states and can react to only a finite set of input signals.

Simple Perceptron

A simple perceptron is a computing device with a threshold logic unit. When receiving *n* real inputs thru connections with *n* associated weights, a simple perceptron outputs 1 if the net input of weighted sum is not less than the threshold, and outputs 0 otherwise.

Linear Separability

Two sets of data in an *n*-dimensional space are said to be (absolutely) linearly separable if *n*+1 real weights (including a threshold) exist such that the weighted sum of a datum in one set is always greater than or equal to (greater than but not equal to) the threshold and that in the other set is always less tan the threshold.

Perceptron Convergence Algorithm

1) Initialize weights and threshold randomly.
2) Calculate actual output of the perceptron: For all *p*

$$y^p = f\left(\sum_{i=1}^{n} w_i x_i^p - \theta\right)$$

1) Adapt weights: for all *p*

$$w_i(t+1) = w_i(t) + \eta(z^p - y^p)x_i^p, \eta > 0$$

2) Repeat until *w* converges.

Perceptron Convergence Theorem

If two sets of data are linearly separable, the perceptron learning algorithm converge to a set of weights and a threshold in a finite steps.

Limitations of Perceptrons

Only linearly separable data can be classified

The convergence rate may be low for high-dimensional or large number of data.

LMS Learning Algorithm

1) Initialize weights and threshold randomly.
2) Calculate actual output of the ADALINE:

$$y = \alpha\left(\sum_{i=1}^{n} w_i x_i - \theta\right), \alpha > 0$$

3) Adapt weights:

$$w_i(t+1) = w_i(t) + \sum_p \eta(z^p - y^p)x_i^p, \eta > 0$$

4) Repeat until *w* converges

Training Modes

Sequential mode: input training sample pairs one by one orderly or randomly.

Batch mode: input training sample pairs in the whole training set at each iteration.

Perceptron learning: either sequential or batch mode.

ADALINE training: batch mode only.

Perceptron vs. Adaline

Architecture: Perceptron uses bipolar or unipolar hardlimiter activation function, Adaline uses linear activation function.

Learning rule: Perceptron learning algorithm is not gradient-descent and can operate in either sequential or batch training mode, whereas Adaline learning (LMS) algorithm is gradient descent, but can only operate in batch mode.

Number of Logic Functions vs. Number of Threshold Functions

The number of threshold functions defined by hyperplanes is a function of $2^{n(n-1)}$ whereas that of logical functions is 2^{2^n}.

The learnbability problem: when *n* is large, there is not enough classification regions in weight space to represent all logical functions.

Cover's Theorem (1965):

A dichotomy $\{X^+, X^-\}$ is said to be φ-separable if there exist an *m*-dimensional vector w such that

- $w^T\varphi(x)$ 0, if x in X^+
- $w^T\varphi(x) < 0$, if x in X^-
- The hyperplane defined by $w^T\varphi(x) = 0$ is the separating surface between the two classes.

Backpropagation Algorithm

Also known as generalized delta rule.

Invented and reinvented by many researchers, popularized by the PDP group at UC San Diego in 1986.

A recursive gradient-descent learning algorithm for multilayer feedforward networks of sigmoid activation function.

Compute errors backward from the output layer to input layer.

Minimze the mean squares error function.

Backpropagation Algorithm (cont'd)

1) Initialize weights and threshold randomly.
2) Calculate actual output of the MLP:
3) Adapt weights for all layers:

$$w_{ij}(t+1) = w_{ij}(t) - \eta \sum_p \frac{\partial E_p}{\partial w_{ij}}$$

4) Repeat until *w* converges

Radial Basis Functions

A radial basis function (RBF) is a real-valued function whose value depends only on the distance from its origin or center.

Related to kernel theory in statistical learning.

Used as the means for approximating or interpolating multivariate functions.

An RBF network can transform the linearly inseparable XOR data in the input space to linearly separable data in the hidden state space.

Extreme Learning Machine

Proposed by Guangbin Huang at NTU in mid 2000's

One-layer feedforward architecture

Random connection weights from inputs to hidden neurons

Fast learning process for weights in output layer.

Local minima eliminated

Support Vector Machine

- Proposed by Vladmir Vapnik based on statistical learning and kernel theory in early 1990's.
- Minimization of structural risk.
- Maximal generalization power.

SVM Learning

1. Choose a kernel function
2. Choose a value for *C*
3. Solve the quadratic programming problem (many software packages available)
4. Construct the discriminant function from the support vectors

SVM Summary

1. Maximal Margin Classifier
   - Better generalization ability & less over-fitting
2. The Kernel Trick
   - Map data points to a higher dimensional space to make them linearly separable.
   - Since only dot product is used, we do not need to represent the mapping explicitly.

MAXNET

A sub-network for selecting the input with maximum value - winner takes all.

By means of mutually prohibition, a MAXNET keeps the maximal input and presses down the rest.

It is often used as the output layer in some existing neural networks

A recurrent neural network with self excitatory connections and laterally inhibitory connections.

The weight of self excitatory connections is 1.

The weight of self inhibitory connections is -*w* where *w*<1/*m*, and *m* is the number of output neurons.

Desirable Properties

The *k*WTA model with Heaviside activation function has been proven to be globally stable and globally convergent to the *k*WTA solutions in *finite time*.

Derived lower and upper bounds of convergence time are respectively

$$\bar{t} \geq \frac{\epsilon|\bar{y} - y_0|}{\max\{n-k, k\}} \qquad \bar{t} \leq \epsilon|\bar{y} - y_0|.$$

Vigilance Parameter in ART1 Network

Value ranges between 0 and 1.

A user-chosen design parameter to control the sensitivity of the clustering.

The larger its value is, the more homogenous the data are in each cluster.

Determine in an *ad hoc* way.

ART1 Network

Invented by Stephen Grossberg at Boston University in 1970's.

Used to cluster binary data w/ unknown cluster number.

A two-layer recurrent neural network.

MAXNET serves as its output layer.

Bidirectional adaptive connections called bottom-up and top-down connections.

ART1 for Clustering

1) Initialize weights: $w_{ij}^{td}(0)=1, w_{ij}^{bu}(0)=\frac{1}{1+n}$
2) Compute net input for an input pattern x^p:
$$u_i^p=\sum_{j=1}^{n} w_{ij}^{bu}(t)x_j^p, i=1,2,\ldots,m$$
3) Select the best match using the MAXNET $u_k^p=\max_i\{u_i^p$
4) Vigilance test: If $\sum_{j=1}^{n} w_{kj}^{td}(t)x_j^p \Big/ \sum_{j=1}^{n} x_j^p \geq \rho$ then next; otherwise, disable neuron k and go to step 2).
5) Adapt weights:
$$w_{kj}^{td}(t+1)=w_{kj}^{td}(t)x_j^p, w_{kj}^{bu}(t+1)=\frac{w_{kj}^{td}(t)x_j^p}{0.5+\sum_{j=1}^{n} w_{kj}^{td}(t)x_j^p}$$

Hopfield Networks

- Invented by John Hopfield at Princeton University in 1980's.
- Used as associative memories or optimization models.
- Single-layer recurrent neural networks.
- The discrete-time model uses bipolar threshold logic units and the continuous-time model uses unipolar sigmoid activation function.

Stability Conditions

- Stability: $\lim_{t\to\infty} v(t)=\bar{v}$
- Sufficient conditions:

1. $w_{ij}=w_{ji}, w_{ii}=0; i,j=1,2,\ldots,n$
2. Activation is conducted asynchronously; i.e., the state updating from $v(t)$ to $v(t+1)$ is performed for one neuron each iteration.

Limitations

- Very limited capacity: $\frac{n}{2\log n}$, where n is the memory length
- Many spurious states; e.g., $-s^q$

Associative memories are content-addressable mechanisms for storing prototype patterns such that the stored patterns can be retrieved with the recalling probes (cues).

Memory Processes

Storage (Information encoding)

Given a set of prototype patterns to be memorized, place them into the memory indefinitely.

- Retrieval (Information decoding)
- Given any probe (key or cue), recall the corresponding prototype patterns in the memory.

Discrete-Time Hopfield Network as an Optimization Model

- Formulate the energy function according to the objective function and constraints of a given optimization problem.
$$E(v)=-\frac{1}{2}v^TWv-x^Tv, v\in\{-1,1\}^n$$
- Form a Hopfield network, then update the states asynchronously until convergence.
- Shortcoming: slow convergence due to asychrony.

Continuous-Time Hopfield Network as an Optimization Model

- Formulate the energy function according to the objective function and constraints of a given optimization problem.
$$E(v)=-\frac{1}{2}v^TWv-x^Tv, v\in[0,1]^n$$
- Synthesize a continuous-time Hopfield network, then an equilibrium state is a local minimum of the energy function. .

Objective Function and Constraints

- Constraints (permutation matrix):
  - One neuron should be on in each row
  - One neuron should be on in each column
- Objective function:
  - Total distance should be minimized
- Determine the network weights and bias so that the Lyapunov function is minimized when constraints are met and objective function is minimum

Characteristics of Simulated Annealing

The higher the temperature, the higher the probability of an energy increase.

As the temperature approaches to zero, the simulated annealing procedure becomes an iterative improvement one.

The temperature parameter has to be lower gradually to avoid prematurity.

Boltzmann Machine

A stochastic recurrent neural network invented by G. Hinton (Univ. of Toronto) and T. Sejnowski (Salk Institute) in 1983.

It has binary state variables $\{-1, 1\}^n$ with a probabilistic activation function.

A parallel implementation of simulated annealing procedure.

It can be seen as a stochastic, generative counterpart of the Hopfield networks.

Mean Field Annealing Network

- A deterministic recurrent neural network.
- Based on mean-field theory.
- Continuous state variables on $[-1, 1]^n$.
- use a bipolar sigmoid activation function.
- Use a gradual decreasing temperature parameter like simulated annealing.
- Used for combinatorial optimization.

Self-Organizing Maps (SOMs)

Developed by Prof. T. Kohonen at Helsinki University of Technology in Finland in 1970's.

A single-layer network with a winner-take-all layer using a unsupervised learning algorithm.

Formation of topographic map through self-organization.

Map high-dimensional data to one or two dimensional feature maps.

Kohonen's Learning Algorithm

1. (Initialization) Randomize $w_{ij}(0)$ for i = 1,2,…n; j = 1,2,…m; p = 1, t = 0.
2. (Distance) for datum x^p, $d_j=\sum_{i=1}^{n}[x_i^p-w_{ij}(t)]^2$
3. (Minimization) Find k such that $d_k=\min_j d_j$
4. (Adaptation)
$$\forall j\in N_k(t), i=1,2,\ldots n, \Delta w_{ij}(t)=\eta(t)[x_i^p-w_{ij}(t)]$$
$$0\leq\eta<1, d\eta/dt<0, p\leftarrow p+1, \text{goto Distance.}$$

Echo State Network

Proposed by Herbert Jaeger and Harald Haas at Jocobs University in 2004.

Also called reservoir computing.

It is a recurrent neural network with sparse connections and random weights among hidden neurons.

Fuzzy Set

Fuzzy set A is the set of all pairs $(x, u_A(x))$ where x belongs to X; i.e., $A=\{(x,u_A(x))\,|\,x\in X\}$

If X is discrete, $A=\sum_i u_A(x_i)/x_i$

If X is continuous, $A=\int_X u_A(x)/x$

Support set of A is $\mathrm{supp}(A)=\{x\in X\,|\,u_A(x)>0\}$

Cardinality and Entropy of Fuzzy Sets

Cardinality: $|A|$ is defined as the sum of the membership function values of all elements in X; i.e., $|A|=\sum_{x\in X}\mu_A(x)$ or $|A|=\int_X \mu_A(x)dx$

Entropy: $E(A)$ measures fuzziness and is defined as
$$E(A)=\frac{|A\cap\bar{A}|}{|A\cup\bar{A}|}$$

Logic Operations on Fuzzy Sets

- Equality: For all x, $u_A(x)=u_B(x)$
- Degree of equality:
$$E(A,B)=\deg(A=B)=\frac{|A\cap B|}{|A\cup B|}$$
- Subset: $A\subseteq B$, if $u_A(x)\leq u_B(x), \forall x\in X$
- Subsethood measure:
$$S(A,B)=\deg(A\subseteq B)=\frac{|A\cap B|}{|A|}$$

Typical Defuzzifiers

Centoid (also know as center of gravity and center of area) defuzzifier:
$$x^*=\frac{\int_X x\mu(x)dx}{\int_X \mu(x)dx} \text{ or } x^*=\frac{\sum_i x_i\mu_A(x_i)}{\sum_i \mu_A(x_i)}$$

Center average (mean of maximum) defuzzifier:
$$x^*=\frac{\sum_i x_i ht(x_i)}{\sum_i ht(x_i)}$$

Fuzzy Inference Process

- When imprecise information is input to a fuzzy inference system, it is first fuzzified by constructing a membership function.
- Based on a fuzzy rule base, the fuzzy inference engine makes a fuzzy decision.
- The fuzzy decision is then defuzzified to output for an action.
- The defuzzification is usually done by using the centoid method.

Genetic Algorithms

- A stochastic search method simulating the evolution of population of living species.
- Optimize a fitness function which is not necessarily continuous or differentiable.
- A genetic algorithm generates a population of seeds instead of one in traditional algorithms.
- The computation of the population can be carried out in parallel.

Elements in Genetic Algorithms

- A coding of the optimization problem to produce the required discretization of decision variables in terms of strings.
- A reproduction operator to copy individual strings according to their fitness.
- A set of information-exchange operators; e.g., crossover, for recombination of search points to generate new and better population of points.
- A mutation operator for modifying data.

Swarm Intelligence

Initialized by Gerardo Beni and Jing Wang in 1989 in the context of cellular robotic systems

Typically made up of a population of simple agents interacting locally with one another and with their environment

Typical representatives include particle swarm optimization, ant colony optimization, etc.

Evolutionary Computation

- Population-based stochastic and meta-heuristic search algorithms for global or multi-objective optimization
- Motivated by the natural evolution based on Darwinist or other principles
- Use collective wisdom to accomplish given tasks via efforts of many generations.

Particle Swarm Optimization

A robust stochastic optimization technique based on the movement and intelligence of swarms

Applies the concept of social interaction to problem solving

It uses a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution

Particle Swarm Optimization

- Each particle is treated as a point in a multi-dimensional space which adjusts its "flying" according to its own flying experience as well as the flying experience of other particles