Artificial Neural Networks

Chapter 3: Supervised Learning Neural Networks
Supervised Learning Rules

Prof. D.Sc. BARSEKH-ONJI Aboud

Facultad de Ingeniería
Universidad Anáhuac México

January 14, 2026

Agenda

Supervised Learning Overview

Goal

In Supervised Learning, the network is provided with a dataset $D = \{(\mathbf{z}_p, \mathbf{t}_p)\}_{p=1}^{P}$, where:

- $\mathbf{z}_p$ is the input vector.
- $\mathbf{t}_p$ is the desired target vector.

The goal is to adjust the weights $\mathbf{W}$ to minimize the error between the actual output $\mathbf{o}_p$ and the target $\mathbf{t}_p$.

The Error Signal

The difference between the target and the actual output is the error signal.

$$\mathcal{E} = E(\mathbf{t} - \mathbf{o}) \tag{1}$$

Learning rules optimize the weights based on this error signal.

The Supervised Learning Problem

Definition

Supervised learning is the process of approximating an unknown function $\boldsymbol{\mu}(\mathbf{z})$ using a finite set of input-target pairs $D = \{d_p = (\mathbf{z}_p, \mathbf{t}_p) \mid p = 1, \ldots, P\}$.

Signal-Plus-Noise Model

The target is modeled as the true function plus noise:

$$\mathbf{t}_p = \boldsymbol{\mu}(\mathbf{z}_p) + \boldsymbol{\zeta}_p \tag{2}$$

where $\boldsymbol{\zeta}_p$ represents zero-mean, independent, identically distributed noise.

Data Organization

The dataset D is typically divided into:

1. **Training set (D_T)**: Used to find the approximation $f_{NN}(\mathbf{z}_p, \mathbf{W})$.
2. **Validation set (D_V)**: Used to determine memorization/overfitting.
3. **Test set (D_G)**: Used to estimate generalization accuracy.

Error Definitions

Empirical (Training) Error

The objective function to be minimized:

$$E_T(D_T, \mathbf{W}) = \frac{1}{P_T} \sum_{p=1}^{P_T} (f_{NN}(\mathbf{z}_p, \mathbf{W}) - \mathbf{t}_p)^2 \tag{3}$$

Generalization Error

The expected error on new data:

$$\mathcal{E}_G(\Omega, \mathbf{W}) = \int (f_{NN}(\mathbf{z}, \mathbf{W}) - \mathbf{t})^2 d\Omega(\mathbf{z}, \mathbf{t}) \tag{4}$$

Optimization Approaches

- **Local Optimization**: The algorithm may get stuck in a local optimum (e.g., Gradient Descent, Scaled Conjugate Gradient).
- **Global Optimization**: Searches larger areas of the search space (e.g., LeapFrog, Simulated Annealing, Evolutionary Algorithms, Swarm Optimization).

Agenda

Error-Correction Learning

Principle

Weights are modified only if the neuron responds incorrectly.

Perceptron Learning Rule: Used for binary classification (outputs 0 or 1).

$$v_i(t+1) = v_i(t) + \eta(t_p - o_p)z_{i,p} \tag{5}$$

- If $t_p = o_p$, error is 0, no change.
- If $t_p = 1, o_p = 0$, weights increase.
- If $t_p = 0, o_p = 1$, weights decrease.

Perceptron Convergence Theorem

Theorem

If the training data is linearly separable, the Perceptron Learning Rule is guaranteed to converge to a solution (a hyperplane that separates the classes) in a finite number of steps.

Agenda

The Delta Rule

Also known as the **Widrow-Hoff** or **Least Mean Squares (LMS)** rule.

- Designed for continuous activation functions (or linear units).
- Minimizes the Sum Squared Error (SSE):

$$\mathcal{E} = \sum_{p=1}^{P} (t_p - o_p)^2 \tag{6}$$

Update Rule

$$\Delta v_i = \eta (t_p - o_p) z_{i,p} \tag{7}$$

(assuming linear activation $f(net) = net$, so $f'(net) = 1$).

Agenda

Steepest Gradient Descent

For general non-linear differentiable activation functions (like Sigmoid), we use Gradient Descent.

Concept

The error surface can be visualized as a landscape. We want to move "downhill" towards the minimum error.

The weight change is proportional to the negative gradient of the error with respect to the weight:

$$\Delta v_i = -\eta \frac{\partial \mathcal{E}}{\partial v_i} \tag{8}$$

General Update Rule

Using the chain rule:

$$\frac{\partial \mathcal{E}}{\partial v_i} = \frac{\partial \mathcal{E}}{\partial o_p} \cdot \frac{\partial o_p}{\partial net_p} \cdot \frac{\partial net_p}{\partial v_i} = -(t_p - o_p) f'(net) z_{i,p} \tag{9}$$

Thus:

$$v_i(t+1) = v_i(t) + \eta(t_p - o_p) f'(net) z_{i,p} \tag{10}$$

This is the basis for **Backpropagation** in Multi-Layer Perceptrons.

Stochastic vs. Batch Learning

Stochastic (Online)

Weights are updated after presenting **each** pattern p.

- More noise, can escape local minima.
- Faster for large datasets.

Batch

Weights are updated after the **entire** dataset (epoch) is presented.

$$\Delta v_i = \sum_{p=1}^{P} \Delta v_{i,p} \tag{11}$$

- True gradient descent.
- Smooth convergence.

Agenda

Correlation Learning (Hebbian)

Hebbian Principle

"Cells that fire together, wire together."

Supervised Hebbian learning adjusts weights based on the correlation between input and target:

$$\Delta v_i = \eta t_p z_{i,p} \tag{12}$$

- If input z_i and target t_p are both high, the weight increases strongly.

Reinforcement Learning

In Reinforcement Learning, the network receives a **grade** or **reward** r rather than a precise target t.

- If the output is "good" (high reward), the weights corresponding to that output are strengthened.
- Often uses stochastic units allowing exploration.

Agenda

Summary

- **Error-Correction** drives the Perceptron to classify linearly separable data.
- **Gradient Descent** allows training of non-linear neurons (e.g., Adaline, MLP) by searching the error surface.
- **Stochastic vs. Batch** updates offer trade-offs between speed and convergence stability.
- These rules form the mathematical engine for training supervised neural networks.