

An Introduction to Computing with Neural Nets

Richard P. Lippmann

Abstract

Artificial neural net models have been studied for many years in the hope of achieving human-like performance in the fields of speech and image recognition. These models are composed of many nonlinear computational elements operating in parallel and arranged in patterns reminiscent of biological neural nets. Computational elements or nodes are connected via weights that are typically adapted during use to improve performance. There has been a recent resurgence in the field of artificial neural nets caused by new net topologies and algorithms, analog VLSI implementation techniques, and the belief that massive parallelism is essential for high performance speech and image recognition. This paper provides an introduction to the field of artificial neural nets by reviewing six important neural net models that can be used for pattern classification. These nets are highly parallel building blocks that illustrate neural-net components and design principles and can be used to construct more complex systems. In addition to describing these nets, a major emphasis is placed on exploring how some existing classification and clustering algorithms can be performed using simple neuron-like components. Single-layer nets can implement algorithms required by Gaussian maximum-likelihood classifiers and optimum minimum-error classifiers for binary patterns corrupted by noise. More generally, the decision regions required by any classification algorithm can be generated in a straightforward manner by three-layer feed-forward nets.

INTRODUCTION

Artificial neural net models or simply "neural nets" go by many names such as connectionist models, parallel distributed processing models, and neuromorphic systems. Whatever the name, all these models attempt to achieve good performance via dense interconnection of simple computational elements. In this respect, artificial neural net structure is based on our present understanding of biological nervous systems. Neural net models have greatest potential in areas such as speech and image recognition where many hypotheses are pursued in parallel, high computation rates are required, and the current best systems are far from equaling human performance. Instead of performing a program of instructions sequentially as in a von Neumann computer, neural net models explore many competing hypotheses simultaneously

using massively parallel nets composed of many computational elements connected by links with variable weights.

Computational elements or nodes used in neural net models are nonlinear, are typically analog, and may be slow compared to modern digital circuitry. The simplest node sums N weighted inputs and passes the result through a nonlinearity as shown in Fig. 1. The node is characterized by an internal threshold or offset θ and by the type of nonlinearity. Figure 1 illustrates three common types of nonlinearities; hard limiters, threshold logic elements, and sigmoidal nonlinearities. More complex nodes may include temporal integration or other types of time dependencies and more complex mathematical operations than summation.

Neural net models are specified by the net topology, node characteristics, and training or learning rules. These rules specify an initial set of weights and indicate how weights should be adapted during use to improve performance. Both design procedures and training rules are the topic of much current research.

The potential benefits of neural nets extend beyond the high computation rates provided by massive parallelism. Neural nets typically provide a greater degree of robustness or fault tolerance than von Neumann sequential computers because there are many more processing nodes, each with primarily local connections. Damage to a few nodes or links thus need not impair overall performance significantly. Most neural net algorithms also adapt connection weights in time to improve performance based on current results. Adaptation or learning is a major focus of neural net research. The ability to adapt and continue learning is essential in areas such as speech recognition where training data is limited and new talkers, new words, new dialects, new phrases, and new environments are continuously encountered. Adaptation also provides a degree of robustness by compensating for minor variabilities in characteristics of processing elements. Traditional statistical techniques are not adaptive but typically process all training data simultaneously before being used with new data. Neural net classifiers are also non-parametric and make weaker assumptions concerning the shapes of underlying distributions than traditional statistical classifiers. They may thus prove to be more robust when distributions are generated by nonlinear processes and are strongly non-Gaussian. Designing artificial neural nets to solve

Welcome to

ECE5424 CS5824

Advanced Machine Learning

Instructor: (Joseph) Yue Wang, Ph.D.

Bradley Dept. of Electrical & Computer Engineering
Virginia Polytechnic Institute and State University

Call me “Joseph”, no Dr.!



Outline


Textbook(s):

Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000.


T Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2001.

Simon Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice-Hall, Inc., 1999.

Hyvarinen, A., Karhunen, J. & Oja, E. *Independent Component Analysis*, John Wiley, New York, 2001.



This course will cover supervised learning, pattern classification, neural networks, support vector machine, unsupervised learning, cluster analysis, feature discovery, cross-validation, principal component analysis, independent component analysis, and matrix factorization with an emphasis on the strategic frontier between machine learning and big-data sciences. The course is theoretical, practical, and challenging in nature. Major effort will be made to present applications in the areas of computer-aided decision making, big data analytics, human genome data, and biomedical imaging.



Introduction to statistical and machine learning theory, statistical decision principles, linear discriminant analysis, Fisher criterion, single and multilayer perceptrons, radial basis function neural networks, support vector machine, feature selection, cross-validation, bootstrap and permutation re-sampling; principal component analysis, self-organizing maps, mixture model and data clustering, principal/independent component analysis, matrix factorization.

Reading completion and homework	30%
Term paper proposal	20%
Final term paper	50%

Introduction

College mathematics

ECE5605 Stochastic Signal Processing

Information Theory

Signal Detection and Estimation

Machine Learning

Adaptive Signal Processing

Data Mining

...

Digital Signal and Image Processing

Computer Vision and Graphics

Data and Information Visualization

Signal Detection and Estimation

Hypothesis Testing Bayes Risk and Decision Rule

Binary hypothesis testing problem:

$$H_0 : Y \sim P_0$$

$$H_1 : Y \sim P_1$$

Given an observation y , how do we determine from which hypothesis (H_0 or H_1) it originated?
Bayes decision rule minimizes the average risk.

1. Terminology/Formulae

(a) Prior probabilities

π_j = a priori probability that H_j is the true source

(b) decision rule

A partition $\Gamma = \Gamma_0 \cup \Gamma_1$ where $y \in \Gamma_0$ implies the choice H_0 and $y \in \Gamma_1$ implies the choice H_1 .

Sometimes denoted $\delta(y) = \begin{cases} 1 & y \in \Gamma_1 \\ 0 & y \in \Gamma_0 \end{cases}$.

(c) cost function

C_{ij} = cost of choosing H_i when H_j is true

(d) conditional risk

$P_j(\Gamma_i)$ = probability of choosing H_i given that H_j is true

$R_j(\delta) = C_{1j}P_j(\Gamma_1) + C_{0j}P_j(\Gamma_0)$ = cost of rule δ given that H_j is true

(e) Bayes (or average) risk

$$r(\delta) = \pi_0 R_0(\delta) + \pi_1 R_1(\delta)$$

(f) Bayes decision rule

$$\delta(y) = \begin{cases} 1 & L(y) \geq \tau \text{ (i.e., choose } H_0) \\ 0 & L(y) < \tau \text{ (i.e., choose } H_1) \end{cases} \quad \text{where ...}$$

$$L(y) = \frac{p_1(y)}{p_0(y)} = \text{likelihood ratio}$$

$$\tau = \frac{\pi_0(C_{10} - C_{00})}{\pi_1(C_{01} - C_{11})} = \text{threshold} = \frac{\pi_0}{\pi_1} \bigg|_{\text{uniform costs}} = 1 \bigg|_{\text{uniform costs \& equal priors}}$$

Note that this is a *likelihood ratio test* (LRT).

2. Method for determining the rule:

- (a) Calculate the likelihood ratio $L(y)$.
- (b) Try to reduce LRT to " $y \gtrless \tau'$ " or " $|y| \gtrless \tau'$ " or a test involving some other simple function of y .
- (c) Determine the decision regions Γ_0, Γ_1 .
- (d) Calculate the risk $r(\delta)$.

3. Caveats:

- (a) If y is discrete, need to determine $L(y)$ for each value of y .
- (b) The critical regions may be the union of **disjoint** sets.
- (c) Don't assume that the LRT has the form $y \gtrless \tau'$.
- (d) Make sure that $\Gamma_j \subset \Gamma$.

4. Note: a **complete solution** to an optimal Bayes detection problem includes the explicit statement of the decision rule and/or critical region **and** a calculation of the risk incurred by the rule.

Maximum Likelihood Parameter Estimation

A popular non-random parameter estimation strategy which chooses the estimate to maximize the likelihood of observing what was actually observed.

1. Terminology/Formulae:

(a) Maximum likelihood estimator

$$\hat{\theta}_{\text{ML}}(y) = \arg \max_{\theta \in \Lambda} p_{\theta}(y)$$

(b) Likelihood equation

$$\left. \frac{\partial}{\partial \theta} \log p_{\theta}(y) \right|_{\theta = \hat{\theta}_{\text{ML}}} = 0$$

- There may be many, one, or no solutions to the likelihood equation.
- If $\hat{\theta}$ is an ML estimate and $\hat{\theta}$ lies in the interior of Λ , then $\hat{\theta}$ is a solution to the likelihood equation.
- If $\hat{\theta}$ achieves the CRLB, then $\hat{\theta}$ is a solution to the likelihood equation.
- If $p_{\theta}(y)$ is from an exponential family, there is a unique solution to the likelihood equation.

(c) Consistency

An estimator is consistent if $\hat{\theta}_n$ converges to θ (as $n \rightarrow \infty$) in some probabilistic sense.

$$\hat{\theta}_n \xrightarrow{i.p.} \theta \quad \text{weakly consistent}$$

$$\hat{\theta}_n \xrightarrow{m.s.} \theta \quad \text{mean-square consistent}$$

$$\hat{\theta}_n \xrightarrow{a.s.} \theta \quad \text{strongly consistent}$$

(d) Asymptotic Efficiency

An estimator is *asymptotically efficient* if it is asymptotically unbiased *and* if its asymptotic variance approaches the CRLB.

2. Asymptotic Properties of MLE's for IID Observation Sequences:

(a) *Consistency:*

Under appropriate regularity conditions, $\hat{\theta}_n|_{\text{ML}} \xrightarrow{i.p.} \theta$.

(b) *Asymptotic normality:*

Under appropriate regularity conditions, $\sqrt{n} \left(\hat{\theta}_n|_{\text{ML}} - \theta \right) \longrightarrow \mathcal{N} \left(0, \frac{1}{i_\theta} \right)$ in distribution, where

$$\begin{aligned} i_\theta &= \mathbf{E}_\theta \left\{ \left(\frac{\partial}{\partial \theta} \log f_\theta(y_1) \right)^2 \right\} \quad \text{"information per sample"} \\ &= \frac{I_\theta}{n} \quad (\text{since we assume i.i.d. observations}) \end{aligned}$$

(c) *Asymptotic efficiency:*

The consistency and asymptotic normality properties above, with additional regularity conditions, imply that *ML estimates are asymptotically efficient* (hence asymptotically MVUE).

3. Vector Parameter Case ($\Lambda \subset \mathbb{R}^m$):

(a) Likelihood equation

A system of m equations:

$$\begin{aligned}\frac{\partial}{\partial \theta_1} \log p_{\underline{\theta}}(y) \Big|_{\underline{\theta}=\hat{\underline{\theta}}_{\text{ML}}} &= 0 \\ \vdots & \\ \frac{\partial}{\partial \theta_m} \log p_{\underline{\theta}}(y) \Big|_{\underline{\theta}=\hat{\underline{\theta}}_{\text{ML}}} &= 0\end{aligned}$$

(b) Fisher information matrix

$$\begin{aligned}\mathbf{I}_{\underline{\theta}} &= \mathbf{E}_{\underline{\theta}} \left\{ \left(\frac{\partial}{\partial \underline{\theta}} \log p_{\underline{\theta}}(y) \right) \left(\frac{\partial}{\partial \underline{\theta}} \log p_{\underline{\theta}}(y) \right)^T \right\} \\ \text{where } \frac{\partial}{\partial \underline{\theta}} \log p_{\underline{\theta}}(y) &= \left(\frac{\partial}{\partial \theta_1} \log p_{\underline{\theta}}(y), \dots, \frac{\partial}{\partial \theta_m} \log p_{\underline{\theta}}(y) \right)^T\end{aligned}$$

(c) Cramér-Rao lower bound (CRLB)

If, for matrices A and B , we use $A \geq B$ to denote that $(A - B)$ is non-negative definite, then

$$\begin{aligned}\text{Cov}_{\underline{\theta}}(\hat{\underline{\theta}}) &= \mathbf{E}_{\underline{\theta}} \left\{ (\hat{\underline{\theta}} - \underline{\theta})(\hat{\underline{\theta}} - \underline{\theta})^T \right\} \geq \mathbf{I}_{\underline{\theta}}^{-1} \\ \Rightarrow \text{Var}_{\underline{\theta}}(\hat{\theta}_i) &\geq [\mathbf{I}_{\underline{\theta}}^{-1}]_{i,i}\end{aligned}$$

(d) Asymptotics for i.i.d. observations

Under similar conditions as in the scalar case, consistency and asymptotic normality still hold:

- $\|\hat{\underline{\theta}}_n - \underline{\theta}\|_2 \longrightarrow 0$ in probability
- $\sqrt{n} \left(\hat{\underline{\theta}}_n - \underline{\theta} \right) \longrightarrow \mathcal{N}(\underline{0}, \mathbf{I}_{\underline{\theta}}^{-1})$ in distribution

Basic AML tasks and issues

Supervised learning from data

- learning class conditional or posterior probability

- Learning decision boundary by partitioning space

- Selection of relevant features

- Predictive classification by trained classifiers

- The curse of dimensionality and the bias/variance dilemma (sample size vs dimensionality)

- Cross-validation schemes (expected performance)

- Bootstrap re-sampling (stability/uncertainty)

Basic PR tasks and issues

Unsupervised learning from data

- Data clustering for cluster discovery

- Learning the number of clusters (model selection)

- Selection of relevant features

Latent Variable Modeling LVM

- Principal component analysis (PCA)

- Independent component analysis (ICA)

- Convex analysis of mixtures (CAM)

- Non-negative matrix factorization (NMF)