TECHNICAL RESEARCH REPORT

A Better Activation Function for Artificial Neural Networks

by D.L. Elliott

T.R. 93-8

Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

A Better Activation Function for Artificial Neural Networks *

David L. Elliott
Institute for Systems Research
University of Maryland
College Park, MD 20742

ISR Technical Report TR 93-8

January 29, 1993

*Research supported in part by Washington University and by NeuroDyne, Inc.

Abstract

An activation function, possibly new, is proposed for use in digital simulation of artificial neural networks, on the ground that the computational operation count for this function is much smaller than for those employing exponentials and it satisfies a simple differential equation generalizing the logistic equation.

Introduction: activation functions

In the digital simulation of neural networks a feedforward memoryless neuron is represented by the input-output relation $y = \sigma(\sum_1^n w_k x_k)$ where $\sigma(\cdot)$, the sigmoidal activation or "squash" function, should have the property that it is positive monotone between the values -1 and 1 (or between 0 and 1) for $u \in (-\infty, \infty)$.

For use in finding optimal weights w_k to minimize $||y - y_{desired}||^2$ by backpropagation (gradient) algorithms, it is also a requirement that $\sigma(\cdot)$ be differentiable and that it satisfy a simple differential equation, thus permitting the evaluation of increments of weights via the chain rule for partial derivatives. In the seminal paper [1] on functional approximation with neural networks, the above properties are the only ones invoked.

For these reasons, $\sigma(\cdot)$ has almost always been one of the two closely related functions

$$\sigma_1(u) = \frac{1}{1+\exp(-u)}; \tag{1}$$

$$\sigma_2(u) = \tanh(u). \tag{2}$$

although other optimization schemes (not using gradients) may permit the use of the signum function or the saturation function

$$\begin{aligned} \sigma_3(u) &= u \text{ if } |u| < 1, \\ &= \operatorname{sgn}(u) \text{ if } |u| > 1. \end{aligned}$$

Noting that the derivatives are given by

$$\sigma_1'(u) = \frac{\exp(-u)}{(1+\exp(-u))^2}$$

and

$$\sigma_2'(u) = \operatorname{sech}^2(u);$$

substituting from (1) and (2) respectively, the differential equations for σ_1 and σ_2 are

$$\sigma_1' = \sigma_1(1-\sigma_1), \tag{3}$$

$$\sigma_2' = 1-\sigma_2^2. \tag{4}$$

The simplicity of these "Logistic Equations" (3),(4) permits easy computation of partial derivatives in the optimization algorithm.

The new squash function

Our proposed activation function is

$$\sigma_e(u) = \frac{u}{1+|u|} \tag{5}$$

and we have

Proposition 1 *The real function σ_e is differentiable everywhere with derivative*

$$\sigma_e'(u) = \frac{1}{(1+|u|)^2}$$

and satisfies a generalized logistic differential equation

$$\sigma_e' = (1-|\sigma_e|)^2. \tag{6}$$

Proof: Note that $\sigma_e(u)$ is differentiable for $u \neq 0$ and that its left and right derivatives at $u = 0$ are both equal to 1. Since

$$|x| = \frac{|\sigma_e|}{1-|\sigma_e|},$$

the differential equation (6) is satisfied.□

Equation (6) is no more complex than Eq. (3) or (4) and permits simple backpropagation code. The operation count for σ_e is an order of magnitude less than for the exponential functions used in σ_1 or σ_2. In production backpropagation code at NeuroDyne, Inc. (as well as the author's experiments with his own code and with Scott Fahlman's 1989 Quickprop code) the CPU time saved in net training is sufficiently substantial to warrant calling attention to σ_e.

It also nicely satisfies the properties needed by Barron [2] in proving that sigmoids approximate at a rate independent of the dimension of the input vector. Any previous use of σ_e and Eq. (6) in the backpropagation literature is unknown to the author.

References

[1] K. Hornik, H. White, M. Stinchcombe, "Multilayer feedforward networks are universal approximators," *Neural Networks* **2** 359-366, 1989.

[2] Andrew R. Barron, "Neural net approximation," *Proc. 7th Yale Workshop on Adaptive & Learning Systems*, New Haven, 69-72, 1992.