League of Legends Database Project Fall 2014

Brandon Fenton
Department of Mathematical Sciences
Montana State University
Bozeman, MT 59717
Email: brandon.fenton@gmail.com

Jenna Lipscomb
Department of Computer Science
Montana State University
Bozeman, MT 59717
Email: jennalynnlipscomb@gmail.com

John Sherrill
Department of Mathematical Sciences
Montana State University
Bozeman, MT 59717-2400
Email: prof.sherrill@gmail.com

***Abstract*—Two machine learning frameworks, a radial basis function network and a feedforward neural network, were created to approximate the generalized Rosenbrock function for a variable number of dimensions. An iterative k-fold cross validation process was implemented for both networks to tune model parameters and the resultant models were tested on a final large-scale test set to evaluate performance and thus compare the two different frameworks in the context of function approximation for this particular example. It was hypothesized that the feedforward neural network would possibly be able to model the algebraic representation of the Rosenbrock function and would thus outperform the radial basis function network. *This hypothesis was refuted by sound evidence from the model evaluation process indicating the the radial basis function performs better in this context*. The radial basis function network was trained by a recursive least squares implementation that was hypothesized to be significantly faster than the backpropogation algorithm used in training the feedforward neural network. *This was confirmed in all of the different configurations for testing the networks*.**

I. Introduction

The generalized N-dimensional Rosenbrock function may be expressed as

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$$

$$\text{where} \quad \mathbf{x} = [x_1, \ldots, x_N] \in \mathbb{R}^N.$$

This function was approximated for N = 2, 3, 4, 5, and 6 with each dimension constrained on the interval -1.5 to 1.5. That is, the domain was a 2, 3, 4, 5, and 6 dimensional hypercube $\mathbf{x} \in [-1.5, 1.5]^N$, $N \in \{2, 3, 4, 5, 6\}$.

These functions were to be approximated using a feedforward neural network and radial basis function network. The broad goal of this exercise was two fold: 1) comparing the rate of convergence to an exceptable level of error between the two frameworks and 2) comparing final model performance.

II. Network and Algorithm Descriptions

Two neural networks with different training algorithms were implemented. A feedforward (FF) neural network framework was created that utilizes stochastic gradient descent via the now famous backpropogation algorithm [1]. A radial basis function (RBF) network framework was also created that utilizes a k-means clustering algorithm for initializing model parameters (to be described in more detail below) and recursive least squares method for iteratively updating the model.

A. Feedforward Neural Network

A standard FF network consists of a sequence of l layers, each consisting of n_i different "nodes". The first layer is called the input layer and unsurprising takes as input the d_{in} different attributes of the data to be modeled. Thus, $n_1 = d_{in}$. The last layer is called the output layer and also unsurprising provides the output of the network, i.e. the predicted value for the attribute to be predicted. Thus, if the dimensionality of the output space is d_{out}, then $n_l = d_{out}$. The other $l - 2$ layers inbetween the input and output layers are referred to as hidden layers. Each node in hidden layer i is connected with all nodes in layer $i-1$ and $i+1$. Lastly, the edges in the graph described are all unidirectional, such that if one begins at an input node, one nescissarily ends at an output done. A simple 3 layer network is presented visually in Fig. 1 where $n_1 = 4$, $n_2 = 5$, and $n_3 = 1$. Excluding the edges contected to the input layer, for each edge there is an associated weight. Nodes take as input the weighted sum of outputs from their upsteam nodes (a node in layer i is considered upstream from nodes in layers $j > i$ and a node in layer i is considered downstream from nodes in layers $j < i$). The output value of a node is the value of a network-wide specified activation function of the input to the node. A typical activation function used is the logistic function or the hyperbolic tangent function. For this project, both the logistic function and a linear activation function were implemented with the choice left up to the user. If the linear activation function is chosen a slope must be supplied.

B. Radial Basis Function Network

III. Results

IV. Conclusion

References

[1] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations Volume 1: Foundations, MIT Press, Cambridge, MA.

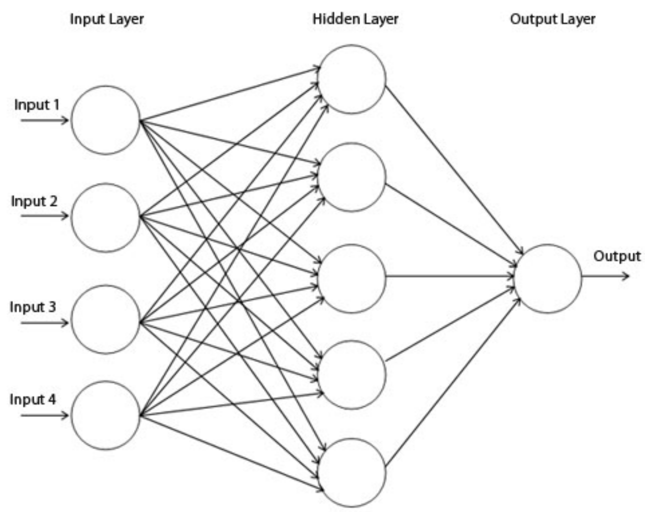

Fig. 1. Example Neural Network

TABLE I
RELATION SIZES

Relation Name	number of entities (n)
CHAMPIONS	122
ITEMS	280
BANNED	19,874
MATCH-CHAMPS	9,892
MATCH-CHAMP-ITEMS	62,781
MATCH-CHAMP-STREAKS	4,478