Sean Lantto

**Artificial Neural Network Individual Project**

MAE 565 – Spring 2017

Artificial Intelligence Techniques in Engineering

Department of Mechanical and Aerospace Engineering,

West Virginia University, Morgantown, WV 26506/6106

04/26/2017

# Abstract

Table of Contents

[Abstract I](#_Toc480925035)

[Table of Contents II](#_Toc480925036)

[Table of Symbols III](#_Toc480925037)

[Introduction 1](#_Toc480925038)

[Procedure 1](#_Toc480925039)

[Technical Approach 1](#_Toc480925040)

[Results 1](#_Toc480925041)

[Conclusions and Recommendations 1](#_Toc480925042)

[References 1](#_Toc480925043)

[Appendix A 1](#_Toc480925044)

# Table of Symbols

# Introduction

Artificial neural networks (ANN), are networks that are structured and operate like the way neurons in the brain work. They are networks of computational units that operate in parallel, that can “learn” from training data. An artificial neuron, can have multiple or a single input, where it then applies a weight (synaptic weights which are created/updated during learning). The weighted input is summed to create a total input which is then passed into an activation function to formulate the output. The output is either passed into other artificial neurons or out of the network. Artificial neural networks can be characterized by their information flow (feed forward or feedback), stratification (number of layers and neurons), and connectivity (degree of connectivity and type of connection). ANNs have many useful applications that can be grouped into function approximation (useful for control of systems), data processing, and classification.

A radial basis function (RBF) ANN, approximates non-linear functions by a non-linear mapping to a hidden layer of the ANN, then linearly maps the output of the hidden layer to the output layer. The hidden layer does not use inputs directly, instead using a basis function. The basis function has a center, which may or may not be the input.

# Procedure

*Figure 1* shows the general structure of the radial basis function ANN used in this work.

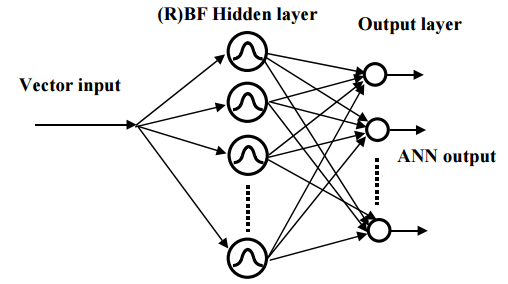


Figure 1: Radial Basis Function Artificial Neural Network

Prior to training the ANN, the centers and the variance of the Gaussian distribution for the basis function (shown below in **equation 1**) are defined.

Equation 1: Gaussian basis function used

The training process begins by first calculating the distance (*D*) from the chosen centers to the inputs of the training data. These distances are used in **equation 1** to get the output of the hidden layer. The weights for the input to the output layer are then computed by multiplying the inverse of the basis function against the known output of the training data. To validate these weights are correct, the previously calculated output of the hidden layer, is multiplied by the weights.

# Technical Approach

# Results

# Conclusions and Recommendations

# References

# Appendix A

% MAE 565 Aritificial Neural Network Indiividual Assignment

%Sean Lantto

%701168294

%April 26,2017

% RBF ANN - Arbitrary number of neurons (centers)

for TrialCenter=1:2

for TrialSigma=1:2

% close all

clearvars -except TrialCenter TrialSigma

load NN\_proj\_data\_06

inp1=inp\_train;

y=y6;

% figure,plot(inp1,y)

% title('Function To Be Approximated - Training Data')

% xlabel('Independent Variable x')

% ylabel('Function y=f(x)')

% grid

% Define the ANN parameters

if TrialCenter == 1

centers = inp1;

elseif TrialCenter == 2

centers = [-2:0.1:2];

end

if TrialSigma == 1

sigma = 0.05; % width of the Gaussian

elseif TrialSigma == 2

sigma=[-0.05:0.1/length(centers):0.05];

sigma=sigma(1:length(centers));

end

% Start the training process - only one iteration to determine the weights

if TrialSigma == 1

for i = 1:length(centers)

for j = 1:length(inp1)

D(i,j) = abs(centers(i)-inp1(j)); % Compute the distance between centers and the training data points (Matrix D)

PhiofD(i,j) = 1/sqrt(2\*pi\*sigma^2)\*exp(-D(i,j).^2/2/sigma^2);

end

end

elseif TrialSigma == 2

for i = 1:length(centers)

for j = 1:length(inp1)

D(i,j) = abs(centers(i)-inp1(j)); % Compute the distance between centers and the training data points (Matrix D)

PhiofD(i,j) = 1/sqrt(2\*pi\*sigma(i).^2)\*exp(-D(i,j).^2/2/sigma(i).^2);

end

end

end

% Compute the values of the Gaussian function for distances between centers and points

% = output of the HL

% Compute the weights

Z(:,TrialCenter,TrialSigma) = pinv(PhiofD)'\*y';

% ANN validation process

% Step #1 - Test the ANN with the inputs of the training data

for i = 1:length(inp1)

% distance to centers

d2c = abs(centers-inp1(i));

% output of the HL

if TrialSigma == 1

yHL = 1/sqrt(2\*pi\*sigma^2).\*exp(-d2c.^2/2/sigma^2);

elseif TrialSigma == 2

yHL = 1./sqrt(2\*pi.\*sigma.^2).\*exp(-d2c.^2./2./sigma.^2);

end

% output of ANN

YestO(i) = yHL\*Z(:,TrialCenter,TrialSigma);

end

% figure, plot(inp1,y,'k',inp1,YestO,'r')

% title('ANN Response to Training Data')

% legend('Training Data','ANN Estimation',2)

% xlabel('Independent Variable x')

% ylabel('Function y=f(x)')

% grid

% Step #2 - Test the ANN with different data but still produced by the

% same "function"

inp1val=inp\_valid;

yval=y6val;

% Compute the ANN estimation using the weights obtained through training

for i = 1:length(inp1val)

% distance to centers

d2c = abs(centers-inp1val(i));

% output of the HL

if TrialSigma == 1

yHL = 1/sqrt(2\*pi\*sigma^2).\*exp(-d2c.^2/2/sigma^2);

elseif TrialSigma == 2

yHL = 1./sqrt(2\*pi.\*sigma.^2).\*exp(-d2c.^2./2./sigma.^2);

end

% output of ANN

YestO(i) = yHL\*Z(:,TrialCenter,TrialSigma);

end

TrialFile = ['RBF\_ANN\_CenterCase' num2str(TrialCenter) '\_SigmaCase' num2str(TrialSigma)]

save(TrialFile);

figure, plot(inp1val,yval,'k',inp1val,YestO,'r')

title('ANN Response to Validation Data')

legend('Validation Data','ANN Estimation')

xlabel('Independent Variable x')

ylabel('Function y=f(x)')

grid

end

end

clear

figure;

for TrialSigma = 1:2

for TrialCenter = 1:2

TrialFile = ['RBF\_ANN\_CenterCase' num2str(TrialCenter) '\_SigmaCase' num2str(TrialSigma)]

load(TrialFile);

plot(inp1val,yval-YestO); hold on

title('Error of the ANN Response to Validation Data Input')

xlabel('Independent Variable x')

ylabel('Estimation Error')

grid on

end

end