DEPARTMENT OF CIVIL ENGINEERING AND CONSTRUCTION

BRADLEY UNIVERSITY

Illinois Asphalt Pavement Association

Scholarship Research Report

Neural Network Modeling of Pavement Rutting through Climate Data

Eric Winkelman
Senior Civil Engineering Student
January 23, 2023

ABSTRACT

Pavement rutting negatively impacts vehicular travel by potentially damaging tires; this can bring about an economic burden to both citizens and industry. The ability to predict pavement rutting through an Artificial Neural Network (ANN) can ensure rutting is fixed before severe damage to tires can occur. Climate and rutting data were collected from the Long Term Pavement Performance (LTPP) database to model an ANN that can predict pavement rutting accurately. An optimal model was determined through trial and error based on LTPP data from pavement sections in the Midwestern United States. This model was able to predict rutting in other pavement sections with high accuracy, which results in this method being a viable way to predict pavement performance and the need for repairs.

INTRODUCTION

Rutting is a permanent deformation of pavement that occurs over time due to structural and climate factors. When rutting occurs, a visible wheel path forms that can affect the lifespan of pavement and tires. It is necessary to fix rutting in the pavement to prevent tire damage and extend the life of a pavement structure. If a pavement structure is not repaired, it can result in economic losses and negatively affect people's daily lives.

OBJECTIVE

This paper aims to determine the effectiveness of using an Artificial Neural Network (ANN) model to predict pavement rutting due to climate factors. This will involve determining an optimal model and measuring prediction error.

Artificial Neural Networks can be trained based on existing data and used to predict a desired output variable. ANNs are helpful when there are several input variables; they can produce regression data much more quickly than through numerical analysis. Various parameters can be adjusted when creating ANNs. Hidden layers, neurons, training sets, and activation functions can be used to create and train an ANN.

Climate and rutting data were collected from the Federal Highway Administration's Long Term Pavement Performance (LTPP) database. LTPP includes records of pavement designs, traffic counts, climate, and performance over time. This data is collected for specific pavement sections designated by road signs and codes; a sample sign is located in Figure 1 below. For this paper, existing climate and rutting data from fifteen sections in the Midwestern United States will be used to train an ANN to predict pavement rutting.

Figure 1. LTPP Designated Pavement Section

MODEL TESTING

To create the model, pavement sections from the Midwest were selected so that all data used came from a similar climate zone. The sections selected from the LTPP database all had asphalt surface courses; these sections also contained several years of rutting data. Rutting measurements were compiled into an output spreadsheet in order for the ANN to have a model for output prediction. The following variable data was compiled into a separate spreadsheet and used as the ANN inputs: Annual Average Precipitation, Annual Average Temperature, Annual Average Freeze Index, Annual Average Humidity, AADTT, 18-Kip ESAL, and Time. These inputs were used to train an ANN to match the measured rutting output data.

The inputs pass through a set of neurons within hidden layers to collect the predicted rutting output data. The neurons process the input data by calculating its weighted averages using an activation function. There are three activation functions MATLAB uses: tansig, logsig, and purelin. The algorithms of each activation function are as follows:

$$tansig(x) = \frac{2}{1 + e^{-2x}} - 1$$

$$logsig(x) = \frac{1}{1 + e^{-x}}$$

$$purelin(x) = x$$

A training ratio of 70% Training Data, 15% Test Set, and 15% Validation Set was used to create the model. For all models, 20 iterations were conducted to train the network. To find the optimum number of neurons, models were tested using different activation functions with the number of neurons in one hidden layer ranging from 2-10. The tansig and logsig activation functions were used for testing in one hidden layer, since the purelin activation function would return negative predicted rutting values. Root Mean Square Error (RMSE) and Mean Absolute

Percentage Error (MAPE) were calculated based on the predicted output to validate the models' accuracy. To calculate the RMSE and MAPE, the model was used to compare predicted and actual rutting on all the selected LTPP pavement data. A model with a MAPE of less than 10% is considered ideal; the results of the models are displayed in Table 1 below.

Table 1. One-Layer Model Testing

# Neurons	Tansig		Logsig	
	RMSE	MAPE	RMSE	MAPE
2	1.2736	13.5558	0.8660	10.2541
3	0.6721	10.9080	0.9013	10.0982
4	0.4761	8.8244	3.0750	8.8974
5	0.5919	4.1720	0.7380	6.0550
6	0.3875	5.5907	0.5920	9.7622
7	0.4498	4.5445	0.4003	9.9364
8	0.3607	9.1473	0.5313	8.3269
9	0.3861	4.0959	0.4188	8.8081
10	0.3646	8.5192	0.4213	10.6675

It can be determined from this data that a 9-neuron model using the tansig activation function provides the greatest level of accuracy. This model results in the smallest MAPE, with a calculated value of approximately 4.1%. Further modeling can be done with more hidden layers and a combination of activation functions; 9 neurons were used in each layer for both 2 hidden layer and 3 hidden layer models. Table 2 displays the results of 2 hidden layer models with 9 neurons in each layer.

Table 2. Two-Layer, 9-Neuron Model Testing

Function Combination	RMSE	MAPE
Tansig-Logsig	0.4105	4.0774
Tansig-Tansig	0.3758	8.7079
Tansig-Purelin	0.3881	8.0212
Logsig-Logsig	0.4298	8.9044
Logsig-Tansig	0.4104	9.1728
Logsig-Purelin	0.4930	6.9111
Purelin-Logsig	0.3857	7.1775
Purelin-Tansig	0.3986	11.2639

From this data, the tansig-logsig function combination provides the smallest MAPE, approximately 4.08%. This MAPE is lower than the one-layer, 9-neuron tansig model result. One more set of trials was run using three hidden layers with 9 neurons in each layer. Table 3 contains the results of the three-layer tests.

Table 3. Three-Layer, 9-Neuron Model Testing

Function Combination	RMSE	MAPE
Tansig-Logsig-Logsig	0.3842	9.3802
Tansig-Tansig-Tansig	0.3199	9.1397
Tansig-Purelin-Purelin	0.4549	6.7588
Logsig-Logsig-Logsig	0.4095	9.2333
Logsig-Tansig-Tansig	0.4049	8.3873
Logsig-Purelin-Purelin	0.3883	11.1597
Purelin-Logsig-Logsig	0.4449	7.2254
Purelin-Tansig-Tansig	0.3471	6.9468

The three-layer function combined with the smallest MAPE is the tansig-purelin-purelin model, which is approximately 6.8%. This indicates that out of all the models tested, the two-layer, 9 neurons, tansig-logsig model results in the highest accuracy for predicting pavement rutting. The MATLAB code for the ANN model is located in the Appendix section; a graphical representation of the 7-9-9-1 model is depicted in Figure 2 below.

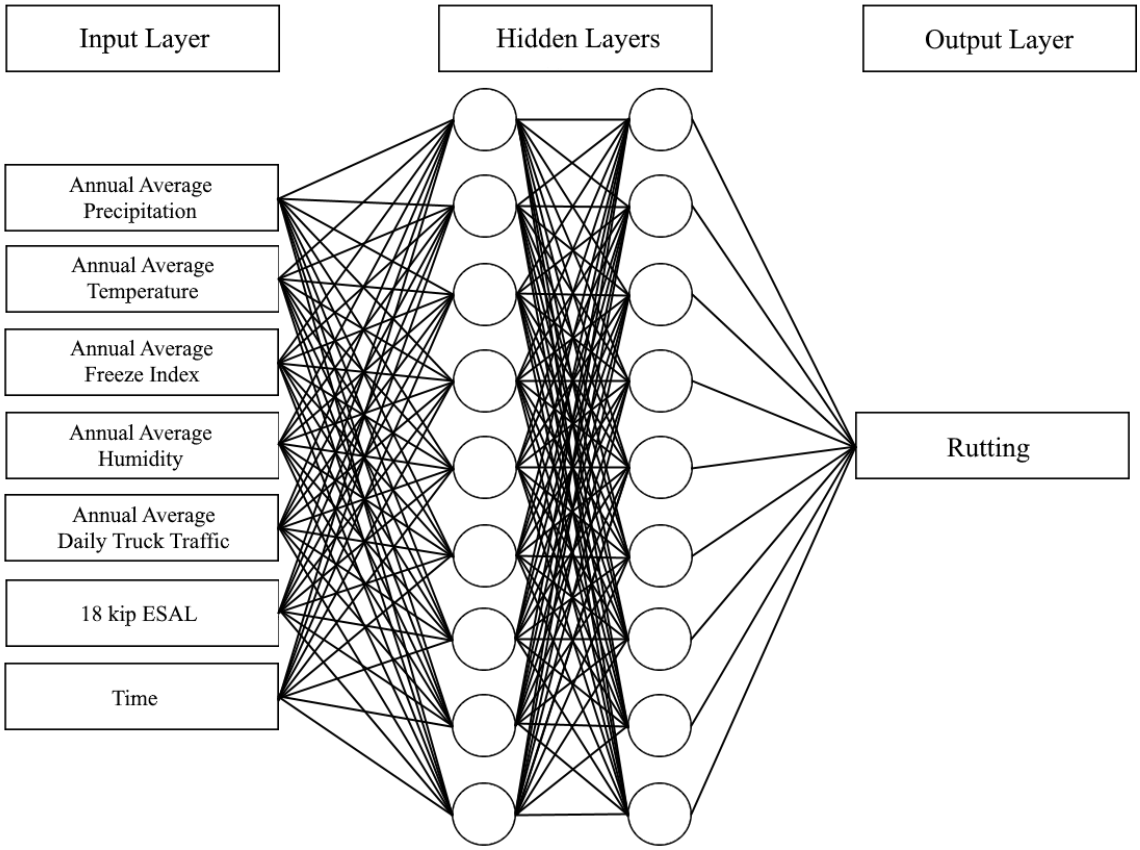

Figure 2. Diagram of 7-9-9-1 Model

Predicted rutting values from the 7-9-9-1 model are displayed in Figure 3 below. The predicted rutting is compared against the actual rutting from the Evansville, IN, pavement section. From the figure, the linear trend line of the predicted values contains similar results to the actual rutting values. Figure 4 displays the predicted rutting values of all the input data used to create

the model. Ideally, the predicted and actual values should align to form a linear trend line. There is some deviation with the model, but the R-squared value of 0.9516 indicates that this model should produce accurately predicted rutting values for pavement sections in the Midwestern United States.

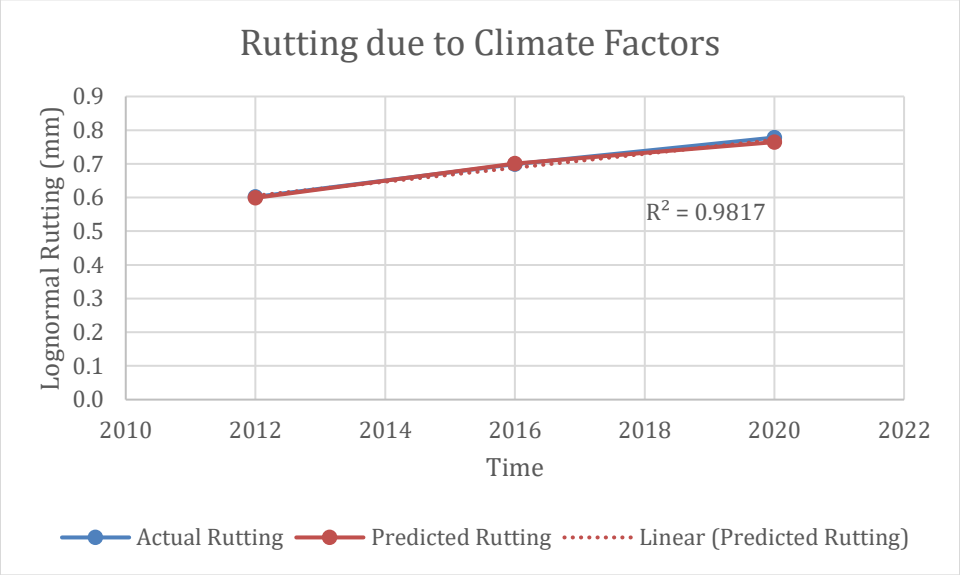

Figure 3. Comparison of Rutting on I-64 Section in Evansville, IN

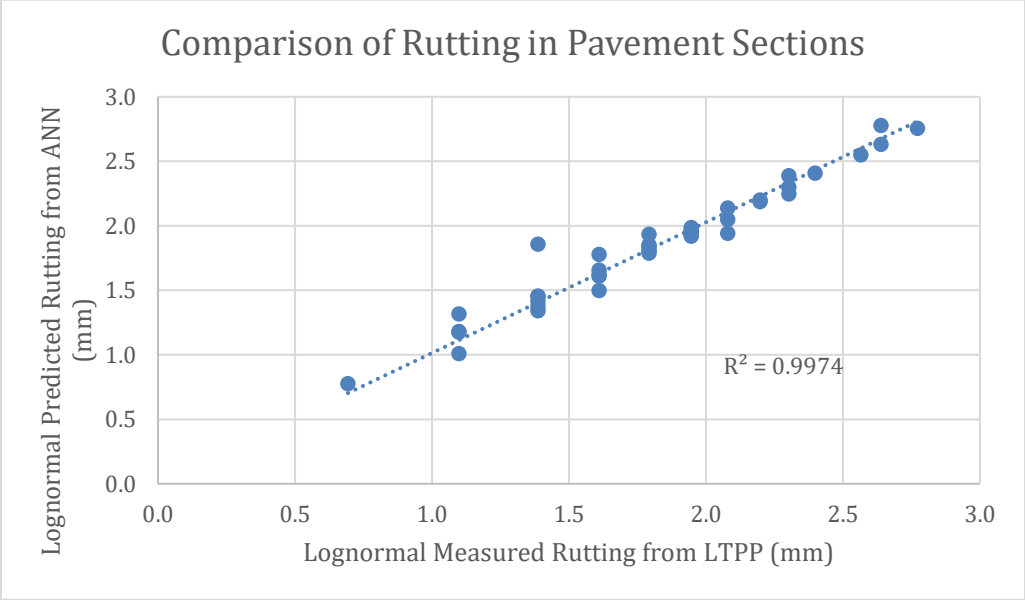

Figure 4. Comparison of Measured Rutting to Predicted Rutting

CONCLUSION

Predicting rutting accurately can aid state Departments of Transportation forecast when to repair rutting in pavement sections. The 7-9-9-1 ANN model used to predict rutting produces favorable results. This model is best used to predict rutting in pavement sections in the Midwestern United States. Modeling can also be done in pavement sections across various regions of the world. Further modeling could be performed on specific types of roads, such as arterials, local roads, and freeways.

REFERENCES

[1] M. Hossain, L. Gopisetti, M. Miah, Artificial Neural Network Modeling to Predict International Roughness Index of Rigid Pavements, Springer, International Journal of Pavement Research and Technology, 2020

[2] M. Hossain, L. Gopisetti, M. Miah, International Roughness Index Prediction of Flexible Pavements Using Neural Networks, ASCE, 2018, DOI 10.1061/JPEODX.0000088

[3] A. Ammari, MATLAB Code of Artificial Neural Networks Estimation, High Business School of Tunis, University of Manouba, Tunisia, 2016

APPENDIX

```
%% Data Input and Preparation
clc; clear; close all;
in=xlsread('_MasterData_Input_Days.xlsx');  % Input File
out=xlsread('_MasterData_Output.xlsx');    % Output File
data = [in out];
input=[1 2 3 4 5 6 7]; % Input Layers
p=data(:,input);
output=[8]; % Output Layer
t=data(:,output);
p=p'; t=t';

% Transposing Matrices
t = log(t+1);

% Defining Validation Dataset
trainRatio1=0.7;
valRatio1=0.15;
testRatio1=0.15;

% Network Definition
trainFcn = 'trainlm';  % Levenberg-Marquardt backpropagation.
nnn1=1;  % First Number of Neurons in the Hidden Layer
nnnj=1;  % Jump in Number of Neurons in the Hidden Layer
nnnf=9; % Last Number of Neurons in the Hidden Layer
net1.trainparam.lr=0.1;
net1.trainParam.epochs=500;

% Training Network Iterations
it = 20;

% Max Number of Iteration
ii = 0;
netopt{:}=1:nnnf;
for nnn=nnn1:nnnj:nnnf
    ii=ii+1; nnn;
    net1=newff(p,t,[nnn nnn]); % No. Hidden Layers
    evalopt(ii)=100;
    for i=1:it
        [net1,tr,y,et]=train(net1,p,t);
        net1.layers{1}.transferFcn = 'tansig'; % First Activation Function
        net1.layers{2}.transferFcn = 'logsig'; % Second Activation Function
        net1.divideParam.trainRatio=trainRatio1;
        net1.divideParam.valRatio=valRatio1;
        net1.divideParam.testRatio=testRatio1;
        estval=sim(net1,p(:,tr.valInd));
        eval=mse(estval-t(:,tr.valInd));
        if eval<evalopt(ii)
           netopt{(ii)}=net1;
           tropt(ii)=tr; evalopt(ii)=eval;
        end
    end
end
```

```
%% Error Plot
plot(nnn1:nnnj:nnnf,evalopt)

%% Output
nn = 1;
ptrain=p(:,tropt(nn).trainInd);
ttrain=t(:,tropt(nn).trainInd);
esttrain=sim(netopt{nn},ptrain);
ptest=p(:,tropt(nn).testInd);
ttest=t(:,tropt(nn).testInd);
esttest=sim(netopt{nn},ptest);
pval=p(:,tropt(nn).valInd);
tval=t(:,tropt(nn).valInd);
estval=sim(netopt{nn},pval);
estwhole=sim(netopt{nn},p);

% Calculation of RMSE
e = t-y;
pre_MAPE = abs((y-t)./t);
MAPE = mean(pre_MAPE(isfinite(pre_MAPE)))*100
RMSE = (mse(net1,t,y,'regularization',0.1))^(1/2)

% Transpose for Excel
t=t';
y=y';

%% Visuals
view(net1)
%figure; plot(ttrain,esttrain,'.b');
%figure; plot(tval,estval,'.g');
%figure; plot(ttest,esttest,'.r');
%figure; plot(t,estwhole,'.k')
figure;
plotregression(ttrain,esttrain,'Train',tval,estval,'Validation',ttest,esttest,'Test',
t,estwhole,'Whole Data');
```