BRADLEY University

Department of Civil Engineering and Construction

IAPA Scholarship Project

Utilizing Artificial Neural Network Technology to Predict Rutting Depth in Flexible Pavement

Name: Chase Gusewelle

Date: 23 January 2022

Spring 2022

Table of Contents

Introduction to Rutting

There is a multitude of factors that go into the design, construction, and overall usability of pavements. Without proper designs that follow guidelines set by state DOTs and federal regulations, along with poor construction practices, the functionality of the roadway can be compromised and even dangerous. With that being said, predicting inevitable pavement failures can be vital when it comes to the maintenance and creation of these asphalt sections.

One significant form of failure is pavement rutting. Pavement rutting is defined as a depression in a pavement structure at the location of a vehicle's wheel path. Figure 1, shown below, is a pronounced example of rutting on a typical roadway.

Figure 1. Typical Pavement Failure - Rutting

Rutting is vital to understanding a pavement structure because of the multitude of factors that play into its existence. These factors mainly fall under two categories, one of which relies more heavily on the pavement construction and the types of loads it is withstanding. At the same time, the other depends on the environment in which the pavement is performing. Most instances of rutting on a well-designed pavement structure only drop a few millimeters from its designed height. With so many underlying components, being able to predict how deep a pavement is going to rut over time can be a challenge. With the use of Artificial Neural Network modeling, it can be done with a more predictable level of certainty.

Introduction to Artificial Neural Networks

Artificial Neural Networks, or ANN for short, are highly complex models that use the same organizational structure as the human brain to create a solution for different input data sets. The model can go from an input value to an output value using a series of hidden layers containing neurons that all hold different weights in terms of importance. Figure 2 below shows this process as the model learns from the data.

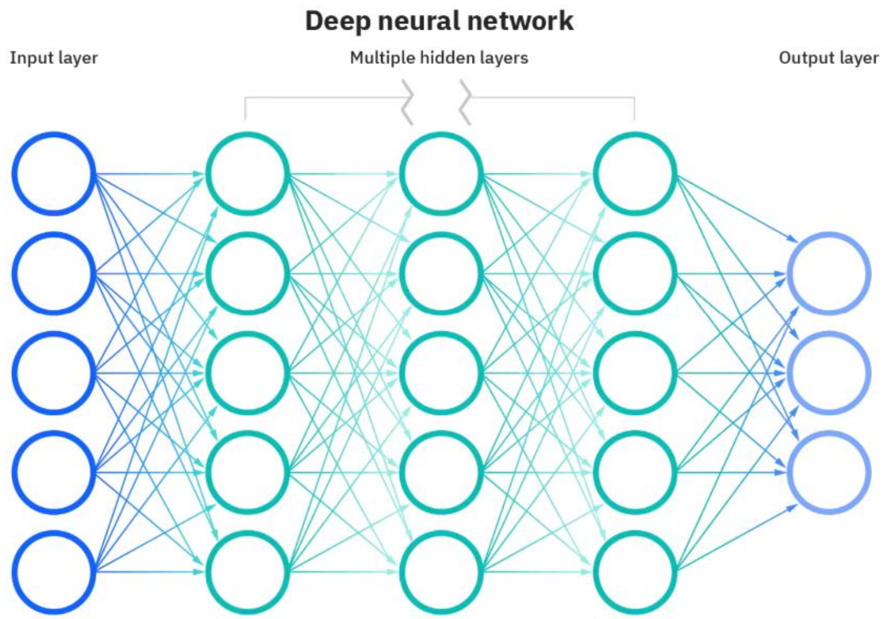

Figure 2. ANN Example Network

To create such a network, MATLAB is one valuable tool. Using this software, the user can specify the number of hidden layers, the number of neurons each layer contains, and how to train the data to produce a predictable model. To create this accurate model, three more factors of concern are how the model is trained, validated, and tested. The training set manipulates the significance of different layers in the neural network. In contrast, the validation set is used to control the different paths that will be taken inside the network (more of the structure of the neural network itself). The test set is used to evaluate the accuracy of the model once the output has been created. By utilizing these three sets of data, the network can be trained so that errors can be reduced to a minimum.

Another subset of the ANN modeling capabilities of MATLAB is transfer functions. Transfer functions are how the created network can decipher nonlinear data from input to output. Using the tansig and logsig transfer functions, a relationship between the neurons in the hidden layers is created to weigh their importance in creating an accurate rutting prediction.

Objective, Scope, and Limitations

This study aims to develop an ANN model that can accurately predict rutting in asphalt pavement structures using structural and loading data from the Long Term Pavement Performance (LTPP) infopave. This data includes values for structural number, 18-kip Equivalent Single Axle Loads, Annual Average Daily Truck Traffic, depth of asphalt concrete, and the number of pavement layers.

The structural number is a value from a pavement layer coefficient, thickness, and drainage coefficient that indicates a pavement strength capacity from its physical makeup or structure. The depth of asphalt concrete and the number of pavement layers has to do with the pavement structure and how it was designed to withhold different stresses applied to it. A pavement with a higher structural number can typically withstand more load before failure is induced from a loading standpoint. 18-kip Equivalent Single Axle Loads, or 18-kip ESAL for short, is a value used to represent the total load that an asphalt pavement has been subject to over a designated period of time converted to a value of 18,000-pound axle loads. This value allows a better structure for comparing loads on a pavement. Annual Average Daily Truck Traffic, or AADTT for short, is defined as the volume of truck traffic for 24 hours on a pavement structure spread over a year. Due to the weight of large vehicles, this value is representative of the loading on a pavement structure that can cause the most damage and rutting.

The scope of this research is limited to pavement values given by LTPP Infopave. Since not all pavement sections studied by Infopave contained information regarding the structural and loading data previously stated, values for pavement rutting in this paper are restricted to Illinois pavement locations that have been tested for up to five decades. LTPP Infopave was initially created as a gathering point for pavement data across the nation, all of which are to be used for further pavement testing and research. Having been used for countless research papers and studies, the values accumulated by the organization are considered helpful for this scope of research.

Modeling Process

Due to the uniqueness of the LTPP data for each pavement, the model had to be tested with different neurons, hidden layers, and transfer functions until a structure was found that had a certain degree of accuracy in predicting rutting depth. Before being able to run an accurate model, the optimal model constraints needed to be accumulated. Finding these optimal conditions meant using a combination of tansig and logsig transfer functions for two hidden layers while increasing the neurons in each layer with each model run. By continually trying these different scenarios, the ideal conditions could eventually be established by gauging each trials Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and r value (the strength of the model in terms of a linear relationship between predicted and actual rutting values). Table 1 below details this trial-and-error process to find the ideal model configuration.

Table 1. Trial-and-Error Process for Optimal Model Configuration

logsig-tansig										
Neurons	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RMSE	0.44	0.8	0.66	0.467	0.47	0.53	0.44	0.41	0.41	0.39
MAPE	15.01	8.78	7.61	14.35	12.1	14.4	14.2	11.02	13.69	5.65
R	0.4	0.72	0.714	0.49	0.49	0.37	0.38	0.65	0.5	0.87

tansig-logsig										
Neurons	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RMSE	1.4	0.66	0.47	0.66	0.69	0.47	0.48	0.42	0.41	0.42
MAPE	14.84	13.5	14.4	12.6	6.8	10.7	6.33	11.46	13.47	13.9
R	0.46	0.53	0.51	0.56	0.79	0.61	0.39	0.67	0.54	0.61

logsig-logsig										
Neurons	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RMSE	3.63	1	0.87	0.71	0.81	0.51	0.44	0.48	0.41	0.58
MAPE	14.23	10.5	1.3	12.8	14.9	12.8	14.2	8.9	14.3	13.23
R	0.47	0.75	0.45	0.48	0.12	0.61	0.61	0.76	0.46	0.71

tansig-tansig										
Neurons	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RMSE	0.54	0.54	0.75	0.51	0.45	0.43	0.39	0.38	0.38	0.34
MAPE	14.61	11.66	12.16	8.59	7.32	4.7	12.43	12.93	10.87	5.01
R	0.47	0.65	0.61	0.85	0.73	0.91	0.68	0.64	0.52	0.77

The RMSE of a model run, which is also a measure of differences between predicted and actual rutting values, is valid when it lies between 0.2 and 0.5. Between these two values, it is safe to say that the model is relatively accurate in prediction. Below is the equation used for RMSE to get this value.

$$RMSE = \sqrt{\frac{\sum_{I=1}^{N} (A - P)^2}{N}}$$

In this equation, N is the number of pavement test sections predicted in the model, A is the rutting depth collected from LTPP Infopave, and P is the predicted rutting depth created through the ANN model.

Another useful tool to determine which model structure will work best in terms of predicting accurate rutting depths is the MAPE equation. This equation shows how accurately the ANN model can forecast rutting depth. MAPE values below 5 are considered fantastic indications of a model's ability to predict rutting depth accurately. Below is the equation used to get the MAPE of each trial run.

$$MAPE = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{A-P}{A} \right|$$

In this equation, n is the number of pavement test sections being predicted in the model, A is the rutting depth collected from LTPP Infopave, and P is the rutting depth calculated through the ANN model.

After running the model with various constraints as stated above, the most accurate model structure, highlighted in the trial-and-error table, contained two hidden layers, each with 6 neurons using the tansig transfer function in both hidden layers. Below is an image consisting of an ANN model with these parameters.

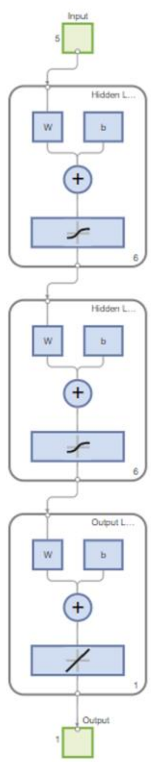

Figure 3. Optimal Model Configuration

With an RMSE of about 0.43, a MAPE of 4.7, and an r-value of 0.91, this structure was the most successful in the trial modeling process.

Results

Using the ANN model structure detailed above, MATLAB was utilized to develop data showing how well the model performed. The calculated MAPE of the ANN's prediction was 4.7, indicating that the model was extremely accurate in forecasting future rut depth in pavement structures. The RMSE calculated inside the software also fell into an acceptable range for the model's prediction efforts with a value of 0.43. Along with these prediction accuracy equations, graphs detailing R values and overall fit between the actual and predicted rutting data were also created. These graphs, shown below, were created in MATLAB to show how accurately the ANN model predicted rutting depth for the 29 pavement sections studied. Since actual rutting depths varied from 3 to 12 mm, MATLAB created these graphs with a factor to consolidate the data into more visually pleasing graphs. The final graph containing the totality of the data was also created in excel to show the R-squared value of the data, just another representation of how well the model could predict rutting values.

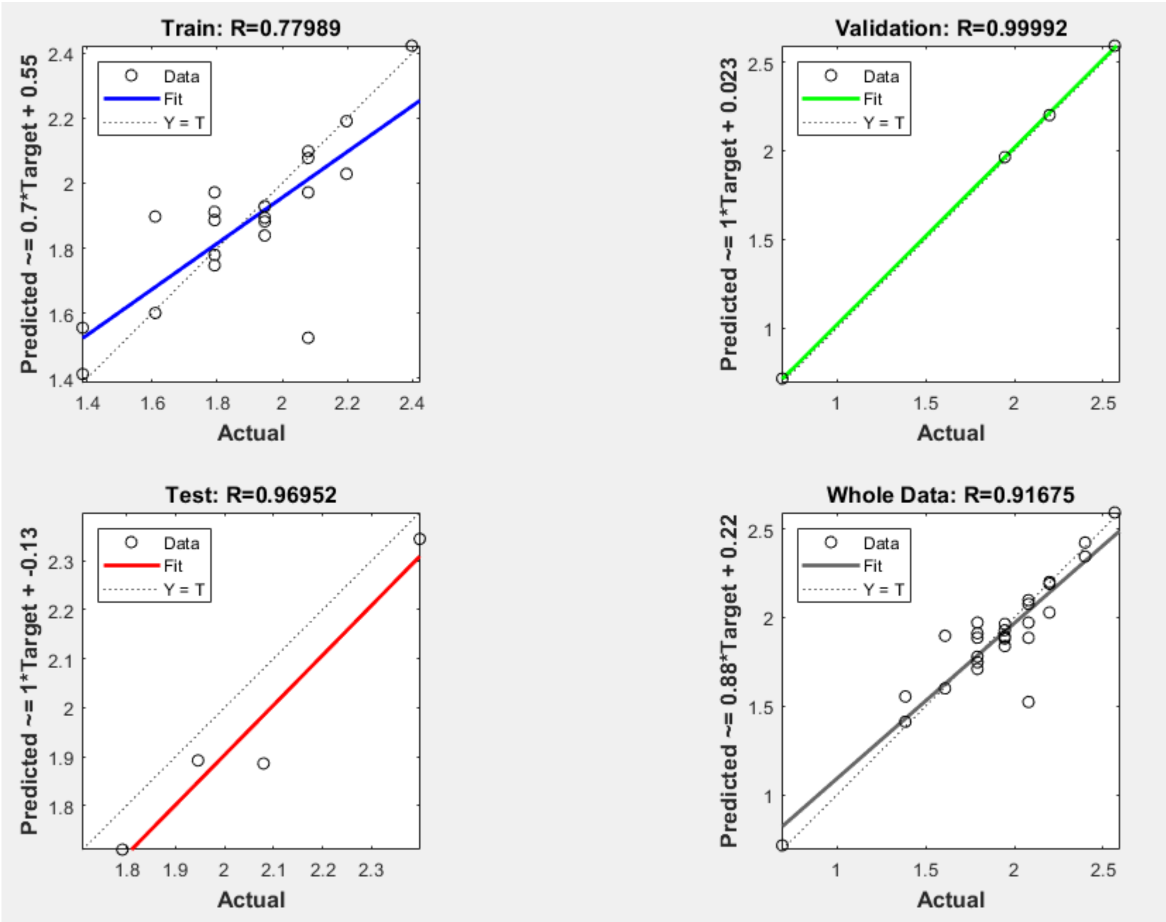

Figure 4. Predicted V Actual Rutting Depth for Training, Validation, and Test Data

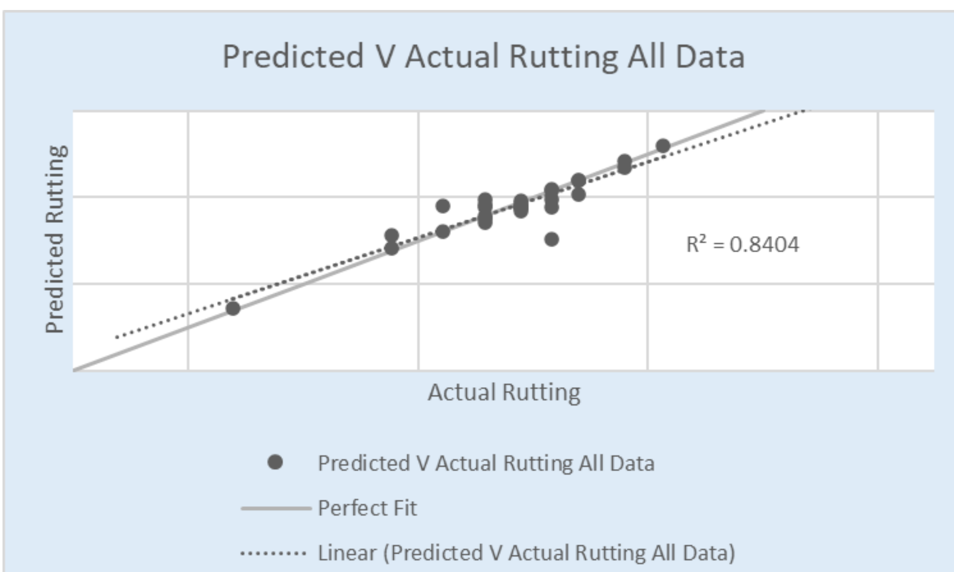

Figure 5. Predicted V Actual Rutting for Entire Dataset (With R squared value)

As can be seen, by the graphs shown, the data is trained using the tansig-tansig combination of transfer functions to have a relatively linear relationship that decides the importance of each hidden layer. From there, the model was able to accurately determine each neuron's weight throughout each hidden layer, which is shown by a nearly perfectly linear validation set ($r = 0.99$). Using these parameters, the test data proved to be accurate, with an R-value of 0.96. The combination of all these predictions for the entire set can be seen in the Whole Data graph, and in the prediction v actual rutting for all data graphs with an R squared value of 0.84. Considering an R squared value of 1 describes perfectly linear data, this model is sufficient with an error of about 16 percent in terms of predicted rutting depth in the pavement.

The reason for this error can be the cause of multiple parameters, the most important being the LTPP data itself. With a range of similar rutting depths to the nearest millimeter, structural numbers calculated and rounded by LTPP Infopave, and other discrepancies between structural and loading values, the model can only be so accurate given amounts of hidden layers, neurons and transfer functions. On top of this, there are some inconsistencies regarding rutting prediction between each complete model run. This is due to the code training, validating, and testing of different pavement sections with each trial. This makes it evident that some sections are more accurate at determining parameter importance inside the complete dataset. This could cause inconsistency when different pavement sections are entered into the program for rut prediction purposes.

Conclusion

Using MATLAB with the addition of LTPP Infopave pavement data, an Artificial Neural Network model consisting of hidden layers, neurons, and transfer functions with varying levels of importance was created to predict rutting depth in the pavement to a certain degree of accuracy. Using this research, rutting depth can be predicted to understand pavement sections in Illinois better and alleviate the severity of this failure in the future.

Reference

Ammari, Aymen. (2016). MATLAB Code of Artificial Neural Networks Estimation.

Hossain, M. I., Gopisetti, L. S. P., & Miah, M. S. (2019). International roughness index prediction of flexible pavements using neural networks. *Journal of Transportation Engineering, Part B: Pavements*, *145*(1), 04018058.

"Office Of Highway Policy Information - Policy | Federal Highway Administration." Fhwa.Dot.Gov, 2022, Accessed 30 Dec 2022.

"Pavement Design – What's My Structural Number? – Pavement Interactive". Pavementinteractive.Org, 2014, Accessed 30 Dec 2022.

Wang, Hongzhi, "Development of Laser System to Measure Pavement Rutting" (2005). Graduate Theses and Dissertations. https://scholarcommons.usf.edu/etd/903

"What Are Neural Networks? | IBM". *Ibm.Com*, 2022, https://www.ibm.com/topics/neural networks. Accessed 29 Dec 2022.

Appendix

Table 2. LTPP Infopave Data Utilized

Pavement Section	SN-Value	18 Kip ESAL	AADTT	Depth AC	Pavement Depth	RUTTING
1	5.9	74	474	11.9	2	6
2	5.8	74	474	11.8	2	5
3	6.0	157	715	13.9	4	8
4	6.5	157	715	14.7	4	3
5	7.1	141	645	12.1	6	7
6	5.8	157	715	13.2	3	6
7	5.9	157	715	13.3	3	5
8	5.8	157	715	13.2	3	6
9	7.0	141	645	11.8	7	3
10	7.2	141	645	12.4	7	10
11	7.1	47	247	0	2	6
12	7.0	141	645	12.1	7	10
13	6.5	87	292	12.1	2	5
14	5.6	76	338	12.8	4	7
15	5.8	74	474	11.8	2	7
16	6.0	157	715	12.2	2	8
17	6.8	38	208	0	1	1
18	7.2	39	212	0	1	4
19	7.1	39	212	0	1	6
20	6.9	141	645	11.6	6	12
21	7.0	39	212	0	1	5
22	5.9	38	208	0	1	5
23	5.6	39	212	0	1	4
24	5.8	39	212	0	1	6
25	5.9	38	208	0	1	7
26	6.0	39	212	0	1	5
27	5.3	10	49	7	8	8
28	5.8	70	312	0	1	6
29	7.2	181	825	12.2	6	7

Optimal MATLAB Code

```
%% Data Input and Preparation
clc; clear; close all;
in=xlsread('Input(1).xlsx');  % Input File
out=xlsread('Output(1).xlsx');    % Output File
data = [in out];
input=[1 2 3 4 5]; % Input Layers
p=data(:,input);
output=[6]; % 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 First Hidden Layer
nnnj=1;  % Jump in Number of Neurons in the first Hidden Layer
nnnf=6; % Last Number of Neurons in the First Hidden Layer
net1.trainparam.lr=0.1;
net1.trainParam.epochs=500;
% Training Network
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]);  % For more functions see: 'Function Reference' in
'Neural Network Toolbox' of Matlab help
    evalopt(ii)=100;
    for i=1:it
        [net1,tr,y,et]=train(net1,p,t);
        net1.layers{1}.transferFcn = 'tansig';
        net1.layers{2}.transferFcn = 'tansig';
        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 = 6;
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)
%% 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');
```