Modeling Complex Engineering Problems with a Neural Network

Kathryn Cloutier, Mileta Tomovic
Old Dominion University
Norfolk, Virginia
kclou002@odu.edu, mtomovic@odu.edu

ABSTRACT

With increasing technological complexity, engineers need creative methods for solving new problems. Matlab and Simulink offer engineers the opportunity to use Artificial Neural Networks to solve complex engineering problems. Artificial Neural Networks eliminate the need for engineers to generate complex equations and response surfaces; instead, large datasets can be used to determine mathematical relationships. Artificial Neural Networks are a beginning component of machine learning. They are conventionally defined as a series of algorithms that can recognize relationships in datasets. Artificial Neural Networks are designed to replicate brain neuron activity; with neuron processing units instead. Input values are weighted with a factor and fed into an activation function; the activation function generates a new output value. Artificial neural networks are excellent to use in the following area: pattern recognition, image compression, stock market prediction, machine translation, speech recognition, and more. For this system, the Artificial Neural Network was used to model the following scenarios: The velocity of a car, the displacement of a damped spring-mass system, predicted teacher evaluation scores, predicted precipitation amounts and predicted temperatures. The predicted response was compared to the data used to train the neural networks, and all the networks correlated well with the training data. The model accuracy was analyzed using model error histograms. The neural network models were also compared with other Matlab tools such as fuzzy logic controllers and Simulink. These features generate responses without the training data for the neural networks.

ABOUT THE AUTHORS

Kathryn Cloutier is a Mechanical Engineering Master's student at Old Dominion University. She is also a Chemical Product Engineer II with 6 years of experience. She graduated with a B.S. in Chemical Engineering from Rochester Institute of Technology. While a student at Rochester Institute of Technology, she completed two internships with Toyota Motor Manufacturing and All Cell Technologies. Kathryn's areas of interest are statistics, lean six sigma principles, mechanical engineering, and computer modeling.

Dr. Mileta Tomovic received BS in Mechanical Engineering from University of Belgrade, MS in Mechanical Engineering from MIT, and PhD in Mechanical Engineering from University of Michigan. Dr. Tomovic is currently serving as Mitsubishi-Kasei Professor of Manufacturing Technology, Batten College of Engineering and Technology, Old Dominion University, Norfolk, VA. Dr. Tomovic has seventeen years of teaching and research experience at Purdue University. Dr. Tomovic served as W. C. Furnas Professor of Enterprise Excellence, University Faculty Scholar, Director of Digital Enterprise Center, and Special Assistant to Dean for Advanced Manufacturing. He has co-authored one textbook on materials and manufacturing processes that has been adopted by over 50 national and international institutions of higher education. He has co-authored four patents, and over 100 technical reports on practical industrial problems related to product design and manufacturing process improvements. In addition, Dr. Tomovic has been actively involved in applied research, and has been a PI or Co-PI on number of externally funded competitive grants exceeding $6 million. In addition, he has been engaged with industry in solving manufacturing problems. The estimated savings to industry, resulting from Dr. Tomovic's recommendations, exceed $5 million over the ten years that he has been actively engaged with Technical Assistance Program, Purdue University.

Modeling Complex Engineering Problems with a Neural Network

Kathryn Cloutier, Mileta Tomovic
Old Dominion University
Norfolk, Virginia
kclou002@odu.edu, mtomovic@odu.edu

ABSTRACT

With increasing technological complexity, engineers need creative methods for solving new problems. Matlab and Simulink offer engineers the opportunity to use Artificial Neural Networks to solve complex engineering problems. Artificial Neural Networks eliminate the need for engineers to generate complex equations and response surfaces; instead, large datasets can be used to determine mathematical relationships. Artificial Neural Networks are a beginning component of machine learning. They are conventionally defined as a series of algorithms that can recognize relationships in datasets. Artificial Neural Networks are designed to replicate brain neuron activity; with neuron processing units instead. Input values are weighted with a factor and fed into an activation function; the activation function generates a new output value. Artificial neural networks are excellent to use in the following area: pattern recognition, image compression, stock market prediction, machine translation, speech recognition, and more. For this system, the Artificial Neural Network was used to model the following scenarios: The velocity of a car, the displacement of a damped spring-mass system, predicted teacher evaluation scores, predicted precipitation amounts and predicted temperatures. The predicted response was compared to the data used to train the neural networks, and all the networks correlated well with the training data. The model accuracy was analyzed using model error histograms. The neural network models were also compared with other Matlab tools such as fuzzy logic controllers and Simulink. These features generate responses without the training data for the neural networks.

INTRODUCTION

The engineering task of designing increasingly complex systems and assuring that they will perform as intended once built/manufactured is driving engineers to develop tools and methods that accomplish this task within a limited time and with desired outcomes. One of the most significant recent developments in engineering design is the evolution of the digital twin concept, which is based on a "virtual model designed to accurately reflect a physical object" [1]. Accurate models of physical systems allow engineers to perform what-if analyses which can lead to improved performance. Analysis of system design and its performance in the digital environment allows for multiple design iterations without committing resources to build the prototype. As a result, engineers can perform analyses with minimal investment of resources. In addition, digital twins allow for physical system monitoring in real-time and identifying potential issues that can lead to system failure, thus predicting the remaining useful life of the system/component [2].

The article presents some of the ANN examples introduced in the graduate courses MAE 682 "Concurrent Engineering" and MAE 785/885 "Advanced Manufacturing Technology." The examples presented in this paper and that students in these two courses used for practice are from the textbook published by Chaturvedi [3]. The students used that knowledge and expanded it to problems relevant to manufacturing processes.

The purpose of introducing ANN concepts/topics in these two courses is to expose students to modern tools that could be applied in engineering design, e.g. House of Quality for deciding on the relationship between Engineering Characteristics and Customer Requirements [4]; determining the quality of manufacturing process, e.g. monitoring of induction hardening process [5], quality of a weld [6]; or forecasting, e.g. forecasting product defects in manufacturing [7].

EXAMPLES

Example 1: Predicting the velocity of a car

One of the first methods that Artificial Neural Networks can be used is to predict behavior of linear and nonlinear dynamic systems. For example, ANNs can predict the velocity of a car as it accelerates on a highway. [3] For this application, the equation used to represent the velocity of a car can be seen in Eq. (1) below

$$F = M * \frac{dv}{dt} + bv \quad (1)$$

The following assumptions were made in the equation: M =1000 kg, F=80*t, and b=40 N/m/s. The Laplace transform was used to determine the final equation for the vehicle shown in Eq. (2) below.

$$V(t) = 2 * t + 50 * e^{-0.04t} - 50 \quad (2)$$

In traditional engineering classes, Eq. (1) and (2), are generated from Newton's Second Law. However, there is another way to obtain the same information. Specifically, using the Artificial Neural Network capabilities of MATLAB. Using this method, the engineering student does not need to use equations to understand the relationship between the vehicle's velocity and time. Instead, they can take experimental data and use the ANN capabilities of MATLAB to show the relationship instead. For this example, an Excel file with velocity data vs. time was imported into MATLAB and analyzed using the Deep Learning Toolbox. The Matlab code was able to generate the graph shown in Fig. 1.

The neural network used the Levenberg Marquardt (LM) Backpropagation Method to train the network. The LM method minimizes the mean square of error, and uses the smallest possible weights when training the data. As indicated in Fig. 1, the velocity curve is shown, as a blue line, with the trained data on top. The trained data matches the pattern of the original velocity curve, indicating that the training method worked well. However, the best way to check the model's effectiveness is to check the error bar histogram shown in Fig. 3.

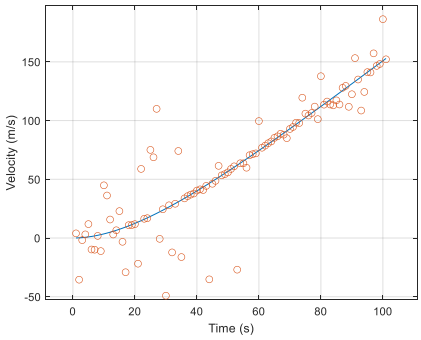


Fig. 1: Graph of velocity vs time with neural trained data using the LM Backpropagation Method.

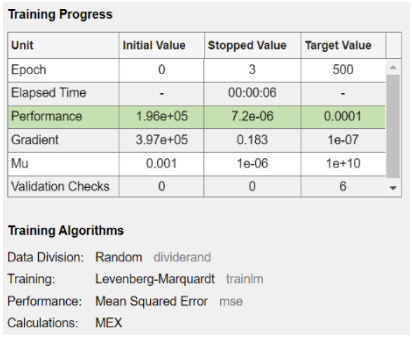


Fig. 2: The results table from the Levenberg-Marquardt

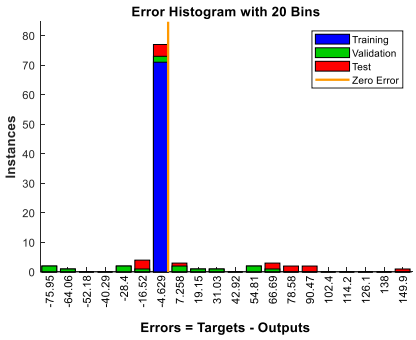


Fig. 3: The error histogram from the Levenberg-Marquardt Backpropagation

The histogram shows that the model error consolidated right next to the zero container. Since the error from the trained data is centered, it indicates that the neural network was properly trained. The generated output values are nearly identical to the ideal target values. When implementing a Neural Network, there are three different algorithms used to train the computer to the input data. The results shown in Fig. 2 and 3 are based on the Levenberg-Marquardt Backpropagation method; however, there are two other basic algorithms that can be used. The other two algorithms are called Bayesian and Resilient backpropagation. The Bayesian backpropagation method also uses the Levenberg-Marquardt algorithm; the main difference between the two methods is that the LM algorithm converges to a solution in a shorter time. Figure 4 shows a graph of the predicted car's velocity with the training data superimposed on top.

The results in Fig. 4 indicate that the Bayesian algorithm is a good model; however, when the Bayesian algorithm error bar histogram the algorithm does not look as promising.

The error histogram shown in Fig. 6 shows that the neural network model error is spread relatively evenly. Although most of the model's error is centered around the zero line, the error is more spread out than the LM backpropagation model; this indicates that the model is overfitted.

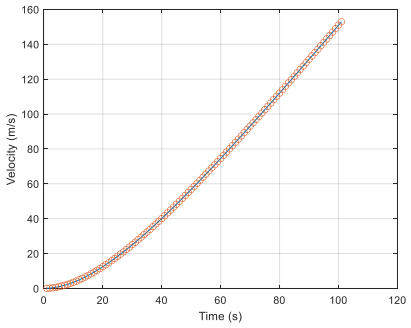


Figure 4: Graph of Velocity vs Time with Neural Trained Data using Bayesian regularization

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	1000	1000
Elapsed Time	-	00:00:18	-
Performance	2.01e+05	2.93e-07	0
Gradient	4.01e+05	0.0512	1e-07
Mu	0.005	500	1e+10
Effective # Param	301	41.5	0
Sum Squared P...	2.63e+06	3.55e+03	0

Training Algorithms

Data Division: Random dividerand
Training: Bayesian Regularization trainbr
Performance: Mean Squared Error mse
Calculations: MEX

Fig. 5: The Results table from the Levenberg-Marquardt

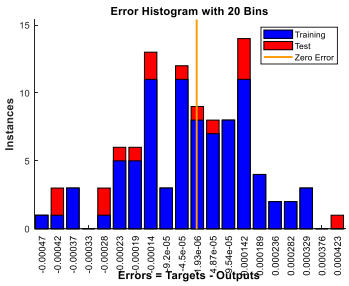


Fig. 6: The results table from the Levenberg-Marquardt Backpropagation.

The Resilient backpropagation model is based on the resilient backpropagation algorithm. Figure 7 shows the velocity curve with the trained data superimposed on top. Like the previous two models, the data does a good job of following the shape of the velocity curve. However, the error histogram, shown in Fig. 9, shows a wide spread of model error. When compared to the previous models, the Resilient backpropagation method is the worst option. The engineering team determined to stick with the LM backpropagation method.

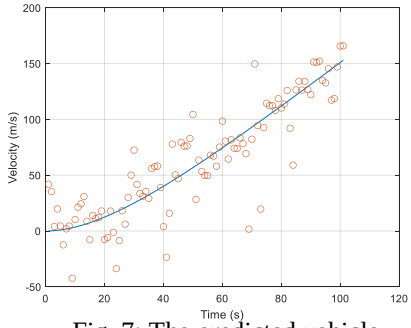


Fig. 7: The predicted vehicle velocity vs time using the Resilient Back Propagation Method.

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	22	1000
Elapsed Time	-	00:00:00	-
Performance	1.77e+05	327	0
Gradient	3.99e+05	2.68e+03	1e-05
Validation Checks	0	6	6

Training Algorithms

Data Division: Random dividerand
Training: RProp trainrp
Performance: Mean Squared Error mse
Calculations: MEX

Fig. 8: Data table using the Resilient Back Propagation

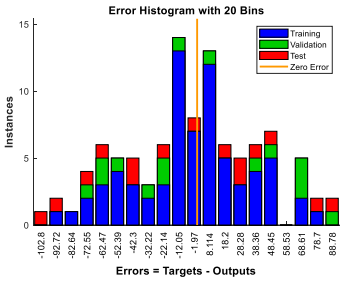


Fig. 9: The Error Histogram using the Resilient Back

Example 2: Determining Car Velocity Neural Network with Fuzzy Logic Model

To further understand the vehicle speed model, the team decided to compare the Artificial Neural Network Model with a fuzzy logic controller. The team generated the following response surface using the fuzzy logic toolbox in MATLAB.

The response surface looks significantly different from the Neural Network model. The flatness in the center of the response surface is due to the center response function. Table 1 shows a data summary of the two different modeling methods. The fuzzy logic model was more accurate than the neural network model at 20 seconds. However, after 20

seconds, the neural network model was a better indicator. The engineering team is confident that with further refinement of the response function, the fuzzy logic can become a better model for this particular problem than the neural network.

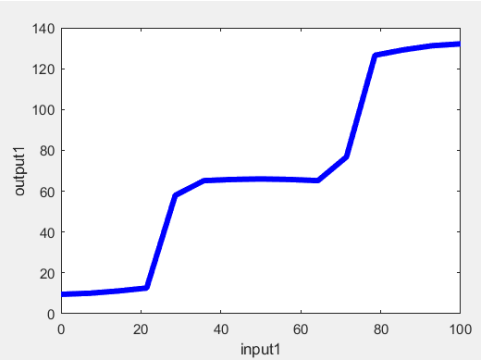


Fig. 10: Fuzzy Response Surface for Vehicle Speed.

Table 1: Compares the Function's velocity compared with the Neural Networks velocity and the Fuzzy Logic Controller's velocity.

Time (second)	Function / Velocity (m/s)	Neural Network / Velocity (m/s)	Fuzzy Logic / Velocity (m/s)
20	12.466	- 46.3947	12.2755
40	40.0948	40.0948	65.5431
60	74.536	73.9025	65.5431
80	112.038	104.568	127.549
100	150.916	166.726	132.161

Example 3: Modeling a mass spring system with a Neural Network

A third example of utilizing the neural network toolbox is a spring-mass system. The spring-mass model is another example of a traditional engineering problem that can be solved by the neural network toolbox. It is a typical problem; that requires knowledge of calculus, differential equations, and physics. However, the neural network can generate results without background knowledge. Based on the theoretical analysis, the mass-spring system is overdamped. The following system parameters are used in this model: mass m = 1 [kg], damping coefficient b = 20 [N/(m/s)], and spring stiffness coefficient k = 10 [N/m]. The initial conditions are assumed to be: the initial displacement x0 = 0 [m], and the initial velocity v0 = 1 [m/s]. Also, the driving force is assumed to be F = 0 [N]. The model is based on the equations below.

$$\frac{d^2x}{dt^2} * m + \frac{dx}{dt} * b + k * x = F \tag{3}$$

Applying MATLAB symbolic toolbox the solution to Eq. (3) is as follows

$$x(t) = \left(-1 - 6\sqrt{10}\right)e^{(-10-3\sqrt{10})*t} + (1 + 6\sqrt{10})e^{(-10+3\sqrt{10})t} \tag{4}$$

The solution, Eq. (4), indicates the mass-damper-spring system is overdamped.

The mass-damper-spring system can be solved by applying two approaches: (a) MATLAB Neural Network toolbox and (b) MATLAB/Simulink toolbox.

The system response to given initial conditions is shown in Fig. 11 – 15. Figure 11 shows the mass displacement versus time, and Figure 12 shows the force on the mass (damper and spring) versus time.

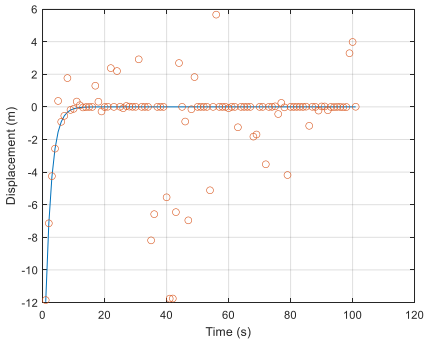


Fig. 11: Mass displacement vs Time with LM backpropagation.

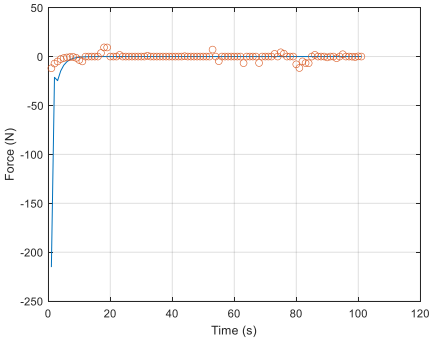


Fig. 12: Force vs time with LM back-propagation.

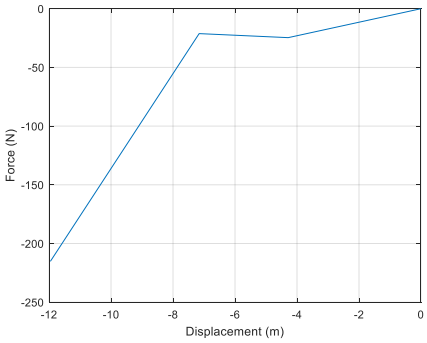


Fig. 13: The force on the mass vs the mass displacement.

The results, Fig. 11 and 12, indicate that the LM back propagation method performed well, fitting both the displacement and the force curves. The neural network model generated a result similar to the theoretical model. Both theoretical and neural network models, determined that the mass-spring system was over-damped. Also, the model error is concentrated around the zero line, as seen in Fig. 15. This indicates that the neural network data is a good model fit without being overfitted. Figure 13 shows the force on the mass versus the displacement. Like the previous figures, Fig. 13 shows that the system is overdamped. A Simulink model of the mass-spring system was created to further evaluate the ANN model results.

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	4	500
Elapsed Time	-	00:00:00	-
Performance	2.26e+05	5.36e-05	0.0001
Gradient	3.99e+05	0.58	1e-07
Mu	0.001	1e-07	1e+10
Validation Checks	0	2	6

Training Algorithms

Data Division:	Random	dividerand
Training:	Levenberg-Marquardt	trainlm
Performance:	Mean Squared Error	mse
Calculations:	MEX	

Fig. 14: Mass Displacement Model Table.

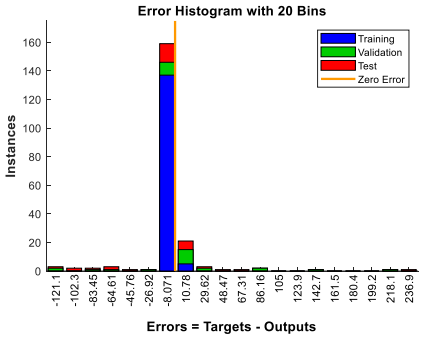


Fig. 15: Mass Displacement Model Error Histogram.

Example 4: Mass-damper-spring system - Neural Network with Simulink Model

The Simulink model of the mass-spring-damper system is shown in Fig. 16. The Simulink model shows similar results of displacement versus time as the ANN model. Like with the ANN model, the mass displacement converges to its final value around 10 seconds, Fig. 17. This indicates that the mass-damper-spring system is over damped.

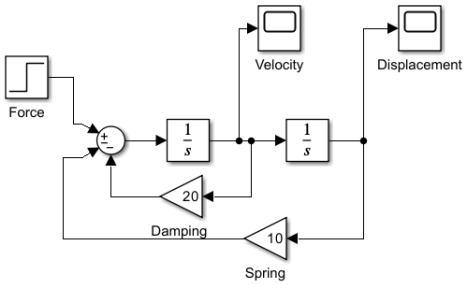


Fig. 16: Mass spring damper Simulink Model.

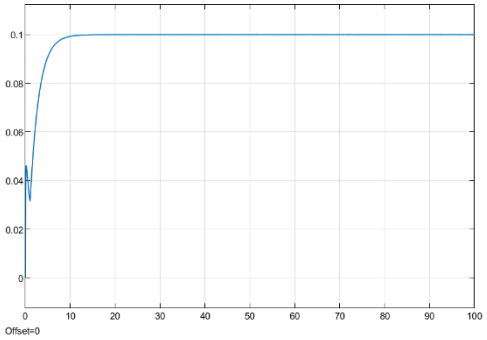


Fig. 17: Displacement vs time graph from Simulink Model.

Example 5: Modeling Evaluation Results with a Neural Network

While Neural Networks are beneficial for analyzing traditional engineering models, the real benefit is the ability to model complex systems. Especially, systems that cannot be easily represented by traditional mathematical/engineering techniques. Those require an advanced understanding of coding and mathematics to create a viable model. The neural network can reduce the burden. In example 5, a neural network was used to model teacher evaluation results. The structure of the model is visually represented in the diagram below, Fig. 18. The diagram shows 4 inputs with 100 hidden neurons, one output, and one activation point.

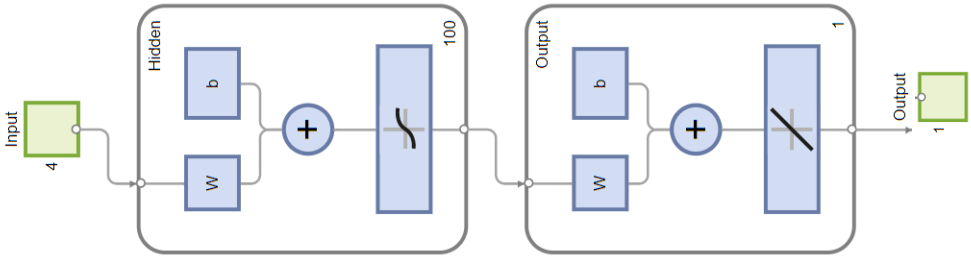


Fig. 18: The neural network diagram for the teacher evaluation.

Figure 19 shows the teacher evaluations fitted with the neural network model. The neural network was trained with the LM backpropagation algorithm. The model performed well, predicting the teacher evaluations, as indicated by the error histogram. The histogram shows that most of the model error is consolidated near the 0 line (Fig. 21). Other backpropagation methods were investigated; however, the other backpropagation methods resulted in a larger model error.

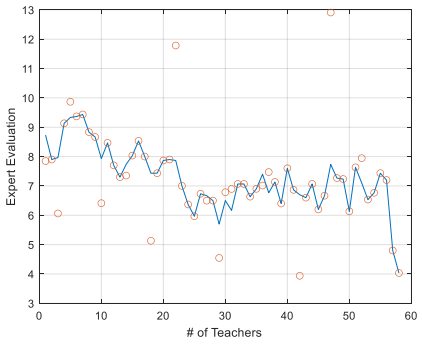
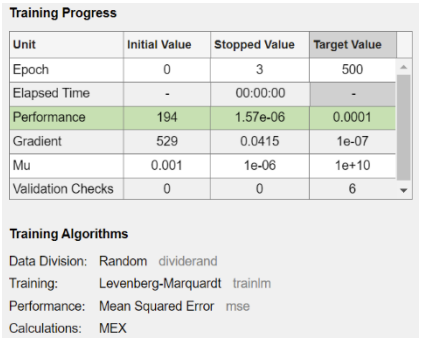


Fig. 19: Expert evaluation score vs number of Teachers.



Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	3	500
Elapsed Time	-	00:00:00	-
Performance	194	1.57e-06	0.0001
Gradient	529	0.0415	1e-07
Mu	0.001	1e-06	1e+10
Validation Checks	0	0	6

Training Algorithms

Data Division: Random dividerand
Training: Levenberg-Marquardt trainlm
Performance: Mean Squared Error mse
Calculations: MEX

Fig. 20: The neural network table.

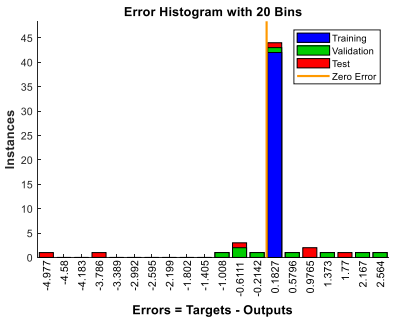


Fig. 21: The neural network error histogram.

Figure 22 shows the relationship between Peer Feedback and Student Scores. As you can see, there is a positive correlation between Peer Feedback and Studen Scores. The relationship is not linear, indicating there is a weak correlation. There is a similar relationship, Fig. 24, between Student Scores and Education Activity. A positive non-linear trend can be observed in this figure. The results indicate a weak correlation between the two variables. The relationship between teacher evaluations and student scores has the most linear relationship, which indicates the strongest correlation between the two categories. However, that is not surprising since Student Scores are a core metric for a teacher's effectiveness. Overall, the neural network was able to successfully model teacher evaluation scores.

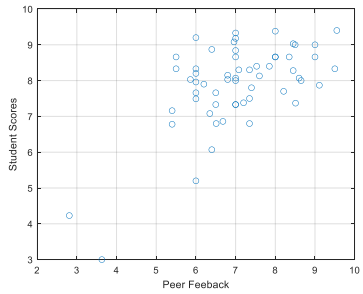


Figure 22: Student Score versus Peer Feedback.

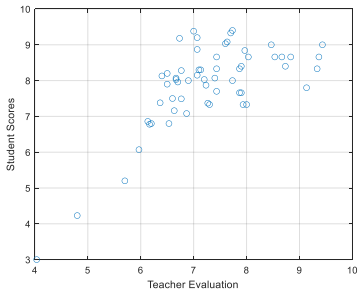


Figure 23: Student Score versus Teacher Evaluation Score.

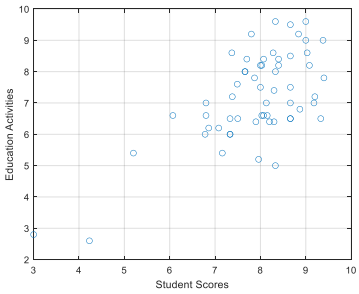


Figure 24: Student Score versus Education Activities.

Example 5: Modeling Temperature and Precipitation Amounts Using Neural Network

This example is another demonstration of the neural network's ability to trend data and predict data points for complex mathematical relationships. For this example, the Norfolk precipitation data from the NOAA website was modeled using a neural network. The model was based on six months of weather data in 2005 (January 2005 to June 2005).

The neural network does a decent job of fitting the precipitation data. The model was able to follow the general shape of the precipitation curve, as seen in Figure 25.

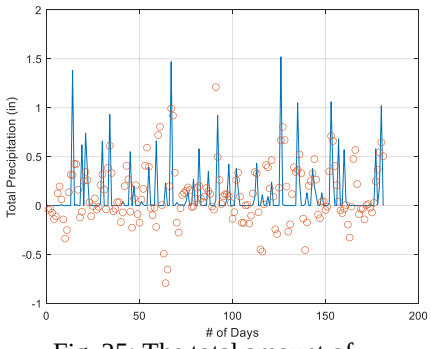


Fig. 25: The total amount of precipitation with the neural network data fit over top.

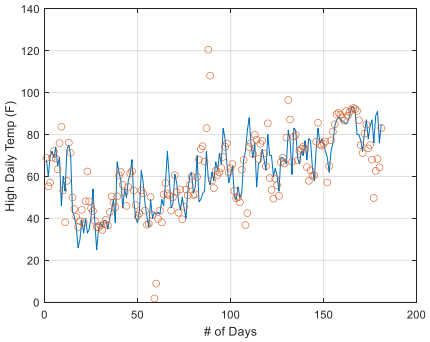


Fig. 26: Norfolk daily high temperatures with the neural network data fit over top.

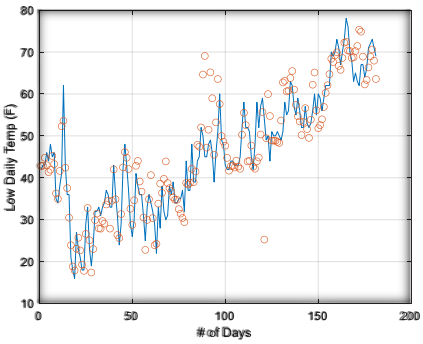


Fig. 27: Norfolk daily low temperatures with the neural network data fit over top.

The neural network model performed well in predicting the daily high and low temperatures, Fig. 26 and 27. The daily amount of predicted rain and snow in Norfolk are shown in Fig. 28 and Fig. 29. It can be observed, that the neural network model predicts the amount of rain reasonably well. However, the snow/sleet/hail predictions were not well modeled. Many predicted data points were not an accurate reflection of Norfolk's weather pattern. This issue is most likely related to the 6-month time frame of the data analysis. To test the neural network, a longer time frame will need to be observed. The neural network model does a significantly better job of predicting the temperature versus the amount of precipitation. However, there was more temperature data than precipitation data used to train the model. Furthermore, the temperature changes with the seasons, making it an easier trend to fit, whereas precipitation amounts do not correlate as well. Data selection is essential to ensure that the neural network creates an accurate model.

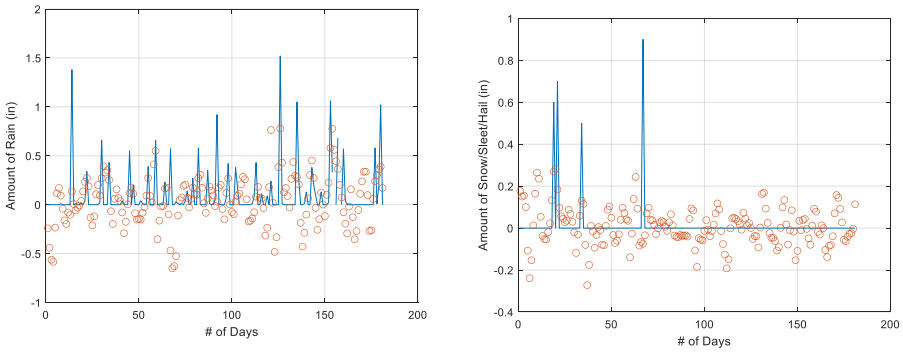


Fig. 28 and 29: The amount of rain and snow versus number of days.

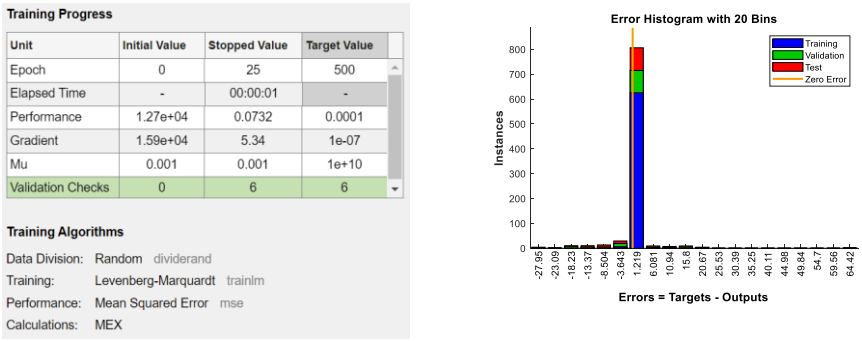


Fig. 29 and 30: The table and error bar histogram for model.

Like the previous models generated in this paper, the neural network model has a low model error. Nearly all the model error is concentrated around the zero error bar, since the targets and outputs are virtually the same. The propagation method could have been changed to get a better visual fit on Fig.s 25 - 29; that would increase the amount and spread of model error in the error histogram.

CONCLUSION

Artificial Neural Networks are a basic form of machine learning and deep computing. Neural networks use mathematical functions to fit or understand a series of data points. With this recent advancement in technology, engineers have the ability to solve complex engineering problems. Some of the areas that have been implementing ANNs include mechanical engineering design and manufacturing. The applications range from process control to predictive and health-based maintenance. The paper summarizes various methods and uses for this technology in different engineering examples. The introductory ANN material has been added to two graduate courses to expose students to modern tools and their potential applications in mechanical engineering. The graduate students

implemented their basic knowledge to real-world problems, e.g. quality/process control of weldments, quality/process control of additive manufacturing, etc. The results from these two courses encouraged students to further explore the theory and application of ANNs and provided stimulus to faculty to develop a graduate course on ANN in advanced manufacturing and engineering design.

REFERENCES

[1] https://www.ibm.com/topics/what-is-a-digital-twin
[2] Tao F., Zhang H., Liu A., Nee Y. C., "Digital twin in industry: State-of-the-art." IEEE trans. On Industrial Informatics, 15(4), April 2019.
[3] Chaturvedi D. K. "Modeling and Simulation of Systems Using MATLAB and Simulink." CRC Press, ISBN 978-1-4398-0672-2, 2010.
[4] Dieter G. E., Schmidt L. C. "Engineering Design." McGraw Hill, ISBN 978-0-07-339814-4, 2009.
[5] Stich T. J., Spoerre J. K., Velasco T., "The application of artificial neural networks to monitoring and control of an induction hardening process." J. of Industrial Technology, 16(1), November-January, 2000.
[6] Saeed M. M., Al Sarraf Z. S., " Using Artificial Neural Networks to predict the effect of input parameters on weld bead geometry for SAW process." Journal European des Systemes Automatises, pp. 309-315, 54(2), April 2021.
[7] Bon A. T., Hui H. S. "Industrial engineering solution in the industry: Artificial Neural Network forecasting approach." Proc. Int. Conf. on Industrial Engineering and Operations Management, Rabat, Morroco, April 11-13, 2017.https://www.ncdc.noaa.gov/cdo-web/datasets/GHCND/stations/GHCND:USW00013737/detail