Proceedings of Meetings on Acoustics

Volume 36 http://acousticalsociety.org/

177th Meeting of the Acoustical Society of America

Louisville, Kentucky

13-17 May 2019

Underwater Acoustics: Paper 5aUW4

A feedforward neural network for source range and ocean seabed classification using time-domain features

David F. Van Komen
Department of Physics & Astronomy, Brigham Young University, Provo, UT, 84602; david.vankomen@byu.edu; david.vankomen@gmail.com

Tracianne B. Neilsen
Brigham Young University, Provo, UT 84602; tbn@byu.edu

David P. Knobles
KSA, LLC., Austin, TX; dpknobles@kphysics.org

Mohsen Badiey
University of Delaware, Newark, DE; badiey@udel.edu

Acoustic source ranging in an uncertain ocean environment is a complicated problem, though classification and regression-based machine learning algorithms show promise. A feedforward neural network (FNN) has been trained to do either classification or regression on both the source-receiver range and ocean seabed type using extracted time-domain features. Pressure time-series are generated to simulate signals received at different ranges in three different ocean environments, representing sandy, muddy, and mixed sediment seabeds. Four features are extracted from these waveforms: peak level, integrated level, signal length, and decay time. These four features are used to train FNN for both classification and regression of range and environment type, and the results are compared to a network trained on the time waveforms. Even for small amounts of training data, the pressure time-series provide a higher accuracy than the extracted features. These results lay a foundation for comparisons to the more computationally expensive convolutional neural networks.

Published by the Acoustical Society of America

1. INTRODUCTION

In ocean acoustics research, localizing acoustic sources and estimating the ocean environment present many challenges. Some challenges of these inverse problems include nonlinear relationships between the unknowns, high-dimensional search spaces, and large uncertainty due to ill-conditioning of the inferred parameters. Solutions can be found using optimization techniques, like matched field processing,[1] but these can be limited by the accuracy of the environmental information. Beyond optimization techniques, machine learning and deep learning are becoming increasingly of interest for ocean acoustics problems due to their ability to extract features and patterns from data.

Recent efforts have used machine learning to estimate source-receiver ranges. Steinberg *et al.*[2] introduced the idea of using neural network techniques to localize an acoustic point source in a homogeneous medium using single- and two-layer neural networks. Hougnigan *et al.*[3] used supervised machine learning for real-time range estimation and obtained an average error of 4.3%. Lefort *et al.*[4] and Niu *et al.*[5,6] found that machine learning classifiers outperform traditional methods such as inversion and matched field processing. Others, such as Huang *et al.*[7] and Wang *et al.*[8] have had similar successes with using machine learning techniques to localize sources in the ocean. Machine learning has also been used for classification of underwater targets as done by Fischell *et al.*[9] Seabed parameters, such as attenuation and sound speed, have also been estimated with machine learning in the work done by Piccolo *et al.*[10] on features extracted from noisy and clean data.

The goal of this paper is to illustrate the potential of machine learning models for predicting environmental information from extracted signal features or from full time-domain signals. Specifically, the experiment employs a feedforward neural network on simulated acoustic signals to predict a range and environment class simultaneously.

2. BACKGROUND

A. UNDERWATER ACOUSTICS AND SUS CHARGES

To generate the data used in this experiment, the signals underwater sound (SUS) charge spectra were simulated through three different environments representing sandy, muddy, and mixed sediment environments. The SUS charge simulations follow the model by Chapman[11] as explained in Sen *et al.*[12] for a 0.034 kg charge in all cases. The parameterization of these three environments is displayed in Figure 1. These figures show a vertical "slice" from the surface of the ocean to the sediment layers that constitute the ocean floor. The water column sound speed profile, typical of a shallow ocean, is shown by the black line, with constant attenuation and density listed in the legend with the subscript w. Each sediment layer l is parameterized by it's thickness h_l in m, compressional sound speed c_l at the top and bottom of the layer in m/s, density ρ_l at the top and bottom in g/cm^3, and compressional attenuation α_l at the top and bottom in dB/λ. The bottom boundary, or half-space, is defined by the compressional sound speed, attenuation and density. These parameters are fed into a forward model to simulate the response of the ocean.

The response of the ocean environment is simulated with ORCA.[13] The ORCA model calculates normal mode functions in range-independent environments by calculating upward- and downward-reflecting plane wave reflections. The mode functions can then be used to generate an impulse response of the ocean at a certain range from the source. This impulse response is convolved with the simulated SUS charge at a depth $z = 18.3$ m, to produce the time-series signal used in this study.

B. MACHINE LEARNING METHODS

Machine learning is a data analysis method that builds models dynamically by identifying patterns. Machine learning has exploded in recent years with new models and methods being published regularly.

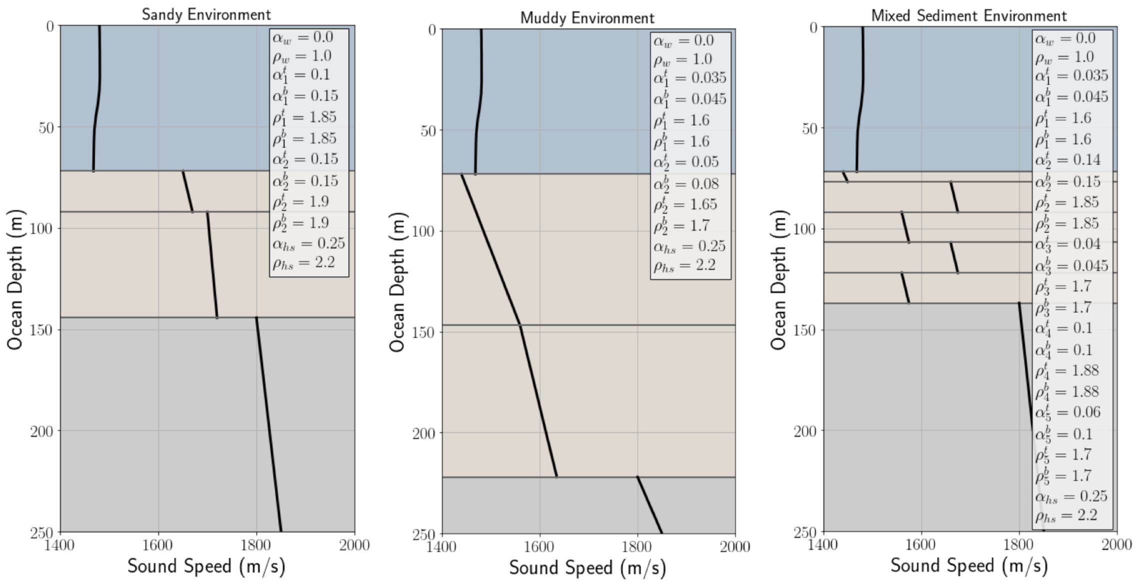


Figure 1: The parameterization of the sandy (left), muddy (center), and mixed sediment (right) environments. The blue color represents the water layer, the light brown indicates the sediment layers, and the gray indicates the half-space boundary. The figures also contain a list of the various parameters at each of the layer boundaries: α is the attenuation in dB/λ and ρ is the density in g/cm^3, while the subscript denotes the layer identifier (w for water, hs for half-space, number for sediment layers) and the superscript denotes the top (t) or the bottom (b) of the layer.

Instead of venturing into the latest and greatest models, this paper explores the application of deep learning to source localization and environmental classification using a feedforward neural network (FNN) model.[14]

The FNN is an artificial neural network that, in its simplest form, takes inputs and maps them to outputs through learned transformations. The "neurons", also referred to as nodes, contain a vector of values. The neurons in the network are ordered into layers, where the first layer is the input layer, the last layer is the output layer, and all layers between are hidden layers. Each neuron in each layer is "fully connected" to every previous neuron in the previous layer, and the value of a neuron j at the next layer can be expressed by O_j:

$$O_j = f(\text{net}_j), \tag{1}$$

$$\text{net}_j = b_j + \sum_{i=1}^{\text{neurons}} w_i x_i \tag{2}$$

where w_i is the weight applied to the previous neuron output x_i, b_j is a bias term unique to each new neuron, and $f(\text{net}_j)$ is an activation function for layer j chosen by the network designer (such as a sigmoid function, the rectified linear unit (ReLU) function,[15] or the softmax function). The weight values w_i and bias values b_j are dynamically learned by the network through many iterative steps by calculating the error of the predictions against the true output and running a gradient descent through the network. Hyperparameters are set by the network designer to control the behavior of learning, the number of neurons in each layer, and the type of activation function used. The mathematics behind gradient descent are beyond the scope of this paper, and there are many tools available to automate the process, such as the PyTorch[16] implementation of the Adam Optimizer.[17]

For this paper, FNNs were selected due to their simplicity and past successes. Their simplicity makes this study a convenient benchmark for other network types in the future as they strive to localize acoustic sources and extract environmental information.

3. METHOD

Development of the FNN approach to predicting environmental information as well as source range requires training data. This section illustrates how the data were generated, how specific features summarizing the data were selected, and how the network was designed for this problem.

A. DATA SIMULATION AND FEATURE EXTRACTION

For this experiment, data were generated with SUS[11] charges in environments simulated by the range-independent ocean model ORCA.[13] The three different environments (sandy, muddy, and mixed sediment) are shown in Figure 1. There were also five different sound speed profiles used to provide variability and expand the dataset. The charges were simulated at nine different ranges (ranging from 0.5 to 11.5 km) away from a receiver. The signals are approximately three seconds each consisting of approximately 5,000 discrete pressures. The three second window was chosen to encompass the majority of the received signal no matter the range while reducing the number of points. To illustrate the variability in the simulated data, Figure 2 shows two example signals from the sandy and mixed environments, where one is at the closest range of 0.5 km and the other is at the farthest range of 11.5 km. Figure 2 also shows that the signals fit within three seconds, demonstrating that the full data do not contain the absolute travel time from the explosion to the sensor.

To reduce the total number of inputs, four specific features were hand-selected to represent each of the signals. The first feature was peak level (shown by the red arrow in Fig. 2b), which is simply the highest recorded pressure in the signal. The second was integrated intensity calculated across the three seconds. The third was the total time of the high-intensity explosion (the dashed green line in Fig. 2b). The fourth and final feature was the decay time (shown by the yellow solid line in Fig. 2b) which was the time taken until the pressure reached a specified threshold.

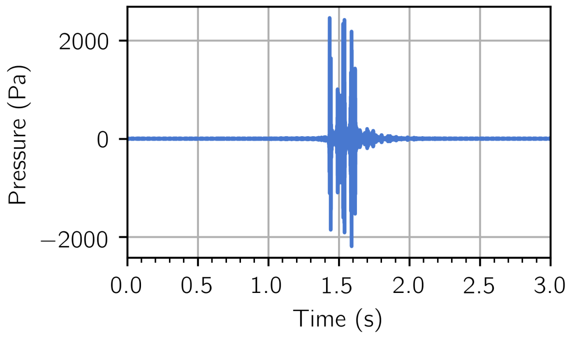
The combination of three environments, seven source ranges, and five sound speed profiles produces a dataset of 135 samples. The authors understand that 135 samples is a very small number of training samples for FNNs and other machine learning models. The number used in this study was limited by the time required to hand-extract some of the features described above. Future work in applying deep learning algorithms to source localization and environmental classification will use many more samples for training and testing to take full advantage of the potential for a neural network to learn a more generalized view of the data. Addition of noise to the simulated data was also not investigated as part of this study, but is being considered for future ways to improve training networks with synthetic data.

B. FEEDFORWARD NEURAL NETWORK TOPOLOGY AND HYPERPARAMETERS

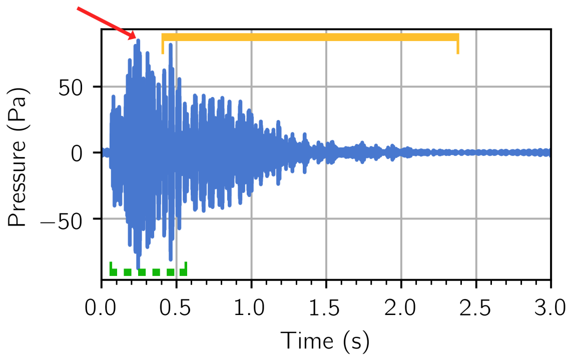
The FNN used to predict environment and source range was built in Python with the PyTorch[16] package. PyTorch[*] is an open source deep learning platform that uses native Python syntax to quickly prototype, train, and test networks of any configuration. PyTorch also automates the required algorithms to perform gradient descent and other learning algorithms making it ideal for proof-of-concept and production work.

The network structure used to test the ability of a FNN classify range and environment type is listed in Table 1. The hidden linear layer can be repeated as many times as desired to create a multi-layered network. This paper shows results from one and two hidden layers with 250 neurons each. The input layer takes either the four extracted features as detailed in Section 3.A or time series waveforms depending on the test. The

[*]More information about how to use PyTorch can be found on their website at `https://pytorch.org/`.



(a) A simulated received SUS signal at a sensor 0.5 km away in the sandy environment.



(b) A simulated received SUS signal at a sensor 11.5 km away in the muddy environment.

Figure 2: Example simulated SUS signals. Three of the four extracted features are displayed in Figure 2b: the red arrow denotes the peak level, the dashed green line denotes the total time, and the yellow line denotes decay time.

Table 1: Details of the FNN. The different operations performed with the network are listed, including the input and output sizes of the linear operations. The type of activation function used at each layer after the linear operation is also reported. The "Hidden to Hidden" layer is optional or repeatable depending on the network designer's needs.

Layer Name	**Input Size**	**Output Size**	**Activation**
Input to Hidden	5,000 or 4	250	ReLU
Hidden to Hidden	250	250	ReLU
Hidden to Output	250	27	Softmax

final layer outputs a probability distribution across 27 values each corresponding to the unique combination of nine ranges and three environments used in this study.

One fundamental question to address is how many hidden neurons should be used in the network. The machine learning community has not found a one-size-fits-all answer for determining the number of neurons in a hidden layer. There appear to be two schools of thought for picking the number of neurons: decreasing dimensionality and increasing dimensionality. A decrease in dimensionality keeps the network size small but could lead to overgeneralization and not getting enough information. An increase in dimensionality increases the network size which increases the computation time but has the benefit of allowing the network to learn to ignore certain connections. This paper utilizes increasing dimensionality with the extracted features and decreasing dimensionality with the full signals in an attempt to compare identical networks. Due to the preliminary nature of this paper and multiple tests leading to the selected hyperparameters, the authors believe that the network sizes are acceptable. If FNNs are to be used in future research, this question should be more fully addressed.

The network was trained with the Adam optimizer[17] using a learning rate of 0.001 that annealed via a cosine function over 1000 epochs. The loss function used was cross entropy,[18] which is useful for classification. The learning rate was annealed to allow the network to approach the optimal weights early in training and then refine them later in training. The 135 data samples were split into training and testing datasets with a random 75/25 split (101 training samples, and 34 testing samples randomly divided each training instance). The results of training this network are shown with the 34 testing samples that were not used during training.

The data were also normalized before being input into the network. The selected normalization depended on the type of input data. The normalization applied to the extracted features is detailed in Section 4.A, and the normalization for the full wave forms is detailed in Section 4.B. Normalization is encouraged in machine learning models to avoid the need for large weights, which would require a longer time to learn.

4. RESULTS

The results have been divided into two sections. The first is to show how the network learns from the four, extracted, time-domain features. The second is to show the ability of the network to learn directly from the simulated time series.

A. RESULTS USING ONLY EXTRACTED FEATURES

This section shows the results of the FNN trained on four extracted features. Before the data were passed to the network, each feature was normalized by the maximum value across all samples of that particular feature to keep the weight values low. Two different networks were used: one with one hidden layer and another with two hidden layers.

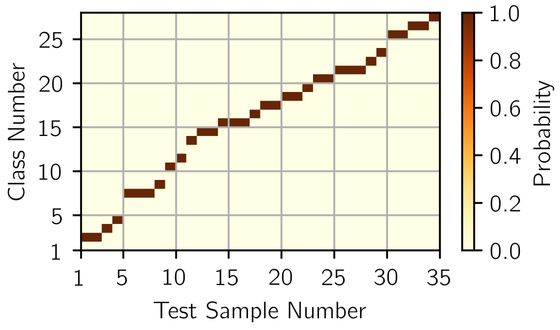
When a new sample is passed through the FNN trained for classification, a probability distribution is returned. The network predicts which classes are more probable than others. There are 27 separate class numbers that correspond to a unique combination of range and environment. These probability distributions are shown in Figure 3 for an FNN with one hidden layer on the top and with two hidden layers on the bottom. The x-axis corresponds to the test sample number, which were different random draws for the two cases shown. The plots on the left correspond to the true class distribution that the network, while the plots on the right correspond to the FNN predicted distribution.

Not every combination of range and environment is represented in these figures due to the random draw. A random draw, in this case, was useful to quickly determine the network's potential for learning as well as seeing the network's potential for generalizing in the cases where more than one instance of a particular class is in the testing dataset. As mentioned earlier, there are only 135 total samples in this preliminary dataset, which is not sufficient for a machine learning model to generalize for all cases. To counter this, the networks were trained with different random splits across six different runs to obtain an average percentage. While there could have been a hold-out set of one from each environment, a random draw seemed more intuitive for these preliminary results.

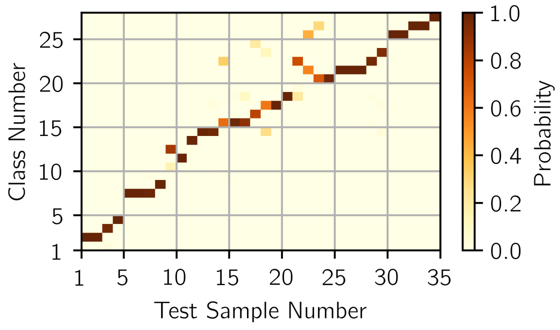
The class number with the highest probability in each case is used to get a measure of accuracy. To get a better measure of accuracy, six different networks were trained with a different train-test split in each instance. The accuracy of the single hidden layer network in one instance (as in Fig. 3b) was 91.1%, while the average accuracy across six different tests is 86.3%. On the other hand, the two hidden layer network in one instance (as in Fig. 3d) had an accuracy of 94.1%, while the average accuracy was 91.1% across six different tests. The FNN can classify the unique combinations of range and environment when using hand-selected features.

B. RESULTS USING THE FULL WAVEFORM

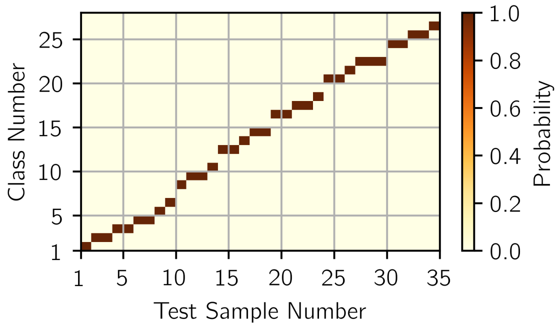
Building on the results from the extracted features, the FNN are now applied to the entire pressure time series. In this case, each input sample has approximately 5,000 input features, each containing a value of the simulated signal. The simulated signals are normalized by the absolute maximum of the entire dataset. Again, the normalization is done to help the network learn faster and keep smaller weight values.



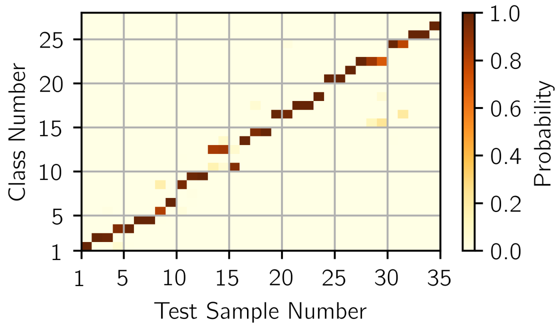
(a) True class probability for each of the test samples with one hidden layer.



(b) Predicted class probability for each of the test samples with one hidden layer.

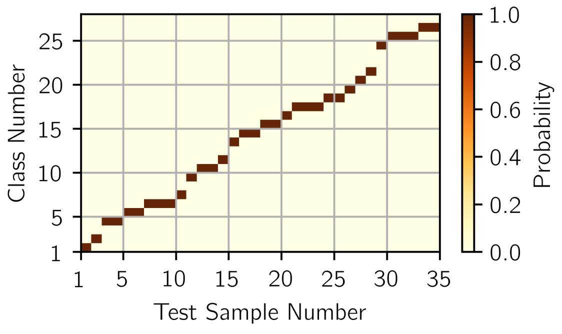


(c) True class probability for each of the test samples with two hidden layers.

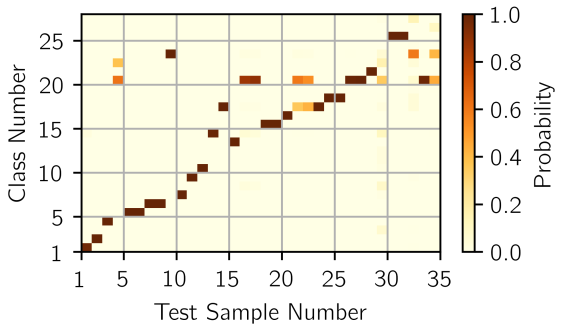


(d) Predicted class probability for each of the test samples with two hidden layers.

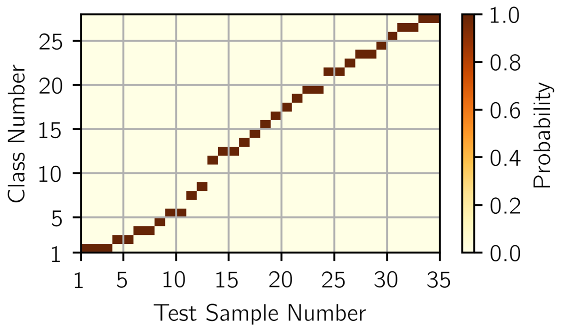
Figure 3: Probability distribution results of training a feedforward neural network by using only extracted features (peak level, integrated intensity, total time, and decay time) on one random split. These figures show two different cases: Figures 3a and 3b refer to a FNN with one hidden layer while Figures 3c and 3d refer to a FNN with two hidden layers. Comparison of the predicted class with the highest probability to the actual class shows that in these particular random training and testing split, the FNN with one hidden layer is 91.1% accurate, while the FNN with two hidden layers is 94.1% accurate.



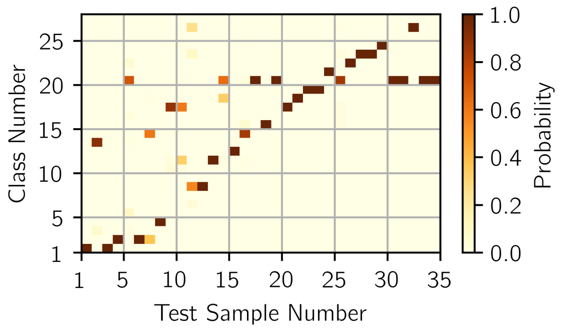
(a) True class probability for each of test samples with one hidden layer.



(b) Predicted class probability for each of test samples with one hidden layer.



(c) True class probability for each of test samples with two hidden layers.



(d) Predicted class probability for each of test samples with two hidden layers.

Figure 4: Probability distribution results of training a feedforward neural network by using the full waveforms. These figures show two different cases: Figures 4a and 4b refer to a network with one hidden layer while Figures 4c and 4d refer to a network with two hidden layers. When asked to choose the class with the highest probability, the network was 61.7% accurate with one hidden layer and 52.9% accurate with two hidden layers in these particular random training and testing splits.

The motivation behind using the whole waveform is to reduce the amount of time that feature extraction requires.

The FNNs with one and two hidden layers were trained on 101 waveforms and then tested on the remaining 34. The resulting probability distributions for the predicted classes of the testing samples for one instance of the random training and testing split are shown in Figure 4. Again, the left figures correspond to the actual class distribution while the right figures show the probability distribution predictions of the FNN on the same samples.

The network trained on the time series does significantly worse than the network trained on extracted features. The predicted class with the highest probability is used to describe the accuracy of the predictions. In the case of one hidden layer, a single test could produce an accuracy of 61.7% (as in Fig. 4b), while there was an average of 63.2% accuracy across six tests. With two hidden layers, a single test could produce an accuracy of 52.9%, while the average accuracy across six tests increases to 67.2%. Full waveforms may contain more information than what is represented by extracted features, but a FNN does not have the correct structure to make accurate predictions on the waveforms.

There is also a significant increase in computational load when using the full waveforms. Using the extracted features in the one-hidden-layer FNN took approximately 25 seconds on a personal laptop to train over 1000 epochs. With the same network structure, using full waveforms took approximately two minutes and 40 seconds. In the case of two hidden layers, the features took approximately one minute while the waveforms took three minutes to train.

5. CONCLUSION

This paper has shown that FNNs have the potential to learn source range and environmental classification simultaneously. The FNN applied to four extracted features from waveforms performed better than on an FNN applied to the waveforms themselves. This is because the FNN takes into account the exact positioning of input "features". This positioning works well for a series of features that are always in the same position, but does not work on a signal that could vary significantly from point to point. The successes of the FNN on these simulated data also illustrates the potential for deeper and more complicated networks, such as convolutional neural networks[19] and recurrent neural networks (such as long short-term memory networks[20]) which have shown success in image and speech recognition due to the spatial and temporal relationships within in the data. These more advanced networks tend to do their own feature extraction, which would remove the time needed to extract features by hand. Future work seeks to expand on these preliminary results by increasing the amount of data, developing deeper and more advanced networks, and by testing the networks on real data.

ACKNOWLEDGMENTS

This work was funded in part by the Office of Naval Research, SBIR Grant #N68335-18-C-0806 and #N00014-19-C-20001.

REFERENCES

[1] A. Tolstoy, *Matched Field Processing for Underwater Acoustics* (World Scientific, 1993).

[2] B. Z. Steinberg, M. J. Beran, S. H. Chin, and J. H. Howard Jr, "A neural network approach to source localization," J. Acoust. Soc. Am. **90**(4), 2081–2090 (1991).

[3] L. Houégnigan, P. Safari, C. Nadeu, M. van der Schaar, and M. André, "A novel approach to real-time range estimation of underwater acoustic sources using supervised machine learning," in *OCEANS 2017-Aberdeen*, IEEE (2017), pp. 1–5.

[4] R. Lefort, G. Real, and A. Drémeau, "Direct regressions for underwater acoustic source localization in fluctuating oceans," J. Appl. Acoust. **116**, 303–310 (2017).

[5] H. Niu, E. Ozanich, and P. Gerstoft, "Ship localization in Santa Barbara Channel using machine learning classifiers," J. Acoust. Soc. Am. **142**(5), EL455–EL460 (2017).

[6] H. Niu, E. Reeves, and P. Gerstoft, "Source localization in an ocean waveguide using supervised machine learning," J. Acous. Soc. Am. **142**(3), 1176–1188 (2017).

[7] Z. Huang, J. Xu, Z. Gong, H. Wang, and Y. Yan, "Source localization using deep neural networks in a shallow water environment," J. Acoust. Soc. Am. **143**(5), 2922–2932 (2018).

[8] Y. Wang and H. Peng, “Underwater acoustic source localization using generalized regression neural network,” J. Acoust. Soc. Am. **143**(4), 2321–2331 (2018).

[9] E. M. Fischell and H. Schmidt, “Classification of underwater targets from autonomous underwater vehicle sampled bistatic acoustic scattered fields,” J. Acoust. Soc. Am. **138**(6), 3773–3784 (2015).

[10] J. Piccolo, G. Haramuniz, and Z.-H. Michalopoulou, “Geoacoustic inversion with generalized additive models,” J. Acoust. Soc. Am. **145**(6), EL463–EL468 (2019).

[11] N. R. Chapman, “Source levels of shallow explosive charges,” J. Acoust. Soc. Am. **84**(2), 697–702 (1988).

[12] M. K. Sen, L. N. Frazer, S. Mallick, and N. R. Chapman, “Analysis of multipath sound propagation in the ocean near 49° N, 128° W,” J. Acoust. Soc. Am. **83**(2), 588–597 (1988).

[13] E. K. Westwood, C. T. Tindle, and N. R. Chapman, “A normal mode model for acousto-elastic ocean environments,” J. Acoust. Soc. Am. **100**(6), 3631–3645 (1996).

[14] T. D. Sanger, “Optimal unsupervised learning in a single-layer linear feedforward neural network,” Neural Netw. **2**(6), 459–473 (1989).

[15] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (2010), pp. 807–814.

[16] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in PyTorch,” in *NIPS-W* (2017).

[17] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980 (2014).

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT press, 2016).

[19] Y. LeCun and Y. Bengio, “Convolutional networks for images, speech, and time series,” The Handbook of Brain Theory and Neural Networks **3361**(10), 1995 (1995).

[20] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation **9**(8), 1735–1780 (1997).