# Deep Learning for Source Localization in a Laboratory Tank

Corey Emerson Dobbs
*Brigham Young University*

# Deep Learning for Source Localization in a Laboratory Tank

Corey Emerson Dobbs

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Physics and Astronomy

Brigham Young University

ABSTRACT

Deep Learning for Source Localization in a Laboratory Tank

Corey Emerson Dobbs
Department of Physics and Astronomy, BYU
Master of Science

Deep learning has been applied to underwater acoustics problems in many forms. One difficulty in applying deep learning techniques to ocean acoustics is the spatially and temporally varying environmental properties. Another challenge is the lack of labeled data for training large networks. The overall goal of this work is to develop deep learning approaches for source localization that can be adaptable to different conditions. In this work, a convolutional neural network was trained on acoustic data measured in a water tank, while the water was at room temperature, to predict source-receiver range. Tests were done to ascertain ideal quantities of training data for the task at different frequency bands. For all test cases, a dataset size of 1,806 data samples was found to be sufficiently large. In addition, comparisons between regression and classification models were made. It was found that regression models performed better than their classification counterparts when tested for generalizability. However, in all cases, the trained neural networks failed to generalize when the appropriate environmental variability was not included in the training data. To improve network performance, transfer learning can modify a pre-trained network to make predictions on data that was recorded under different conditions than the original dataset. Transfer learning was used to update the pre-trained model with a smaller set of data measured at different water temperatures. The resulting model better generalizes to measurements at different temperatures.

Keywords: source localization, deep learning, convolutional neural network, transfer learning, underwater acoustics

## ACKNOWLEDGMENTS

Much of my success as a student is due to the efforts of Dr. Traci Neilsen. Since I started with her as an undergraduate researcher, throughout my master's degree, she has been a phenomenal mentor. I am grateful for the opportunities that she has given me to grow as a researcher, scientist, writer, and person. I was able to attend conferences and workshops that helped me improve my public speaking and networking skills. I was also given opportunities to solve problems on my own and mentor new students in the lab. After my master's degree, I feel infinitely more prepared to go on with my career.

I would also like to thank all of my fellow student researchers in the Hydroacoustics Lab, both current and past. Natalie Bickmore, Madilyn Moore, and Ben White, for helping collect much of the data used in this thesis, and for being great labmates. Cameron Vongsawad, for his mentorship when I was an undergraduate student. Kaylyn Terry, Scott Hollingsworth, and Alexandra Hopps, for being good friends and classmates from whom I learned a lot.

One of the best decisions of my college career was deciding on a Physics Major. The BYU Physics Department, including the professors of all my classes over the last six years, have helped me develop a love for the field. I would not have made it through my degrees without the help and care of the faculty or my many friends who helped me get through classes, including Jay Spendlove, Joey Ehlert, Bethany Wu, Rylee Russell, and Carson Cunningham.

Most importantly, I would not be where or who I am today without the support of my wife, Renae. She is a great support to me at home and in my career, and I'm grateful for everything she has done for me.

Finally, I want to acknowledge and thank the Office of Naval Research for funding this research, under ONR Grant # N00014-22-12402.

# Contents

# Chapter 1

# Introduction

Ocean sound propagation is complex and has applications in many fields, including oceanography, marine biology, undersea warfare, geology, meteorology, and climate science. These fields use various forms of SONAR, or Sound Navigation and Ranging, to probe and learn about the ocean environment. Two types of SONAR exist: active and passive. Active SONAR relies on listening to echoes from pings sent into the water to obtain information about the surrounding area. By contrast, passive SONAR utilizes sound already present in the ocean. While passive SONAR is more difficult than active, it has the advantages of increased stealth and decreased noise pollution in the ocean [1].

Some of the challenges of passive SONAR result from the complex environment that exists in the ocean. First, the ocean water sound speed profile (SSP) is depth-dependent because it depends on the water salinity and temperature and the hydrostatic (over-burden) pressure, which vary with depth. Gradients in the SSP cause refraction as sound propagates, dictated by Snell's law. Additionally, differences in seabed layering and composition affect the magnitude, phase, and direction of reflections off the ocean floor. All these factors become more difficult to model when one considers the spatial and temporal variability of the ocean due to location, season, currents, etc. This variability causes difficulties in developing robust machine learning approaches to problems of source ranging.

The measurements tank in the Underwater Acoustics Lab at BYU represents a controlled, easily accessible environment for studying the impact of environmental variability on sound propagation underwater. The goal of this project is to use the measurements tank to test machine learning algorithms for robustness and generalizability in the presence of uncertainty in the SSP. In addition, transfer learning is used to improve the performance of a network–trained for source localization on a large dataset with one SSP–on data obtain with different SSPs using a small data set. This work provides an example of the potential for transfer learning to adapt machine learning to different environmental conditions more efficiently than training new models for each environment.

## 1.1 Machine Learning

Due to developments in technology and methods in the past decade, machine learning (ML) [2] has become a valuable tool for many complex problems in physics and acoustics [3]. The use of machine learning algorithms has proven to be useful for analyzing systems with many parameters and recognizing patterns that are too difficult for humans to see.

Machine learning is a term for systems that can autonomously learn without being explicitly programmed. Machine learning is a type of artificial intelligence, which refers to any sort of method by which computers can perform tasks that would normally require human intelligence. A specific subset of machine learning includes deep learning algorithms. Deep learning differs from other methods of machine learning by employing deep neural networks that require large amounts of data and much less human intervention. In addition, the neural networks used in deep learning typically have more layers than other neural networks. [4]

Many deep learning neural networks can be classified as convolutional neural networks, or CNNs. A CNN's distinguishing feature is the use of convolutional layers, which are particularly good for helping computers identify features in images [5, 6]. The CNN has been the cause of many recent advances in computer vision in the past decade. While most CNNs are designed to deal with 2-D images, applications to 1-D data are also possible. As described in a survey by Kiranyaz, *et al.* [7], 1-D CNNs have diverse applications, including automatic speech recognition, real-time electrocardiogram monitoring, and vibration-based structural damage detection. 1-D CNNs also can be used in various fields in acoustics [8].

## 1.2 ML in Ocean Acoustics

Some recent ML applications in underwater acoustics include predicting uncertainty in transmission loss calculations due to uncertainty in the measured environmental characteristics [9], enhancing underwater autonomous vehicle performance [10–13], locating sound sources, and characterizing the ocean environment. Because the last two topics are connected to this thesis work, a brief literature review of each is provided.

A common application of machine learning to ocean data is to identify an acoustic source's range, or distance between the source and receiver microphone, and the signal source's depth below the ocean surface. As surveyed in Grumiaux, *et al.*, [14], deep learning has been applied to sound source localization problems in many different acoustical contexts. In the last 10 years, much work has been done with training neural networks to localize a sound source [15–17] and predict source range [18–20] in both deep [21, 22] and shallow ocean environments [23–25]. While many previous projects have trained neural networks on pressure data, some have used vector sensor data instead without loss of performance [24]. This research typically has treated the source localization problem as a regression problem and given accurate results for source range and depth [26]. However, efforts have also been made to frame the problem as a classifier, with mixed results [15, 27–29]. In addition, work has been done to give physical meaning to what the neural network is learning by examining the activation maps, [30] which could further increase the usefulness of neural networks in learning about the ocean environment. Many source localization or identification papers with machine learning have used CNNs as the neural network of choice [31–33].

Machine learning is also prevalent in work to characterize properties of the ocean floor. These efforts often use ambient ship-radiated noise as a sound source [34]. An alternate approach is to deploy SUS explosive charges and listen to the response. Significant work has been done using both ship-radiated noise and SUS charges in the New England Mud Patch (NEMP) as training data for deep learning models. The problem has been considered as one of classifying seabeds into different predesignated seabed types [20, 33, 35], as well as one of regression that inverts for values of different seabed properties [25, 36–38]. Much of this work also predicts properties of the source ship such as range and speed.

## 1.3 ML in Water Tanks

As described above, most of the ML for source ranging has been conducted on ocean data. Limited research has been conducted to do ML source ranging in laboratory water tanks. One significant example is by Yangzhou *et al.* [39], who trained a CNN to localize a source in two dimensions based on data from a four-channel hydrophone array in a shallow tank. The only variation in the dataset resulted from moving the source signal to different locations on a two-dimensional plane. Another example is found in Lefort *et al.* [40]; they used both simulated data and tank data to train a regression model to localize a sound source in a tank. Comparisons between source localization error from ML and traditional inversion methods were made, finding that ML provided more accurate predictions. Uncertainty in the tank dataset was caused by (1) varying the source position along a single axis and (2) using an acoustic lens that caused refraction in the tank. No temperature variation was introduced into the water in either experiment. In Yangzhou's work, the ML models were tested with data from the same statistical distribution, as opposed to testing the models with data taken from measurements under different conditions. In Lefort's work, all variability from the acoustic lens was included in the training data.

A third example of a tank-based experiment in source localization is described in Lin, *et al.* [41]. This work, however, dealt only with simulations of tank measurements; one a simple 1 x 1 x 10 m simple rectangular acoustic field, and the other a more complex cavitation chamber with dimensions of 10 x 2.6 x 1.5 m. No recorded tank data was used for training or testing of the neural network. This paper also claimed improvements over Yangzhou, *et al.* [39], and Kita, *et al.* [29], due to modifications in feature extraction methods. The paper by Kita, *et al.* [29], discussed a source localization neural network for analyzing noise sources in a small acrylic-walled space. Analysis of acoustics in an enclosed space in air is not dissimilar to an analysis in a water tank, because both share properties of reverberant environments.

A more common application of machine learning models with tank data is for autonomous underwater vehicles (AUVs). For example, Raza et al. [42] used a water tank to test a wireless communication network with a AUV based on machine learning, and Sung et al. [13] used a tank to test an AUV's object detection capabilities.

## 1.4 Transfer Learning

As mentioned at the beginning of Ch. 1, the ocean is a dynamic, changing environment. As a result, training a deep learning model that can generalize to the variability in the ocean is challenging. On the scale of this research, training a neural network that can still perform source ranging when the water temperature changes poses an analogous challenge. The proposed solution to this problem is transfer learning [43]. Transfer learning is the process of refining a pretrained network for a slightly different task. Advantages of transfer learning include requiring less training data and lessening computation time. Transfer learning has great potential to be applied to underwater source localization, as has already been shown [44–48]. A model pretrained with simulated data can be refined using additional data simulated for updated environmental parameters using ocean measurements, e.g., bathymetry and HYCOM data for water sound speed. Because the training time for transfer learning is much shorter than training an entire network, transfer learning provides the opportunity for improving the robustness of real-time ML applications.

## 1.5 Key Contributions

One primary contribution of this research is using measured acoustical data from a water tank for deep learning purposes. A neural network has been trained to predict source-receiver range from the recorded acoustical data. While similar methods are described in [39] and [40], this work differs by showing comparable deep neural network performance using only a single hydrophone when compared to an array of hydrophones, shown in Sec. 3.2.1. In addition, the neural networks trained for this work are tested against new recorded datasets, referred to as generalizability tests (Sec. 3.1.1) .

The other key contribution in this work is the use of transfer learning between different sound propagation environments. Transfer learning has been applied in other ocean acoustics research [44–48]. However, this thesis describes transfer learning with underwater acoustical data from a laboratory tank. The work in this thesis is also the first to use a measured acoustical dataset for the training of the base model, before transfer learning; all other examples have used simulated acoustical datasets. The transfer learning methodology was also applied to tank environments where the temperature was deliberately changed.

## 1.6 Outline of Thesis

As stated at the beginning of this chapter, this project aims to explore the development of machine learning methods for sound source localization in the presence of environmental variability. This project combines different topics, as described here.

In Chapter 2, the experimental methods are described. The first main part of the process involved recording acoustical measurements in the lab tank. The hardware and software required for the measurement system is described. Once the data are recorded, they undergo preprocessing. Next, the deep learning methods is discussed, including model architectures and many other settings and hyperparameters used. Methods of data augmentation are also given. Finally, the process of using transfer learning for adapting neural networks to different water temperatures is explained.

In Chapter 3, the results of various experiments are given. The chapter begins with a discussion of the different metrics used. Then follows a discussion of the effect of the size of the training dataset on performance of a regression model. Those results are compared with classification models. A short description of the advantages of data augmentation is provided. Finally, the results for transfer learning for both regression and classification models are described. Conclusions from this thesis and suggestions for future work are provided in Chapter 4.

# Chapter 2

# Methods

As summarized in Sec. 1.6, this chapter outlines the methods used in this project. First, the measurement details are described, including the hardware and software necessary to record acoustic measurements in the laboratory tank and details about the dataset itself. Then, the methods relating to deep learning are presented. This section includes descriptions of the neural network architectures, differences between classification and regression, hyperparameters used in the training process, data augmentation, and transfer learning.

## 2.1 Measurement Details

Acoustical measurements are taken inside a laboratory water tank [49]. To take these measurements, hydrophones were placed at the end of robotic arms. The robots moved the hydrophones to different tank positions. The robots' motion and the hydrophone settings were controlled by a custom LabVIEW program called Easy Spectrum Acoustics Underwater (ESAU). For all measurements, a single source hydrophone sent a linear chirp into the water tank to be recorded by other hydrophones. The hydrophone type was varied depending on the bandwidth of the signal of interest.

### 2.1.1 Hardware

Measurements are taken in the laboratory water tank of the Underwater Acoustics Lab at BYU. The tank, shown in Fig. 2.2, is a rectangular acrylic tank made by Engineering Laboratory Design, Inc., that is 3.66 m long and 1.2 m wide, and is filled to a depth of 0.5 m. The tank is filled initially with tap water. As the water evaporates naturally over time, the tank is refilled with distilled water to prevent the water hardness from increasing. A water filter with a built-in debubbler is used to clean the water periodically. Additionally, the filter has the capability to heat the water. The side walls of the tank are lined with Apltile SF5048 attenuating polyurethane panels from Precision Acoustics, which reduce side-wall reflections for ultrasonic frequencies, up to 30 dB.

For the lower frequency band studied in this project, 50-100 kHz, Brüel & Kjær 8103 piezoelectric transducers are used as source and receiver. This hydrophone was chosen because of the relatively flat frequency response over the selected band. The transmitting B&K 8103 receives the source signal from an arbitrary waveform generator (AWG) (Spectrum Instrumentation M2p.6546-x4) card, which is passed through a power amplifier (TEGAM Model 2350). Next, the TEGAM output is passed through a transformer designed to better match the impedance between the amplifier and a piezoelectric source [50]. The transducer sends the signal into the tank, where three B&K 8103 hydrophones record the signal, shown in Fig. 2.1. The receiving hydrophones are connected to a B&K Nexus Conditioning Amplifier. The user defines two different settings on this amplifier. One setting is a charge sensitivity in pC/Pa, which is specific to each B&K hydrophone, and is on the order of 0.09 pC/Pa. The other setting is generic across all hydrophones, for the purpose of processing the recorded signal. This transducer sensitivity is set at 10 mV/Pa. The received signal is finally digitized by a Spectrum Instrumentation M2p.5932-x4 input card, before being recorded by the computer. The signal is sampled at 1 MHz.

For the 150-200 kHz case, Teledyne Reson TC4038 hydrophones are used as the source and receivers. The setup is similar to the B&K hydrophones. The source signal is still sent from the

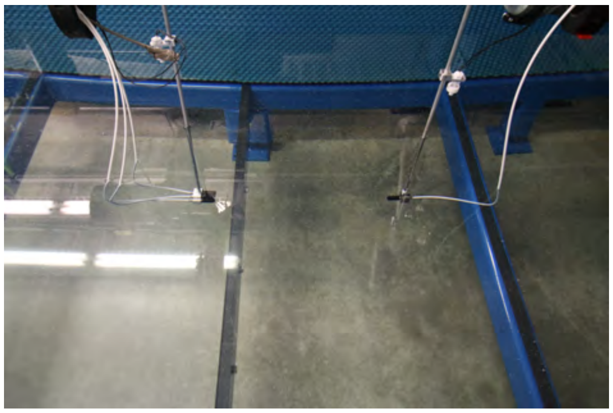


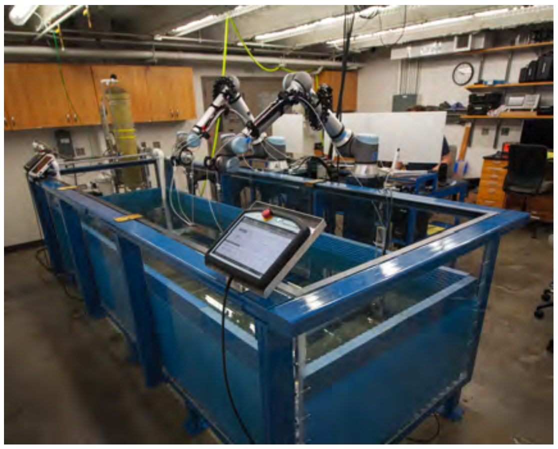
**Figure 2.1** Hydrophone setup in laboratory tank, using B&K8103s. The source hydrophone is on the right. Three receiver hydrophones are attached on the left.

AWG card to the TEGAM amplifier and passed through an impedance matching transformer before going to the source transducer. The receivers are each plugged into a Teledyne Reson VP2000 Voltage Preamplifier EC6081 mk2. These preamplifiers require gain settings and band-pass filter settings; a gain of +50 dB and band-pass filtering between 50 kHz and 500 kHz are used. The output of the preamplifier is also recorded by the Spectrum input cards. Again, the signal is sampled at 1 MHz.

Hydrophones are positioned in the tank with UR10e collaborative robots from Universal Robotics, shown in Figs. 2.2b and 2.1. The robots have a precision of $\pm 1$ mm for repeat measurements. A long, thin aluminum pole extends between the robotic arms and the hydrophone attachments, to allow for hydrophone placement without submerging the robotic arms.

### 2.1.2 Software

The measurement system is controlled from the custom LabVIEW program, ESAU. A screenshot of the user interface is shown in Fig. 2.3. ESAU is a modified version of the LabVIEW program Easy Spectrum Time Reversal (ESTR), described in Kingsley, *et al.* [51]. This system is used to define the generated signal that is sent into the tank as a source. For this project, the source signal


**(a)** A wide shot of the measurements tank.

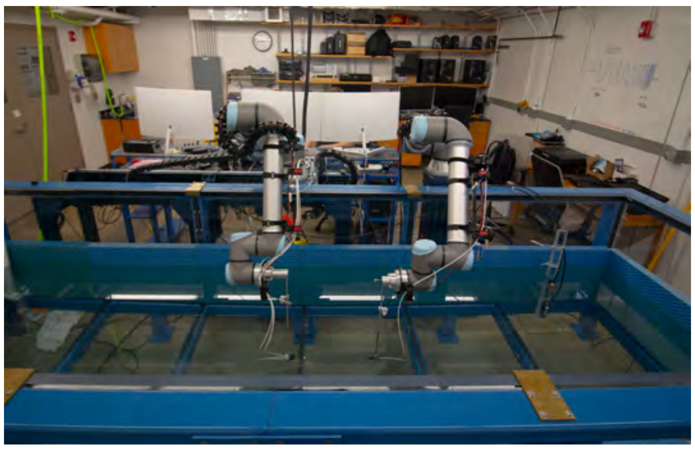

**(b)** View of the tank from the side. The UR10e robotic arms are featured in the center of the image, with hydrophones at the end of the robot attachments.

**Figure 2.2** BYU Underwater Acoustics Lab water tank and automated positioning system.

is a 1-second linear chirp, from the lower bound of the bandwidth to the upper bound, padded by 20 ms of zeros at the beginning and end. A gain of 3600 mV is applied in ESAU. This signal is sampled at 1 MHz, identical to the recorded signal sampling frequency.

ESAU can also control the movement of the UR10e robots that position the transducers at different locations in the tank. Given a grid of measurement points, the program produces the same source signal for each combination of specified source and receiver locations, and sequentially records the received signals. This feature is used extensively to obtain the measurement sets used for deep learning.

## 2.2 Datasets

Measurements were repeated at the two different frequency bands: 50-100 kHz, and 150-200 kHz. Measurements were taken on different days. Each measurement used in the original training data consisted of a scan of 301 different waveforms recorded at evenly-spaced unique source-receiver

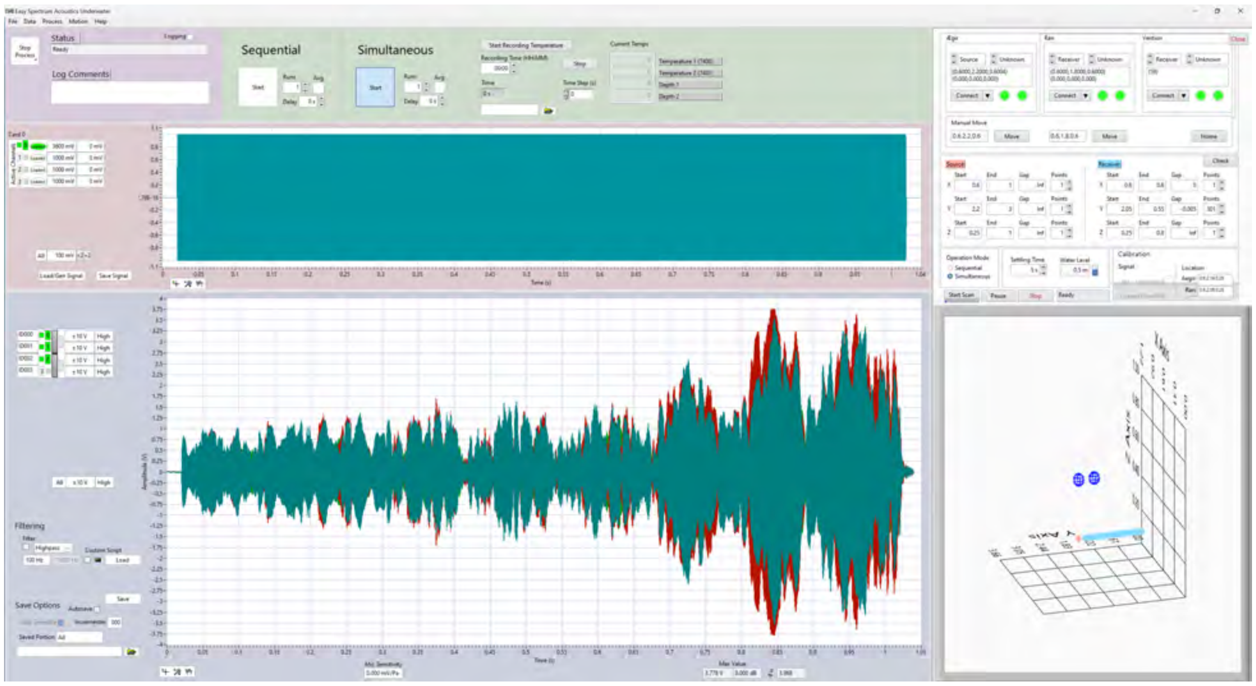


**Figure 2.3** Screenshot of the ESAU user interface. The source signal waveform is generated in the top left of the image. The received signal(s) are shown in the bottom left. Robot position settings for a scan are on the right of the image.

ranges. The ranges vary from 15 cm to 1.65 m, in the $y$-direction of the tank. The source-receiver range is used as labels for the training data. The measurements used for testing the trained networks and for implementing transfer learning consisted of 51 measurement points. The transfer learning datasets were recorded when the water was intentionally heated with the filter, as described in Sec. 2.1.2 and [49]. Additional test datasets were recorded where offsets in the $x$- and $z$-directions were included up to 2 cm.

Each data sample is a time-dependent waveform, $x_i(t)$, from a single measurement at a specific source-receiver range $r_i$. The raw data contain 1.04-second recordings of the received signal. The source signal is a 1-second linear chirp, as described above in Sec. 2.1.2. The 20 ms zero-padding means that the signal can be considered periodic for the FFT and eliminates the need for windowing in the signal processing. In addition, the reverberation time for the tank while lined with panels was calculated to be 6.96 ms in [52]. Thus, 20 ms of trailing zeros after the signal ends is sufficient to capture relevant reverberation.

The measurements undergo preprocessing to be converted into time-averaged spectra, $X_i$, prior to being input to the neural network. To do this, each waveform has its mean set to zero. Then, the time-domain signals recorded in the tank were converted into time-averaged spectra using the FFT function from NumPy's (v1.19.0) built-in FFT module. The FFT result is divided by $\sqrt{N}$ based on the chosen normalization method, and is multiplied by a factor of $\sqrt{2}$ to obtain the single-sided spectrum result for sample $i$

$$X_i(f) = \sqrt{\frac{2}{N}}\mathrm{FFT}[x_i(t)]. \tag{2.1}$$

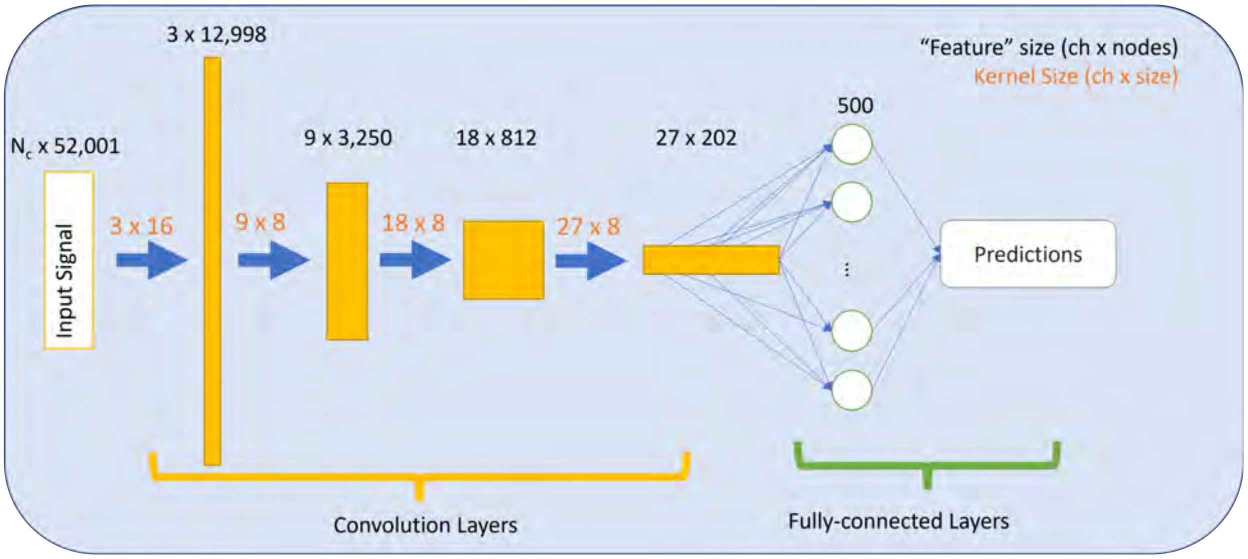
A choice was made early on regarding the format of the data for training. Three types of input were tested: (1) power spectral density levels, (2) a two-channel input containing the real and imaginary parts of the complex spectral density, and (3) two-channel input containing the magnitude and phase of the complex spectral density. Comparisons were made between training and validation of neural networks on each type of input data. Preliminary tests showed similar performance for all three with more consistency obtained using levels. This observation is supported by Herchig, *et al.*, who found that for greater variability in the training data, real-valued data (levels) outperformed complex-valued data [8]. As a result, power spectral density levels are used in this work:

$$L_i(f) = 10\log_{10}\left(\frac{X_i^*(f)X_i(f)}{p_{\text{ref}}^2}\right), \tag{2.2}$$

where $p_{\text{ref}} = 1\mu\text{Pa}$, and $^*$ indicates the complex conjugate.

One step of preprocessing is done on the $L_i$ prior to training. As commonly done in machine learning to expedite training, the spectra are normalized according to the statistics of the entire dataset, which is usually made up of multiple measurements. At this point, the data are already in the frequency domain, with units of dB re 1 µ*Pa*. The mean and the standard deviation of the entire dataset to be used for training are calculated. Then, the normalization is done by subtracting the mean of the dataset and dividing by the standard deviation. The resulting normalized dataset has zero-mean and a standard deviation of 1. When a trained model is tested on new data, the normalization values (mean and standard deviation) of the original training dataset are used, rather than the mean and standard deviation of the new test dataset.

The set of measurements used during training, potentially recorded on different days, constitutes the training dataset. Various combinations of measurements were used to train different models. This variability was added in an effort to increase statistical validity of results in terms of generalizability and robustness. Training data is taken either from all three channels, or just the center channel, if comparing between different numbers of hydrophones.



2

**Figure 2.4** 4-layer convolutional neural network. The predictions layer is modified depending on whether the problem was formatted for regression or classification. The numbers above the layer boxes refer to the feature size at that point, or number of channels by feature length. The numbers above the arrows refer to the kernel or filter size of the convolutional layer operation, in number of filters by filter length. More details are given in Table 2.1.

## 2.3 Neural Network

The model chosen for this work is a four-layer Convolutional Neural Network (CNN), whose architecture is shown in Fig. 2.4. The model is chosen because it performed well for source ranging problems with impulsive ocean data in the past [8, 32, 33, 53]. More details on the uses of CNNs are given in Sec. 1.1. The network inputs are one-dimensional signals, which is different than when CNNs are applied to a two-dimensional image. The network can input only a single channel of data or it can be modified to read all three channels of the hydrophone array, as indicated by the variable $N_c$ in Fig. 2.4. The input data is passed through four convolutional layers, whose filter sizes, strides, and zero paddings are shown in Fig. 2.4 and Table 2.1.

A convolutional layer takes as input a feature map. Given a kernel of a certain size, much smaller than the length of the input feature, the convolution operation applies the kernel at different intervals that are a stride's length apart. At each stride step, an element-wise multiplication is performed, then the results are summed to yield a single value. Those new values form the output of the layer and the feature map input to the following layer. Each convolution layer is followed by a ReLU, or a Rectified Linear Unit, activation function and a max pooling layer. The purpose of the ReLU is to introduce nonlinearity to the model, which allows it to develop complex relationships between the input features and the labels [54].

A max pooling layer involves a kernel sliding across the data, much like a convolutional layer. At each stride, the network takes the maximum value within the kernel and applies that to the next layer [55]. Max pooling is typically used with convolutional neural networks to reduce the number of computations necessary as well as to prevent overfitting by focusing on larger features. In this work, max pooling was used during the tests done in Secs. 3.2, 3.3, and 3.4, where the bandwidth and hydrophone type in addition to the number of training samples were analyzed. However, when the time came to apply transfer learning, a base model without max pooling added was used, as shown in Table 2.2. A preliminary test was done to repeat some of the results in Sec. 3.2 for a model

**Table 2.1** Model architecture and parameters with max pooling. The following abbreviations are used in the table: conv, convolutional layer; relu, rectified linear unit; batchnorm, batch normalization layer; maxpool, max pooling layer; fc, fully connected layer. (list them)

| CNN Layer | {Kernel,Stride,Padding} | Channels | Dimension (Length) | Parameters |
|---|---|---|---|---|
| Input | | $N_{\text{channels}}$ | 52,001 | |
| conv1 (relu) | {16,4,2} | 3 | 12,998 | 147 |
| batchnorm1 | | | | 6 |
| conv2 (relu) | {8,4,3} | 9 | 3,250 | 225 |
| batchnorm2 | | | | 18 |
| conv3 (relu) | {8,4,1} | 18 | 812 | 1,314 |
| batchnorm1 | | | | 36 |
| conv4 (relu) | {8,4,1} | 27 | 202 | 3,915 |
| batchnorm1 | | | | 54 |
| maxpool1 | {8,8,0} | 27 | 25 | |
| fc1 | | 1 | 500 | 338,000 |
| output | | 1 | 1 | 501 |
| Total | | | | 334,216 |

**Table 2.2** Model architecture and parameters without max pooling, used for transfer learning results in Sec. 3.5

| CNN Layer | {Kernel,Stride,Padding} | Channels | Dimension | Parameters |
|---|---|---|---|---|
| Input | | $N_{\text{channels}}$ | 52,001 | |
| conv1 (relu) | {16,4,2} | 3 | 12,998 | 147 |
| batchnorm1 | | | | 6 |
| conv2 (relu) | {8,4,3} | 9 | 3,250 | 225 |
| batchnorm2 | | | | 18 |
| conv3 (relu) | {8,4,1} | 18 | 812 | 1,314 |
| batchnorm1 | | | | 36 |
| conv4 (relu) | {8,4,1} | 27 | 202 | 3,915 |
| batchnorm1 | | | | 54 |
| fc1 | | 1 | 500 | 2,727,500 |
| output | | 1 | 1 | 501 |
| Total | | | | 2,733,716 |

without max pooling, and no significant difference was observed. The model with no max pooling is used for the results in Sec. 3.5.

Finally, the model has two fully-connected, or linear, layers. The final linear layer can be modified depending on whether the current problem is a regression problem or a classification problem. For regression, it is left as a singular value output, which is the predicted range. During training, the predicted ranges are compared to the range labels for each sample.

### 2.3.1 Classification

Some modifications were made to test the efficacy of treating the problem as one of classification, rather than regression. The primary modification is the output of the final layer of the neural network: the size of the output vector is equal to the number of classes, and a softmax function is applied to this final layer before the final prediction. Instead of the final output being a numerical prediction of the source-receiver range, the softmax function on the final layer of the neural network outputs an array of probabilities that the input data belongs to each respective class.

Another change necessary for classification is that when the dataset is loaded in prior to training, the number of classes is input by the user. The number of classes is used to divide the dataset into that many bins, into which the data can be sorted based on the original range label. The range labels are converted into binary vectors indicating the bin number to which the spectrum belongs. For example, if the number of classes is two, every data sample that is in the closer half of the set would have the label class 0, and the data samples in the further half of the set would have the label class 1. Typically, a larger number of classes was selected during testing.

Finally, label smoothing is used to improve generalization [56]. The labels, instead of being exactly one-hot encoded with a one to indicate the correct class and a zero to indicate the incorrect classes, have small values set at 0.1 for the incorrect answers.

### 2.3.2 Training Details

As the training data is loaded in, the user has the option to select its format. Three options were tried: splitting each hydrophone spectrum into two channels, either the real and imaginary parts of the complex spectra, or the magnitude and phase; also, looking at the data as levels in decibels was tested. A discussion of the reason for choosing real-valued inputs as levels was given in Section 2.2. In addition, the user can select to use all three hydrophone channels for training, or only to use one or two of the three channels. In this way, the performance of training based on number of hydrophone channels used in the dataset was tested.

The model is built in Google's PyTorch [57](v1.10.1), through which the machine learning is also set up. Training is done on a Tesla T4 GPU. A default batch size of 32 is used, and the model is trained for a maximum number of 15 epochs. For regression models, PyTorch's MSELoss is used as the criterion (nn.MSELoss). Alternatively, for classification models, Pytorch's CrossEntropyLoss is used, with labelsmoothing=0.1.

The Adam optimizer is used, with weight decay regularization included (AdamW). It has been shown that when using the Adam optimizer, weight decay regularization is a better choice than standard $L_2$ regularization [58]. The Adam optimizer includes a parameter, $\varepsilon$, in the denominator of one of its operations that helps prevent divide-by-zero error. For regression models, the optimal value for this study was found to be $\varepsilon = 0.5$, while for classification models, decreasing the value to $\varepsilon = 0.01$ improved model performance. Details on the AdamW optimizer can be found in the documentation for PyTorch's torch.optim.AdamW [57].

When training begins, an initial learning rate of 0.001 is used. As training progresses, the learning rate is modified by a cosine annealing scheduler. This approach modifies the current learning rate following a sinusoidal pattern, which improves rate of convergence and overall deep learning training performance [59]. Documentation is found under Torch.optim.CosineAnnealingLR [57].

Five-fold cross-validation is used during the training process to remove dependence on the random split of the training and validation data. During a typical ML training process, a portion of the training dataset is left out of training to be used for validation, or validating the trained model. Here, that division is an 80-20 split; or, for each fold, 80% of the data are used for training and 20% are left out for validation. The five-folds refer to the fact that for each training run for a dataset, the 80-20 split is done five times. The dataset is randomly split into five subsets, and five different models are trained, each with a different subset used as the validation set. When dealing with a classifier model, stratified cross-validation was used. Stratified cross-validation forces each class to be distributed equitably between the training and validation datasets, to ensure that each class has an equal chance of being seen during the training process.

An early stopping mechanism is implemented to reduce the possibility of overfitting during training. After each epoch, the root mean-squared error is checked for the validation data. If more than 5 epochs pass without seeing more than 0.001 improvement (or decrease) in the validation RMSE, training is stopped before it gets to the maximum of 15 epochs.

## 2.4 Data Augmentation

Data augmentation refers to methods of artificially generating new data for training a neural network. These data are often drawn from existing sets, but have been modified, or augmented, in some way. Data augmentation is useful when very large datasets are not available for training, specifically by preventing overfitting [60]. Methods of data augmentation for ocean acoustic training data, specifically, ship noise spectrograms, are described in Castro-Correa, *et al.* [61]. In addition, methods of data augmentation for 1-D data have been developed [62]. In this work, data augmentation refers to adding different types of noise to individual data samples during the training process to increase the model's ability to generalize to a variable environment. Two different types of data augmentation were explored to account for background noise or ambient noise in the tank.

The first method was to add randomly weighted Gaussian noise to every spectrum in the batch. To generate this noise, first, a normal distribution of noise the size of a batch of training data is instantiated, with a mean of zero and standard deviation of 0.5. A noise weight upper limit of 0.05 times the standard deviation of the dataset is also defined. Then, a factor defined as "noise weight" is randomly chosen between zero and the noise weight upper limit. Then, the Gaussian distribution of noise is modified by that noise weight, and added to the batch of data.

The second method considered the tonal ambient noise that likely comes from electronics in the lab and outside of the lab. During training, each batch had a 50 % probability of having tonal noise added. When the random chance comes out in favor of adding tonal noise, three tones are selected to be added at random frequencies within the bandwidth of interest. Once the frequencies are chosen the amplitude of the tones are individually randomly sampled from a uniform distribution between 0-30 dB. That value is then added to the tones in the batch.

## 2.5 Transfer Learning

A major hurdle to overcome in machine learning is the lack of a large quantity of labeled data. For some problems, a source of large labeled datasets does not exist. In such cases, transfer learning can take advantage of pretrained large models that have some resemblance to the desired task. Transfer learning refines the pretrained model on a smaller, more specific dataset pertaining to the relevant task.

For this research, transfer learning is used to apply the pretrained source localization model to acoustic data recorded under different conditions. The majority of the training data used in this research was recorded when the water tank was at room temperature. While the generalization comparisons in Sec. 3.2 used test data taken when the water was also at room temperature, the model's generalizability was not tested to changing environmental parameters.

Because temperature affects sound speed, the trained model has trouble generalizing to data where the water temperature is warmer. For this task, transfer learning is used to increase the model's performance for predicting source range when the water temperature is increased. Advantages of transfer learning include only requiring a fraction of the resources, including training data and time, to create an updated model that can perform better in the new environment.

To apply transfer learning in this work, all but the last layer of weights of the original model are frozen during training [63]. For results shown in Sec. 3.5, a new subset of 51 data samples recorded when the water was at a warmer temperature, is used for training. All but the last layer of weights are frozen, so that the training process only updates a single layer.

# Chapter 3

# Results

## 3.1 Overview

This chapter contains the results of using various ML techniques on measured tank data. In the rest of Sec. 3.1, descriptions of the metrics used to analyze the results are provided. In Sec. 3.2, details of the results of experiments done to tune hyperparameters like the dataset size and number of hydrophone channels are described. Then, results of experiments with classification-based neural networks are shown in Sec. 3.3. A small description of the advantages of data augmentation for these types of datasets is given in Sec. 3.4. Finally, Sec. 3.5 provides the results for using transfer learning for warmer water datasets.

### 3.1.1 Validation and Generalization

The results are divided into two different categories: validation and testing, or generalization. Clarification is needed about what is meant by validation and generalization testing. The data used for validation testing are drawn from the same statistical distribution as the training data. The validation dataset in this work is a subset of the training dataset that is left out during model training.

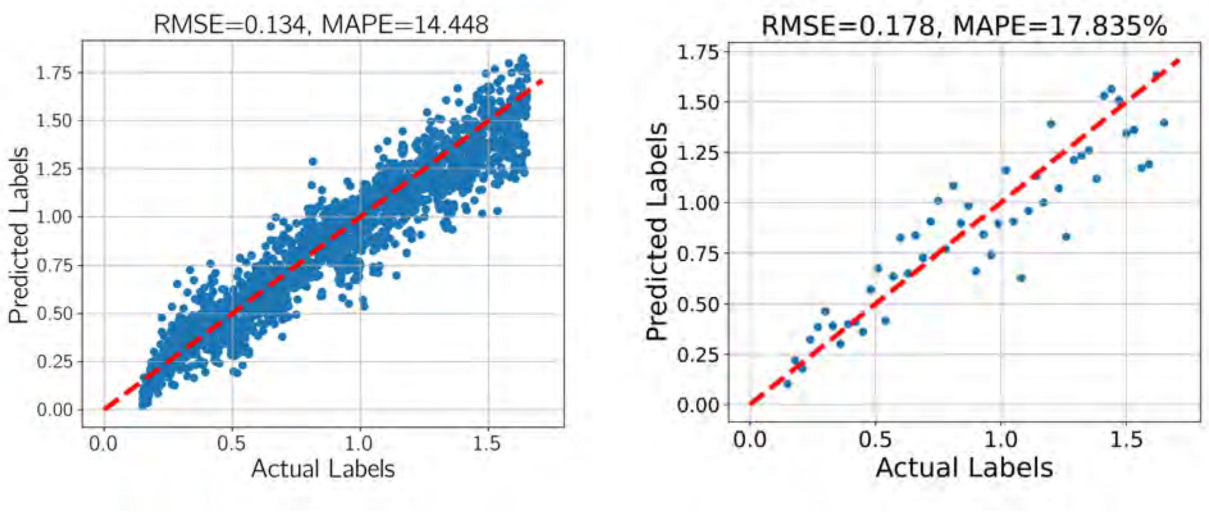
Validation is useful for monitoring model training to provide indicators pertaining to the model performance and prevent overfitting. However, the validation data are still part of the training dataset, and share characteristics with the training data. For this project, these shared characteristics refer to the environmental conditions in the lab during the measurements, i.e., measurements taken on the same day that have the same ambient noise.

Alternatively, generalization testing occurs after the training process is complete to evaluate how well a trained model can generalize what was learned. Generalization testing occurs when a trained model is tested on data drawn from a different statistical than the training data. Generalization testing measures the robustness of machine learning models, especially in the presence of changes in ambient noise and other measurement conditions. As a result, generalization results are considered more significant than validation results. For this project, the generalization or test data, come from measurements taken at least a couple months later with different ambient noise.

An example of the differences between validation and generalization results is shown in Fig. 3.1. The validation results, in Fig. 3.1a, show the predicted and actual labels for range for the 20% of the training dataset at the end of a training session. Results for the same trained model tested on generalization data are shown in Fig. 3.1b. As illustrated here, throughout this thesis, the generalization datasets contain 51 points and are smaller than the validation datasets. Although these two results are visually comparable, the predictions on the validation data had smaller errors than generalization data. The metrics by which performance is measured are described in the next section.

### 3.1.2 Metrics

Two primary metrics are used to evaluate the performance of models trained for regression, as described in Sec. 2.3. The first is the root-mean-square error, or RMSE. For a dataset containing $N$ samples,



**(a)** Validation results

**(b)** Generalization results

**Figure 3.1** Results for the ranges predicted by a a single model trained on 80% of a dataset containing 1,806 data samples. (a) The validation results are on the remaining 20% of that dataset. (b) The generalization results are on a dataset containing 51 data samples that were measured several months later. The range labels are given in meters (m).

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}, \tag{3.1}$$

where $y_i$ represents the true labels for data sample $i$, and $\hat{x}_i$ represent the prediction from the trained model. For this project, the RMSE is calculated between the predictions for source-receiver range ($\hat{y}_i$) and the true labels for range ($y_i$). The validation RMSE is evaluated at the end of every epoch and used to determine when the early stopping criteria, described in Sec. 2.3.2, have been met. If validation RMSE does not significantly improve for 5 epochs in a row, then the training process stops early, before reaching the maximum number of epochs.

With applications like source localization, often the percentage difference between the predicted and true labels is more important. For example in ocean applications, a 500 m error in source range is much more significant is a sound source is 1 km away than if it is 10 km away. To quantify the percentage difference, the second metric used for measuring the performance of regression models is the mean absolute percent error, or MAPE, defined as

$$\text{MAPE} = \frac{1}{N}\sum_{i=0}^{N-1}\frac{|y_i - \hat{y}_i|}{y_i} \times 100\%. \tag{3.2}$$

While both RMSE and MAPE are calculated for the validation datasets and test datasets, MAPE is only used after training, to compare model performance in both instances. MAPE is used for validation to compare between models trained on different size datasets. During generalization, MAPE is used as the primary metric for the models' robustness.

For classification models (described in Sec. 2.3.1), a different metric is used: accuracy. Accuracy is the percentage of correct label classifications out of the total number of predictions:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100\%, \tag{3.3}$$

where TP and TN signify True Positives and True Negatives, or correct guesses, and FP and FN signify False Positives and False Negatives, or incorrect guesses.

The optimal operational tolerance for source ranging in the open ocean, according to our Office of Naval Research Program Officer, is 10% MAPE. This tolerance has not been met by the current work, likely due to the highly reverberant environment of this experiment. (As is described below, the baseline average for validation testing was about 15%, and about 18-20% for generalization testing.) Regardless, the significant result is that transfer learning is to able to achieve the same performance for new data as the baseline averages.

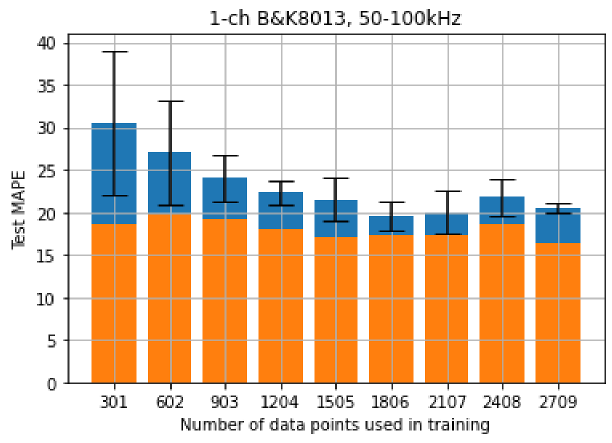
## 3.2 Training Dataset Size

An initial question was how many data samples in a training dataset are needed to train a regression model. An analysis was done to find the optimal number of data points to use for training. This analysis was completed by training models on different quantities of measurement scans combined to form larger datasets. A single measurement scan across the span of source-receiver ranges consisted of 301 distinct data samples, as described in Sec. 2.2. As a result, dataset sizes were defined in intervals of 301 samples. Starting with just one measurement scan comprising a dataset of 301 samples, and going as high as nine measurement scans, or 2,709 data samples, the effects of combining multiple scans together for training were analyzed. These tests were repeated for data in the 50-100 kHz range and the 150-200 kHz range for both a single hydrophone and three hydrophone channels as input (i.e., $N_c$ = 1 and 3 in Table 2.1).

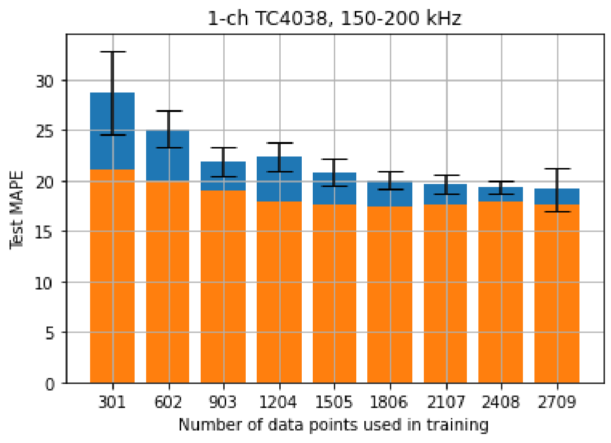
Results for the effects of training data size on model performance are shown in Fig. 3.2. For each number of data points used in training, several unique combinations of measurements from different days were used to train five independent models using five-fold cross validation. For example, 6 different training datasets with 301 points were used to train five models each. Thus, the results on the columns corresponding to 1,806 points represent the mean and standard deviation of the MAPE (Eq. 3.2) across all 30 models. Also important to note is that a model trained on 1,806 data samples was trained only on 80% of that data, with 20% left out for validation. For simplicity, the quantity of samples in the entire dataset is used to describe the different cases.

An individual plot in Fig. 3.2 shows these averages across validation and generalization MAPE values for many different models trained with same size training datasets. Each bar shows results for a different training dataset size, as labeled on the x-axis. The orange bar shows the average percent error in validation across training multiple models, and the blue bar shows average percent error during generalization. The error bars show standard deviation for generalization error.

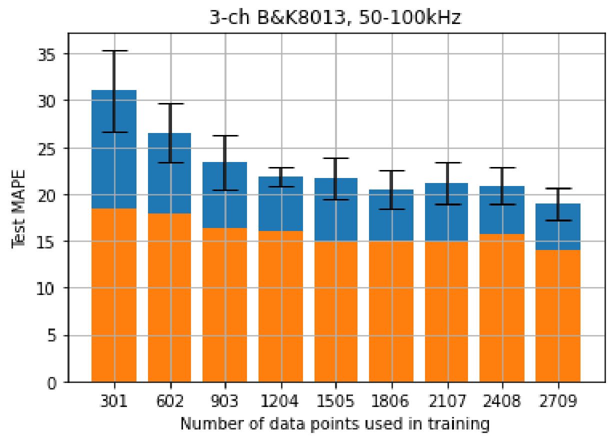
The impact of training dataset size on model performance are shown in Figs. 3.2c and 3.2d and show results of when the model was trained on three hydrophone channels of data. Both cases, the two frequency bands, showed similar results. For smaller dataset sizes, increasing the number of measurement scans included in the dataset caused a significant improvement in model performance,



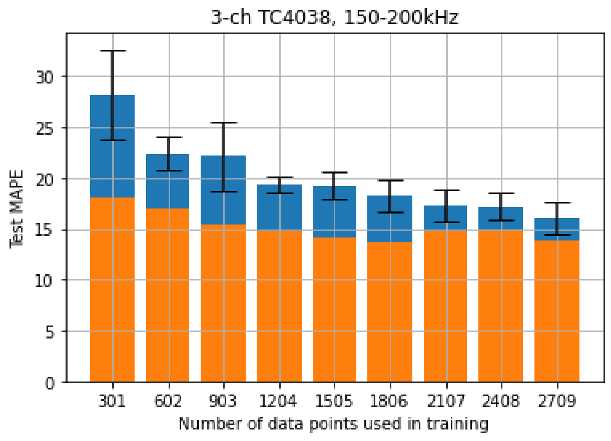
**(a)** 50-100 kHz, 1 channel



**(b)** 150-200 kHz, 1 channel



**(c)** 50-100 kHz, 3 channels



**(d)** 150-200 kHz, 3 channels

**Figure 3.2** Effects of training dataset size on model performance. The blue bars show the MAPE for a given training dataset size, averaged across different folds of cross-validation for models trained with different combinations of training data. The error bars show standard deviation in the MAPE across all trained models. The orange bars (plotted on top of the blue bars) show the average validation MAPE for the same models.

particularly in generalization MAPE. This trend continues until about 1,505 or 1,806 data samples in a set. As dataset size increased beyond this number, some improvement may still be achieved; however, that improvement is not as significant as at the lower end of the plot. For the sake of choosing a value, 1,806 was chosen as an optimal dataset size as a compromise between training time and cost, and improvement in model predictions. A standard dataset size of 1,806 is used for all other tests as a result.

Additionally, the models trained with 1,806 data samples were tested for their ability to predict the source range when an offset in the source's position was introduced. Test datasets were recorded where the location in the $x$- and $z$-directions was varied up to 2 cm from the training dataset, where the $x$ direction means lateral movement and the $z$-direction signifies depth in the tank. For comparison, the training datasets contained no variations in these directions; only the location in the $y$-direction, or source-receiver range, was changed.

When tested on data with offsets in the $x$- and $z$-directions, the models showed comparable results to the previous tests, with no appreciable difference. The significance here is that 2 cm was not enough of a difference to be outside of the range of error already present in the models. Future work will record more test datasets with greater offsets from the training data to probe the limit of the models in these directions.

### 3.2.1 Single-channel Data

While initial testing used the three-channel hydrophone data, similar tests were done for models trained only on one hydrophone channel. At the time of data loading, the number of hydrophone channels loaded in could be selected. Using one hydrophone, the center channel, models were trained and tested in the same way as three-channel data above. This testing was done for both frequency bands of interest.

The results for the single channel testing are shown in Figs. 3.2a and 3.2b. Similar trends are seen as described above for three-channel data. Significant improvements are seen in model prediction ability as dataset size increases until about 1,806 data samples in a set. Beyond this size, improvement comes at a higher cost.

These results should be particularly interesting because the source localization task is able to be completed with a single hydrophone channel, rather than requiring an array of three or more. Traditional signal processing techniques are unable to perform a source ranging task with a single hydrophone without information about the environment or source signal, such as a physics-based model of sound propagation, or the source strength level. The fact that this neural network is able to localize the source in one dimension with no prior information about the environment is notable. The neural network is able to learn patterns in the single-channel spectra that are specifically related to source-receiver range while other aspects remain constant, i.e., the source signal, source-receiver depths, and tank environment.

## 3.3 Classification

As described Sec. 2.3.1, a neural network can be modified so that its outputs are in the form of a classifier rather than functioning as a regression model. After the regression tests for analyzing training dataset size were completed, the same datasets were used to train classification models. The source-receiver ranges were divided into classes. Models were trained with different numbers of classes to provide a method for testing the limits of the models' ability to classify the source-receiver range. Details on how the number of classes affects the size of the localization bin and how large the class bin is compared to the full range are shown in the last two columns of Table 3.1.

A summary of the results for classification are shown in Table 3.1. The number of classes were varied as multiples of 8. The validation accuracy is within 20% of perfect results for up to 24

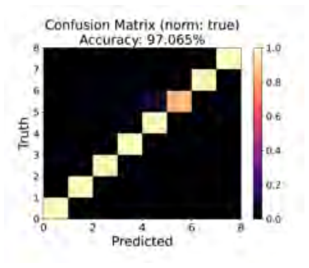
**Table 3.1** Classification results based on number of classes used. Accuracies are given as averages across the five folds of cross-validation, ± the standard deviation. The difference between validation and generalization accuracy is described in Sec. 3.1.1. The bin width describes how wide the range bins are, given the number of classes. The size of the bin relative to the total range of labels (1.5 m, see Sec. 2.2), is given as the uncertainty for the number of classes.

| $N_{classes}$ | Validation | Generalization | Bin Width (cm) | Uncertainty |
|---|---|---|---|---|
| 8 | $96.3 \pm 1.5\%$ | $54.5 \pm 5.3\%$ | 18.75 | 12.5% |
| 16 | $92.3 \pm 2.7\%$ | $34.9 \pm 9.2\%$ | 9.375 | 6.25% |
| 24 | $83.3 \pm 5.2\%$ | $22.7 \pm 5.8\%$ | 6.25 | 4.17% |
| 32 | $71.0 \pm 4.4\%$ | $15.7 \pm 2.8\%$ | 4.6875 | 3.125% |

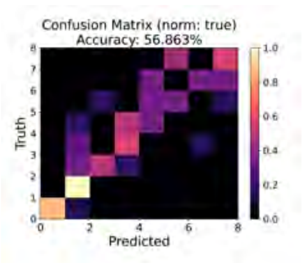
classes. If accuracy out of 100% is compared 1:1 with percent error for regression, then it appears that classification outperforms regression. However, the generalization results tell a different story. Even for the smallest number of classes, 8, the classifier models are only able to predict the correct source location region a little more than 50% of the time. Some efforts were made to improve this number, including random noise data augmentation, label smoothing, and training for fewer epochs. However, these methods only provided marginal improvement; the results shown in Table 3.1 and Fig. 3.3 take into account these modifications.

Classification results can be visualized with confusion matrices, seen in Fig. 3.3. Confusion matrices have the true labels on the y-axis, and the model's predictions on the x-axis. Correct predictions follow a diagonal line from the bottom left of the image to the top right. The results in Fig. 3.3 show results from one model of the five generated from five-fold cross validation. The models chosen for each case had accuracy closest to the average for the five.

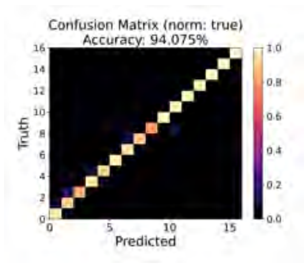
In summary, the current source ranging problem was solved much better by a regression model compared with the classification model results. This failure of the classification model to generalize well contrasts with previous work including that attempted a similar task [15, 27, 28]. In these



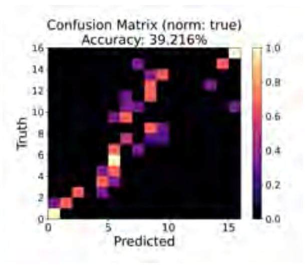
**(a)** 8 classes, validation



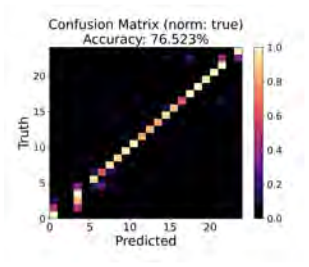
**(b)** 8 classes, test



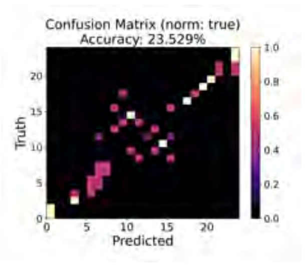
**(c)** 16 classes, validation



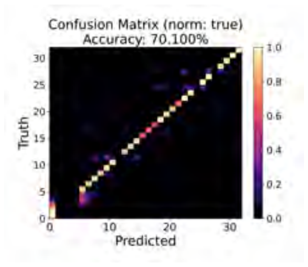
**(d)** 16 classes, test



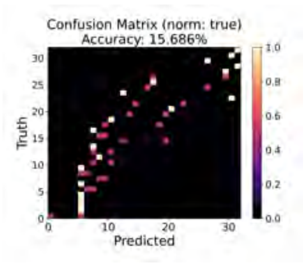
**(e)** 24 classes, validation



**(f)** 24 classes, test



**(g)** 32 classes, validation



**(h)** 32 classes, test data

**Figure 3.3** A subset of classification results. Four different quantities of labels are shown. For each number of classes, the model results are shown for validation data, and for generalization data taken a few months later.

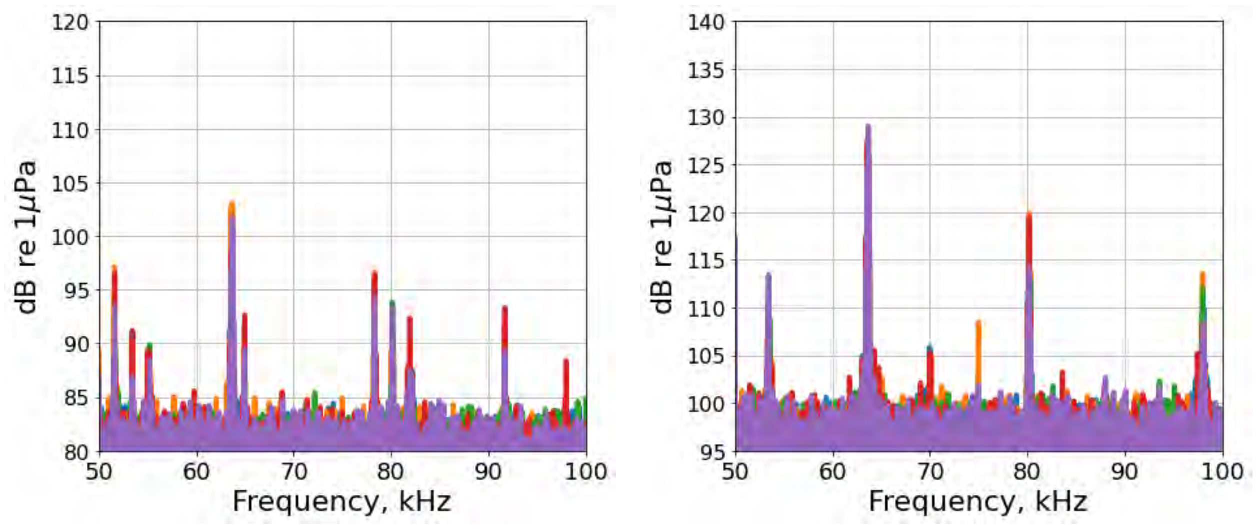
papers by Niu, *et al.*, a classifier performed remarkably well, almost always better than traditional methods like matched field processing. In one of these papers [27], classification outperformed a regression model trained for the same task. However, in this work, comparisons were only done for feed-forward neural networks, support vector machines, and random forests. Some primary differences in the other papers [15, 28] include choosing a ResNet model architecture, training on simulated data instead of tank data, testing on ocean data, and using a dataset far larger than what is used in this work. The dataset size in particular is suspected to be the primary culprit here; when comparing the models used in this work, the main difference between the regression models and classification models is the final layer. For the regression model, the final layer consists of a single node: the output prediction for source range. By contrast, a classification model requires extra outputs in the final layer. These extra outputs correspond to the number of classes, for which the model must predict a probability of each class. The number of extra parameters increases from 501 to $501 \times N_c$, the number of classes. As a result, the current hypothesis is that more training data is needed to handle the training of the larger quantity of parameters required for classification.

## 3.4 Data Augmentation for Smaller Datasets

As measurements were analyzed qualitatively, tonal noise from an external source appeared to be prevalent in the measurements. These noise tones did not appear to be consistent across measurement days. Five separate recordings of ambient noise on two different days are shown in Fig. 3.4. Differences between different days include an overall change in the noise floor (note vertical axes are not the same), changing quantities and levels of tones, and shifts in the tonal frequencies. In addition, between different measurements of the ambient noise on five days, slight changes in the levels of the tones on the order of 5-10 dB can be observed. Because of this time-varying noise, methods of data augmentation to account for the tones were explored. The tonal noise data augmentation is described in Sec. 2.4.

Most results of adding tonal noise did not make significant improvements over the original training. Comparisons were made for the three-channel B&K8103 hydrophone (50-100 kHz) case. These results are shown in Fig. 3.5. This plot is similar to those in Fig. 3.2. The orange bars show validation error for training a collection of different models with combinations of measurements that yield training datasets with 1,806 data points. The blue bars show generalization error, with error bars showing the standard deviations of average MAPE (Eq. 3.2) across the different trained models.

Four different cases are analyzed in Fig. 3.5. In the first three cases, different number of randomly-selected frequency tones of random amplitude have a 50% probability of being added to a batch. The number of tones included were 0, 3, and 6 tones, as indicated on the horizontal



**(a)** Ambient noise on 14 June 2023 **(b)** Ambient noise on 15 June 2023

**Figure 3.4** Background noise from five measurements observed on two different days. The average noise floors differs as well as the frequencies and amplitudes of the tones.

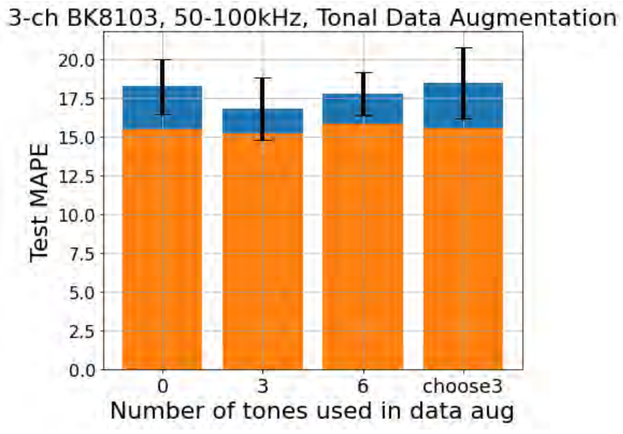


**Figure 3.5** Varying the tones included in the data augmentation

axis. Finally, the fourth option shows results when the frequencies of the three tones are chosen specifically to match the tonal noise observed in background noise on the same day as the test data. Based on these results, the tonal data augmentation did not have a significant positive impact on the neural networks' ability to generalize.

**Table 3.2** Data Augmentation. Models trained on 301 data samples had specific data augmentation applied, based on the ambient noise measured in the tank on the same day as the test data. The results are compared with results for the same size dataset, shown in Fig. 3.2a, which are listed on the last two lines of this table.

| Frequencies Chosen (kHz) | [53.4, 63.6, 75.0, 80.1, 97.9] |
|---|---|
| Data Aug. Val. | $16.1 \pm 5.6\%$ |
| Data Aug. Gen. | $21.9 \pm 3.0\%$ |
| No Data Aug. Val. | $18.7 \pm 1.1\%$ |
| No Data Aug. Gen. | $30.5 \pm 8.5\%$ |

However, adding cleverly-chosen tones to the training data did lead to more significant improvements for models that had only been trained on 301 single-channel data samples for the 50-100 kHz band. Based on analysis of the frequencies where tonal noise was most predominant across days, such as those shown in Fig. 3.4, a set of neural networks were trained with data augmentation in the form of five specific noise tones that matched tones that were common across different days. The results of training this model on a smaller dataset with the tonal noise catered to the test data are shown in Table 3.2. This method proved an 8% decrease in generalization testing percent error, while the validation percent error was only decreased by 2% over the results shown in Fig. 3.2a, which are repeated on the last two lines of this table, for comparison.

These results are interesting for the application of machine learning to situations where a limited amount of labeled data is available. In these cases, clever applications of data augmentation to match characteristics of test dataset conditions would improve results. However, this research has shown that the same thing was accomplished by simply increasing the amount of labeled training data.

## 3.5 Transfer Learning

Building on the preliminary tests described above, the performance of trained models when applied to data measured with different higher water temperatures and correspondingly higher sound speeds. After showing that the trained models do not generalize well to these new data, transfer learning is applied with a small dataset obtained in the warmer water. This process and the improvements obtained with transfer learning are described in this section.

For this work, the original or base models are trained using five-fold cross-validation on datasets with 1,806 samples. The models were trained using the network architecture in Table 2.2, which did not include max pooling.

To test generalizability to different water temperatures, new data were recorded when the water was heated to warmer water temperatures of $22°C$ and $24°C$. This heating was done by manually changing a temperature setting on the tank filter, which heats the water. When the original models, or base models, were applied to these new data, the generalization error was much larger. The average MAPE for both cases are shown in the second column of Table 3.3. As an example, plots of the predicted and actual ranges (in meters) for a single trained model are displayed in Figs. 3.6a and 3.7a for water temperatures of $22°C$ and $24°C$, respectively. The increased errors illustrate the difficulties of applying trained models in cases of changing sound speed or other environmental conditions.

Transfer learning resolves this issue. Transfer learning is applied individually for each new environmental condition, or water temperature. For each case, a 51-point transfer learning dataset, taken at their respective new temperatures, is used to train the transfer learning, as described in Sec. 2.5.

The results from transfer learning as are shown in Table 3.3. The MAPE is averaged across all five folds of training. Both validation and generalization results are comparable to the original

**Table 3.3** Transfer learning results. The original (base) model was trained on data at 20°$C$. The base model is tested on a dataset taken at warmer water temperatures, indicated in the first column. Transfer learning (TL) is applied with 51 data samples. The validation and generalization MAPEs for the transfer learning models for both temperatures are presented as the mean and standard deviation of the MAPE across the five models trained with cross validation.

| Temp(°$C$) | Gen. Error before TL | TL Val. Error | TL Gen. Error |
|---|---|---|---|
| 22 | $33.01 \pm 7.78\%$ | $19.53 \pm 4.23\%$ | $20.59 \pm 3.95\%$ |
| 24 | $41.26 \pm 9.54\%$ | $21.95 \pm 4.76\%$ | $23.06 \pm 4.47\%$ |

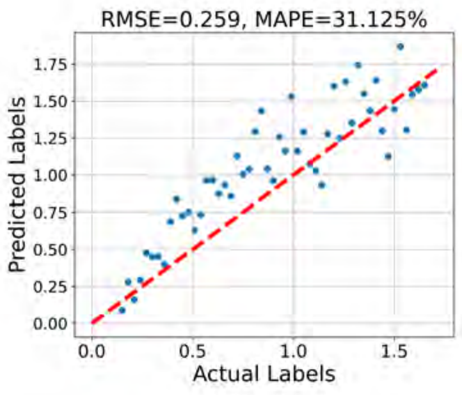
results with data measured at room temperature (see Fig. 3.2c, results for 1,806 data samples). The models obtained after transfer learning was applied are referred to as Model 22 and Model 24.

Finally, the updated models are tested on a new dataset measured on a different day at the warmer water temperatures. Plots of predicted vs. real source-receiver ranges for this new generalization dataset at the individual 22°$C$ and 24°$C$ cases are shown in Figs. 3.6b and 3.7b, respectively. The new generalization error is substantially less than before transfer learning, as shown in the fourth column of Table 3.3. The errors are given as averages and standard deviations across all five folds of training (see Sec. 2.3.2.

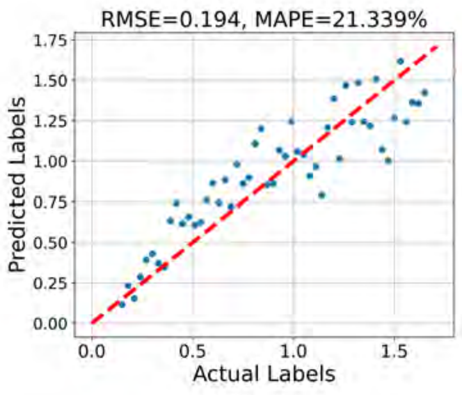
### 3.5.1 Transfer Learning with Classification

The same process for transfer learning was also applied to the classification models. However, just as the classification models had more trouble generalizing to new data, that transfer learning on the classification models did not produce suitable results. This lack of improvement is likely due to the fact that generalization did not perform well in the first place.

The primary reason suspected for this failure comes from the fact that, for classification models, the final layer includes an extra factor of weights equal to the number of classes. For example, a classifier with 8 labels would need 8 times as many weights to be trained in the final layer. To

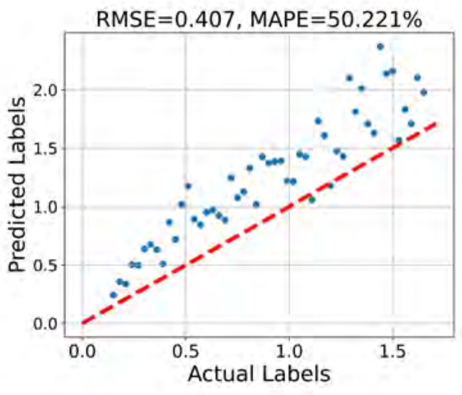


**(a)** Base model tested on 22°$C$ data

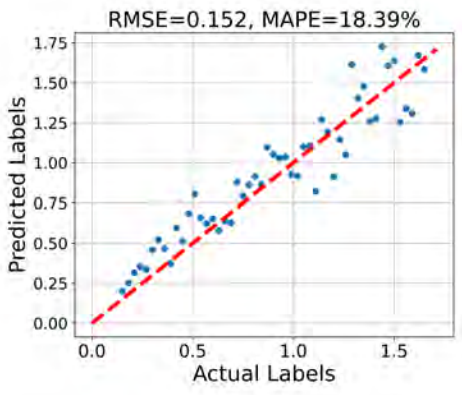


**(b)** Model 22 tested on generalization 22°$C$ data

**Figure 3.6** Transfer learning for water temperature of 22°$C$. Generalization results of (a) a base model and (b) Model22 after transfer learning on the same 51-sample 22°$C$ dataset



**(a)** Base model tested on 24°$C$ data



**(b)** Model 24 tested on generalization 24°$C$ data

**Figure 3.7** Transfer learning for water temperature of 24°$C$. Gneralization results of (a) a base model and (b) Model24 after transfer learning on same 51-sample 24°$C$ dataset

successfully train a classifier that can generalize or be transferred as well as the regression model, more training data and/or training time would be needed.

While transfer learning with classification has been proved in the past, for example, in Ge, *et al.* [45], the same results were not evident here. One potential reason is that the work by Ge *et al.* was confined to validation testing, and they did not ever test the models on new datasets, what we refer to here as generalization testing.

# Chapter 4

# Conclusion

To review, this project was motivated by the desire to improve the robustness of machine learning applications in ocean acoustics; specifically, the need to account for the spatial and temporal variation in the ocean environment. Training data for development of machine learning algorithms were recorded in the tank in the Underwater Acoustics Lab. With that data, neural networks were trained to predict source-receiver range using the recorded acoustical signals as the input. Various hyperparameters were tested to improve the training and validation. In addition, the methodology of transfer learning was applied to increase the neural networks' ability to generalize to different environmental conditions.

Several studies were conducted. First, the performance of validation and generalization tests (see Sec. 3.1.1) were compared for different quantities of training data. The question was, how many measurement scans, each one consisting of 301 recorded acoustic samples, should be used to train a neural network? The primary finding was that the performance converged when 1,806 training data samples were used; this size training dataset includes six measurement sets of 301 points each and provides a good balance between accuracy and efficiency (Sec. 3.2). When the dataset size was increased beyond 1,806, the performance did not increase sufficiently enough to justify the extra time and memory cost.

Another study compared the performance of regression and classification neural networks (Sec.3.3). The main difference was in the format of the output of the network. Both types of networks were tested with validation data and with generalization data. For validation data, the classification models performed better. However, when tested for generalization, regression models had better performance. This result only applied to generalization data that was recorded at the same temperature as the training data.

The final and most important study tackled how transfer learning can be used to improve performance in the presence of environmental mismatch (Sec. 3.5). When the original, or base, model, trained with data measured at room temperature, was tested on measurements from warmer water, the generalization error greatly increased. To counteract this, transfer learning was applied. Transfer learning allowed the final layer of the neural networks to be updated by training on data pertaining to the new environmental conditions. After transfer learning, the new models were able to make predictions on the warmer water data nearly as well as the original base model.

Future work should explore other recent developments in machine learning. The field grows and changes very quickly, and new methods and findings are discovered regularly. In addition, there are many different hyperparameters that were already included that can be modified, of which an exhaustive search would be tedious. Specific hyperparameters that would be interesting to explore include the choice of learning rate scheduler, specifics of the model architecture, and incorporating different types of regularization, such as dropout regularization.

Additional variability in the tank measurements could be incorporated by modifying the acoustic environment in more ways than just the temperature. In ocean measurements, features of the seafloor such as the bathymetry and seabed composition have a large effect on sound propagation as sound waves interact with the bottom of the ocean. While the seabed can affect acoustic propagation in the deep ocean environment, the effects are much stronger in shallow ocean environments. It is important that an ocean acoustic source ranging algorithm, such as a neural network, can account for the effects of the seafloor on sound propagation. Future ideas to test this in the tank include modifying the water depth, inserting a 3-D printed interesting bathymetry on the bottom of the tank, and adding different types of sediments, such as sand or mud, to the bottom of the tank.

# Appendix A

# Figure References

This appendix explains the codes and data used to generate the Figures and Tables included in this Thesis. All results are saved in underwater/corey-dobbs/results/thesis_results/.

- Figure 2.1 and Figure 2.2 are photos taken by Benjamin White. The original copies can be found in the UW Pictures folder on Box.
- Table 2.1 and Table 2.2 were made in Overleaf. Information about the CNN layers came from the deep-learning-tank repo, in models/cnn1d.py. The parameter count was calculated and saved in deep-learning-tank/plotting-nbs/model_arch.ipynb.
- Figure 3.1: The validation result plot (Fig. 3.1a) is saved at data_quantity_tests/data_quantity_1800/days1a2a345b/run00/range_pred_vs_real.png and was created by running train_model.py with the settings toml saved at data_quantity_tests/data_quantity_1800/days1a2a345b/. The generalization result plot (Fig. 3.1b) is saved at data_quantity_1800/days1a2a345b/run00/pos_var_control_test/range_pred_vs_real.png and was created by running test_model.py, testing the model saved above on the bk_3ch_dx_control dataset (see tank_datasets.csv in corey-dobbs/results/thesis_results).

- Figure 3.2: all subfigures were plotted with deep-learning-tank/plots_dataq.ipynb. The .csv files containing the lists of the results reference where the data are saved. See individual .toml files for settings used to train each individual run. Training done with train_model.py
  - B&K8103 3-ch: results transcribed from all_test_stats.csv files found in bk_dq_3ch/data_quantity_tests/*
  - B&K8103 1-ch: results transcribed from all_test_stats.csv files found in bk_dq_1ch/bk_1ch/*
  - TC4038 3-ch: results transcribed from all_test_stats.csv files found in tc_dq_3ch/tc4038_data_quantity/*
  - TC 1-ch: results transcribed from all_test_stats.csv files found in tc_dq_1ch/tc_1ch/*
- The offset tests described at the end of Sec. 3.2 were performed on models found in bk_dq_3ch/data_quantity_tests/data_quantity_1800/.
- Table 3.1 and Figure 3.3 results are found in classification/classification/may02/. They were created by running train_model.py with the .toml files saved in the respective subfolders.
- Figure 3.4 were created by deep-learning-tank/plotting-nbs/noise_check.ipynb. The data used to create the plots are saved in /mnt/underwater/uw-measurements-tank/2023/2023-06-14/noise_robots_on/ and /mnt/underwater/uw-measurements-tank/2023/2023-06-15/noise_robots_on/.
- Figure 3.5 and Table 3.2.
  - Figure 3.5 is generated from data saved in thesis_results/tones_test_bk/. The code used to generate the plot is saved in plotting-nbs/plots_dataq.ipynb, saved in the deep-learning-tank repository.

  - Table 3.2, results for validation and generalization testing with data augmentation are saved in thesis_results/choose5/.
  - Table 3.2, results for validation and generalization testing without data augmentation are copied from the same results used for Figure 3.2c.

- Figure 3.6, Figure 3.7, and Table 3.3 show results for Transfer Learning. All results are saved in underwater/corey-dobbs/code/deep-learning-tank/.

  - Figure 3.6a is saved in transfer_learning/transfer_baseline/ at run02/dT_22b_largetext_test/range_pred_vs_real.png. It was created by running test_model.py; the model saved in transfer_learning/transfer_baseline was tested on on dataset bk_3ch_dT_22b.
  - Figure 3.6b is saved in transfer_learning/transfer_learning/ at test/run02/dT_22b_largetext_test/range_pred_vs_real.png. It was created by running test_model.py; the model saved in transfer_learning/transfer_learning/test was tested on on dataset bk_3ch_dT_22b.
  - Figure 3.7a is saved at transfer_learning/transfer_baseline/run04/dT_24b_test/range_pred_vs_real.png. It was created by running test_model.py; the model saved in transfer_learning/transfer_baseline was tested on on dataset bk_3ch_dT_24b.
  - Figure 3.7b is saved at transfer_learning/transfer_baseline/run04/dT_24b_test/range_pred_vs_real.png. It was created by running test_model.py; the model saved in transfer_learning/transfer_learning/dT_24a was tested on on dataset bk_3ch_dT_24b.

# Appendix B

# Code Repository

The majority of the code used in this Thesis comes from the GitHub repository deep-learning-tank, in the byu-panda-edu organization. Contact Dr. Tracianne Neilsen for access to the repository. The latest edition as of the publication of this Thesis was pushed to GitHub in commit ID 526adf9 on 24 May 2024. Basic information for using the repository to train and test deep learning models on data measured in the water tank of the Hydroacoustics Lab at BYU is provided in the README file associated with the repository.

The results are not uploaded to GitHub, to save space. The results are saved primarily in the underwater drive (owned by Dr. Neilsen and the BYU Physics and Astronomy Department), in underwater/corey-dobbs/results/thesis_results/.

The data are saved in two locations. The first is in the format of binary (.bin) files as saved by ESAU, in underwater/uw-measurements-tank/2023/, in subdirectories corresponding to the dates of the measurements. The specific subdirectories are listed in deep-learning-tank/data/tank_datasets.csv. The second location contains the data that have been preprocessed in preparation for deep learning, as .hdf5 files in underwater/data/measured/tank/2023/. Descriptions of what data are saved in each file are also given in deep-learning-tank/data/tank_datasets.csv. This is accurate for all measured tank data recorded with the purpose of deep learning, as of 24 May 2024.

# Bibliography

[1] M. A. Ainslie, "A Century of Sonar: Planetary Oceanography, Underwater Noise Monitoring, and the Terminology of Underwater Sound," Acoustics Today **11,** 12–19 (2015).

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the Association for Computing Machinery (ACM) **60,** 84–90 (2017).

[3] M. J. Bianco, P. Gerstoft, J. Traer, E. Ozanich, M. A. Roch, S. Gannot, and C.-A. Deledalle, "Machine learning in acoustics: Theory and applications," The Journal of the Acoustical Society of America **146,** 3590–3628 (2019).

[4] "Deep learning vs machine learning | Google Cloud — cloud.google.com,", https://cloud.google.com/discover/deep-learning-vs-machine-learning#section-6, [Accessed 24-05-2024].

[5] "What are Convolutional Neural Networks? | IBM — ibm.com,", https://www.ibm.com/topics/convolutional-neural-networks, [Accessed 24-05-2024].

[6] M. Mishra, "Convolutional Neural Networks, Explained — towardsdatascience.com,", https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939, 2020, [Accessed 24-05-2024].

[7] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," Mechanical Systems and Signal Processing **151,** 107398 (2021).

[8] R. Herchig, N. Palermo, P. Gerstoft, T. Daniel, and D. Sternlicht, "Comparing the Performance of Convolutional Neural Networks Trained to Localize Underwater Sound Sources," In *IEEE, OCEANS 2022, Hampton Roads*, pp. 1–7 (2022).

[9] B. M. Lee, J. R. Johnson, and D. R. Dowling, "Predicting Acoustic Transmission Loss Uncertainty in Ocean Environments with Neural Networks," Journal of Marine Science and Engineering **10,** 1548 (2022).

[10] X. Geng and B. Zhang, "Deep Q-Network-Based Intelligent Routing Protocol for Underwater Acoustic Sensor Network," IEEE Sensors Journal **23,** 3936–3943 (2023).

[11] S. Kong, Z. Wu, C. Qiu, M. Tian, and J. Yu, "An FM-Based Comprehensive Path Planning System for Robotic Floating Garbage Cleaning," IEEE Transactions on Intelligent Transportation Systems **23,** 23821–23830 (2022).

[12] P. Shi, H. Sun, Y. Xin, Q. He, and X. Wang, "SDNet: Image-based sonar detection network for multi-scale objects," IET Image Processing **17,** 1208–1223 (2022).

[13] M. Sung, Y.-W. Song, and S.-C. Yu, "Underwater object detection of AUV based on sonar simulator utilizing noise addition," In *2022 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*, **6,** 1–5 (2022).

[14] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," The Journal of the Acoustical Society of America **152,** 107–151 (2022).

[15] H. Niu, Z. Gong, E. Ozanich, P. Gerstoft, H. Wang, and Z. Li, "Deep-learning source localization using multi-frequency magnitude-only data," The Journal of the Acoustical Society of America **146,** 211–222 (2019).

[16] M. J. Bianco, S. Gannot, E. Fernandez-Grande, and P. Gerstoft, "Semi-Supervised Source Localization in Reverberant Environments With Deep Generative Modeling," IEEE Access **9,** 84956–84970 (2021).

[17] J. Jiang, Z. Wu, M. Huang, and Z. Xiao, "Detection of underwater acoustic target using beamforming and neural network in shallow water," Applied Acoustics **189,** 108626 (2022).

[18] J. Chi, X. Li, H. Wang, D. Gao, and P. Gerstoft, "Sound source ranging using a feed-forward neural network trained with fitting-based early stopping," The Journal of the Acoustical Society of America **146,** EL258–EL264 (2019).

[19] M. Goldwater, D. P. Zitterbart, D. Wright, and J. Bonnel, "Machine-learning-based simultaneous detection and ranging of impulsive baleen whale vocalizations using a single hydrophone," The Journal of the Acoustical Society of America **153,** 1094–1107 (2023).

[20] T. B. Neilsen, C. D. Escobar-Amado, M. C. Acree, W. S. Hodgkiss, D. F. Van Komen, D. P. Knobles, M. Badiey, and J. Castro-Correa, "Learning location and seabed type from a moving mid-frequency source," The Journal of the Acoustical Society of America **149,** 692–705 (2021).

[21] Z. Huang, J. Xu, Z. Gong, H. Wang, and Y. Yan, "Source localization using deep neural networks in a shallow water environment," The Journal of the Acoustical Society of America **143,** 2922–2932 (2018).

[22] Z. Yang, T. Shen, M. Cui, Z. Luo, X. Li, and Q. Zhao, “Source localization in deep ocean based on complex convolutional neural network,” Journal of Physics: Conference Series **2718,** 012096 (2024).

[23] E. L. Ferguson, S. B. Williams, and C. T. Jin, “Convolutional neural network for single-sensor acoustic localization of a transiting broadband source in very shallow water,” The Journal of the Acoustical Society of America **146,** 4687–4698 (2019).

[24] S. Whitaker, A. Barnard, G. D. Anderson, and T. C. Havens, “Recurrent networks for direction-of-arrival identification of an acoustic source in a shallow water channel using a vector sensor,” The Journal of the Acoustical Society of America **150,** 111–119 (2021).

[25] A. Vardi and J. Bonnel, “End-to-End Geoacoustic Inversion With Neural Networks in Shallow Water Using a Single Hydrophone,” IEEE Journal of Oceanic Engineering **49,** 380–389 (2024).

[26] W. Liu, Y. Yang, M. Xu, L. Lü, Z. Liu, and Y. Shi, “Source localization in the deep ocean using a convolutional neural network,” The Journal of the Acoustical Society of America **147,** EL314–EL319 (2020).

[27] H. Niu, E. Reeves, and P. Gerstoft, “Source localization in an ocean waveguide using supervised machine learning,” The Journal of the Acoustical Society of America **142,** 1176–1188 (2017).

[28] H. Niu, E. Ozanich, and P. Gerstoft, “Ship localization in Santa Barbara Channel using machine learning classifiers,” The Journal of the Acoustical Society of America **142,** EL455–EL460 (2017).

[29] S. Kita and Y. Kajikawa, “Fundamental study on sound source localization inside a structure using a deep neural network and computer-aided engineering,” Journal of Sound and Vibration **513,** 116400 (2021).

[30] M. Liu, H. Niu, and Z. Li, “Implementation of Bartlett matched-field processing using interpretable complex convolutional neural network,” JASA Express Letters **3,** 026003 (2023).

[31] A. Weiss, T. Arikan, and G. W. Wornell, “Direct localization in underwater acoustics via convolutional neural networks: A data-driven approach,” In *2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6 (2022).

[32] D. F. Van Komen, T. B. Neilsen, K. Howarth, D. P. Knobles, and P. H. Dahl, “Seabed and range estimation of impulsive time series using a convolutional neural network,” The Journal of the Acoustical Society of America **147,** EL403–EL408 (2020).

[33] D. F. Van Komen, T. B. Neilsen, D. B. Mortenson, M. C. Acree, D. P. Knobles, M. Badiey, and W. S. Hodgkiss, “Seabed type and source parameters predictions using ship spectrograms in convolutional neural networks,” The Journal of the Acoustical Society of America **149,** 1198–1210 (2021).

[34] H. I. Hummel, R. van der Mei, and S. Bhulai, “A survey on machine learning in ship radiated noise,” Ocean Engineering **298,** 117252 (2024).

[35] C. D. Escobar-Amado, T. B. Neilsen, J. A. Castro-Correa, D. F. Van Komen, M. Badiey, D. P. Knobles, and W. S. Hodgkiss, “Seabed classification from merchant ship-radiated noise using a physics-based ensemble of deep learning algorithms,” The Journal of the Acoustical Society of America **150,** 1434–1447 (2021).

[36] J. Piccolo, G. Haramuniz, and Z.-H. Michalopoulou, “Geoacoustic inversion with generalized additive models,” The Journal of the Acoustical Society of America **145,** EL463–EL468 (2019).

[37] C. Frederick and Z.-H. Michalopoulou, “Seabed classification and source localization with Gaussian processes and machine learning,” JASA Express Letters **2,** 084801 (2022).

[38] A. Varon, J. Mars, and J. Bonnel, “Approximation of modal wavenumbers and group speeds in an oceanic waveguide using a neural network,” JASA Express Letters **3,** 066003 (2023).

[39] J. Yangzhou, Z. Ma, and X. Huang, “A deep neural network approach to acoustic source localization in a shallow water tank experiment,” The Journal of the Acoustical Society of America **146,** 4802–4811 (2019).

[40] R. Lefort, G. Real, and A. Drémeau, “Direct regressions for underwater acoustic source localization in fluctuating oceans,” Applied Acoustics **116,** 303–310 (2017).

[41] B.-J. Lin, P.-C. Guan, H.-T. Chang, H.-W. Hsiao, and J.-H. Lin, “Application of a Deep Neural Network for Acoustic Source Localization Inside a Cavitation Tunnel,” Journal of Marine Science and Engineering **11,** 773 (2023).

[42] W. Raza, X. Ma, H. Song, A. Ali, H. Zubairi, and K. Acharya, “Long Short-Term Memory Neural Network assisted Peak to Average Power Ratio Reduction for Underwater Acoustic Orthogonal Frequency Division Multiplexing Communication,” KSII Transactions on Internet and Information Systems **17,** 239–260 (2023).

[43] J. Brownlee, “A Gentle Introduction to Transfer Learning for Deep Learning - MachineLearningMastery.com — machinelearningmastery.com,”, https://machinelearningmastery.com/transfer-learning-for-deep-learning/, 2019, [Accessed 24-05-2024].

[44] W. Wang, H. Ni, L. Su, T. Hu, Q. Ren, P. Gerstoft, and L. Ma, “Deep transfer learning for source ranging: Deep-sea experiment results,” The Journal of the Acoustical Society of America **146,** EL317–EL322 (2019).

[45] F.-X. Ge, Y. Bai, M. Li, G. Zhu, and J. Yin, “Label distribution-guided transfer learning for underwater source localization,” The Journal of the Acoustical Society of America **151,** 4140–4149 (2022).

[46] R. Long, N. Liang, J. Zhou, Y. Yang, and X. Hou, "Range estimation in a weakly range-dependent waveguide in the South China Sea using single hydrophone and multi-task trained deep neural network," Applied Acoustics **213,** 109630 (2023).

[47] Z. Wang, Q. Li, J. Zhu, Q. Li, Z. Juan, and Q. Tong, "Seawater Temperature Profile Reconstruction Based on Transfer Learning," In *2023 IEEE 6th International Conference on Information Communication and Signal Processing (ICICSP)*, (2023).

[48] Y. Liu, H. Niu, Z. Li, and D. Zhai, "Unsupervised Domain Adaptation for Source Localization Using Ships of Opportunity With a Deep Vertical Line Array," IEEE Journal of Oceanic Engineering **49,** 180–196 (2024).

[49] C. T. Vongsawad, T. B. Neilsen, A. D. Kingsley, J. E. Ellsworth, B. E. Anderson, K. N. Terry, C. E. Dobbs, S. E. Hollingsworth, and G. H. Fronk, "Design of an underwater acoustics lab," In *Proceedings of Meetings on Acoustics*, **45,** 070005 (2021).

[50] N. L. Weinberg and W. G. Grantham, "Development of an Underwater Acoustics Laboratory Course," The Journal of the Acoustical Society of America **49,** 697–705 (1971).

[51] A. D. Kingsley, J. M. Clift, B. E. Anderson, J. E. Ellsworth, T. J. Ulrich, and P.-Y. L. Bas, "Development of software for performing acoustic time reversal with multiple inputs and outputs," In *Proceedings of Meetings on Acoustics*, **46,** 055003 (2022).

[52] C. T. Vongsawad, "Development and Characterization of an Underwater Acoustics Laboratory Via in situ Impedance Boundary Measurements," Thesis Archive, Brigham Young University (2021).

[53] K. Howarth, T. B. Neilsen, D. F. Van Komen, and D. P. Knobles, "Seabed Classification Using a Convolutional Neural Network on Explosive Sounds," IEEE Journal of Oceanic Engineering **47,** 670–679 (2022).

[54] "ML Practicum: Image Classification | Machine Learning | Google for Developers — developers.google.com,", https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks, [Accessed 29-05-2024].

[55] "Max Pooling — deepai.org,", https://deepai.org/machine-learning-glossary-and-terms/max-pooling, [Accessed 29-05-2024].

[56] R. Müller, S. Kornblith, and G. E. Hinton, "When Does Label Smoothing Help?," Computing Research Repository (CoRR), arxiv abs/1906.02629 (2020).

[57] "PyTorch documentation &x2014; PyTorch 2.3 documentation — pytorch.org,", https://pytorch.org/docs/stable/index.html, [Accessed 29-05-2024].

[58] I. Loshchilov and F. Hutter, "Fixing Weight Decay Regularization in Adam," Computing Research Repository (CoRR), arXiv abs/1711.05101 (2017).

[59] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Restarts," Computing Research Repository (CoRR), arxiv abs/1608.03983 (2016).

[60] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," Journal of Big Data **6,** 60 (2019).

[61] J. A. Castro-Correa, M. Badiey, T. B. Neilsen, D. P. Knobles, and W. S. Hodgkiss, "Impact of data augmentation on supervised learning for a moving mid-frequency source," The Journal of the Acoustical Society of America **150,** 3914–3928 (2021).

[62] D. Freer and G.-Z. Yang, "Data augmentation for self-paced motor imagery classification with C-LSTM," Journal of Neural Engineering **17,** 016041 (2020).

[63] R. Kalantar, "How to Freeze Model Weights in PyTorch for Transfer Learning: Step-by-Step Tutorial — python.plainenglish.io,", https://python.plainenglish.io/

how-to-freeze-model-weights-in-pytorch-for-transfer-learning-step-by-step-tutorial-a533a58051ef, 2023, [Accessed 24-05-2024].