ACTIVE BAYESIAN DEEP LEARNING WITH AN ACOUSTIC VECTOR SENSOR

Atchley, Sabrina L.
Monterey, CA; Naval Postgraduate School

https://hdl.handle.net/10945/73045

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun
THESIS

ACTIVE BAYESIAN DEEP LEARNING
WITH AN ACOUSTIC VECTOR SENSOR

by

Sabrina L. Atchley

September 2021

Thesis Advisor: Marko Orescanin
Second Reader: Vinnie Monaco

Approved for public release. Distribution is unlimited.
### Abstract

Traditional passive monitoring of the ocean’s acoustic signals is conducted with an omnidirectional hydrophone, which only measures acoustic pressure. Vector sensors, unlike hydrophones, respond to both the acoustic pressure and the vector motion of water, providing additional information. This thesis focuses on utilizing vector sensor data as input to a neural network and studies the advantage of utilizing all four channels over single-channel data from the acoustic pressure sensor. A Bayesian deep learning approach is used to build multi-class classification models that provide estimates of uncertainty. The best model had an F1 score of .798 using single-channel data and .81 when using four-channel data from the vector sensor. However, the addition of information from the four-channel signal significantly reduces predictive uncertainty, demonstrating the advantage of utilizing all four channels for passive sonar classification. Next, active learning is examined, an algorithm that typically depends on uncertainty estimates to select the best training data. This is likely the first study on active learning with Bayesian deep learning models in passive sonar classification. With active learning using 23% of the training data, we trained within two percent of the F1 score compared to the entire training data. Additionally, the active learning experiments demonstrated that uncertainty-based acquisition functions increased performance using four channels over single-channel data.
ACTIVE BAYESIAN DEEP LEARNING
WITH AN ACOUSTIC VECTOR SENSOR

Sabrina L. Atchley
Lieutenant, United States Navy
BS, North Carolina State University, 2009

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE
from the
NAVAL POSTGRADUATE SCHOOL
September 2021

Approved by:  Marko Orescanin
Advisor

Vinnie Monaco
Second Reader

Gurminder Singh
Chair, Department of Computer Science
ABSTRACT

Traditional passive monitoring of the ocean’s acoustic signals is conducted with an omnidirectional hydrophone, which only measures acoustic pressure. Vector sensors, unlike hydrophones, respond to both the acoustic pressure and the vector motion of water, providing additional information. This thesis focuses on utilizing vector sensor data as input to a neural network and studies the advantage of utilizing all four channels over single-channel data from the acoustic pressure sensor. A Bayesian deep learning approach is used to build multi-class classification models that provide estimates of uncertainty. The best model had an F1 score of .798 using single-channel data and .81 when using four-channel data from the vector sensor. However, the addition of information from the four-channel signal significantly reduces predictive uncertainty, demonstrating the advantage of utilizing all four channels for passive sonar classification. Next, active learning is examined, an algorithm that typically depends on uncertainty estimates to select the best training data. This is likely the first study on active learning with Bayesian deep learning models in passive sonar classification. With active learning using 23% of the training data, we trained within two percent of the F1 score compared to the entire training data. Additionally, the active learning experiments demonstrated that uncertainty-based acquisition functions increased performance using four channels over single-channel data.
Table of Contents

1 Introduction .......................... 1
   1.1 Research Objectives and Contributions ................................... 3
   1.2 Organization ................................................................. 5

2 Background ............................ 7
   2.1 Neural Networks ............................................................ 7
   2.2 Bayesian Deep Learning .................................................... 8
   2.3 Active Learning ............................................................. 14
   2.4 Previous Work in Active Bayesian Deep Learning ...................... 17

3 Methodology ............................ 19
   3.1 Datasets ............................................................... 19
   3.2 Deep Learning Models .................................................... 22
   3.3 Bayesian Deep Learning Implementation ............................... 25
   3.4 Performance Metrics ..................................................... 28
   3.5 Experiment Methodology .................................................. 29

4 Results .................................. 33
   4.1 Full Dataset Results ...................................................... 33
   4.2 Case Study: Ship Track Predictions .................................... 39
   4.3 Active Learning Results .................................................. 43

5 Conclusion .............................. 47
   5.1 Future Work ............................................................... 48

List of References ........................ 51

Initial Distribution List .................. 57
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1</td>
<td>Neural Network and Bayesian Neural Networks</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Uncertainty Variance Visualization</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Pool-Based Active Learning</td>
<td>15</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>MNIST Active Learning</td>
<td>17</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>GeoSpectrum M20-105</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Mel-Log Spectrograms</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Custom CNN Model Architecture</td>
<td>23</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>ResNet Residual Block</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>KL Annealing</td>
<td>27</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>KL Reweighing</td>
<td>28</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>BDL Uncertainty Metrics</td>
<td>35</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>MARS Epistemic and Aleatoric Uncertainty</td>
<td>36</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>BDL Uncertainty Measures</td>
<td>37</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>BDL Uncertainty Measures: Single Compared to Four Channel</td>
<td>38</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>MARS Range Versus Time: One Channel</td>
<td>40</td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>MARS Range Versus Time: Four Channel</td>
<td>41</td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>MARS Geographic AIS Location Plot: One Channel</td>
<td>42</td>
</tr>
<tr>
<td>Figure 4.8</td>
<td>MARS Geographic AIS Location Plot: Four Channel</td>
<td>43</td>
</tr>
<tr>
<td>Figure 4.9</td>
<td>Active Learning: 100 Iterations</td>
<td>44</td>
</tr>
<tr>
<td>Figure 4.10</td>
<td>Active Learning: 400 Iterations</td>
<td>45</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1</td>
<td>Dataset Class List</td>
<td>20</td>
</tr>
<tr>
<td>Table 3.2</td>
<td>ResNet Size Search</td>
<td>24</td>
</tr>
<tr>
<td>Table 4.1</td>
<td>MARS Full Data Custom Model</td>
<td>33</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>MARS Full Data ResNet Model</td>
<td>34</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>BDL Uncertainty Filtering Use Case</td>
<td>39</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Active Learning: 1 Versus 4 Channel</td>
<td>46</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>AI</td>
<td>artificial intelligence</td>
<td></td>
</tr>
<tr>
<td>AIS</td>
<td>automatic identification system</td>
<td></td>
</tr>
<tr>
<td>BALD</td>
<td>Bayesian Active Learning by Disagreement</td>
<td></td>
</tr>
<tr>
<td>BDL</td>
<td>Bayesian deep learning</td>
<td></td>
</tr>
<tr>
<td>CNN</td>
<td>convolutional neural network</td>
<td></td>
</tr>
<tr>
<td>DOD</td>
<td>Department of Defense</td>
<td></td>
</tr>
<tr>
<td>ELBO</td>
<td>evidence lower bound</td>
<td></td>
</tr>
<tr>
<td>KL</td>
<td>Kullback-Leibler</td>
<td></td>
</tr>
<tr>
<td>MARS</td>
<td>Monterey Accelerated Research System</td>
<td></td>
</tr>
<tr>
<td>MC</td>
<td>Monte Carlo</td>
<td></td>
</tr>
<tr>
<td>ML</td>
<td>machine learning</td>
<td></td>
</tr>
<tr>
<td>ResNet</td>
<td>residual network</td>
<td></td>
</tr>
<tr>
<td>STFT</td>
<td>Short Time Fourier Transform</td>
<td></td>
</tr>
<tr>
<td>TFP</td>
<td>TensorFlow Probability</td>
<td></td>
</tr>
</tbody>
</table>
First of all, I would like to express my gratitude to Dr. Marko Orescanin, my thesis advisor, for his advice, mentoring, and encouragement throughout the thesis process.

Additionally, I would like to thank LT Andrew Pfau, USN, a previous NPS student whose thesis was the basis of this work. Thanks also go to LCDR Brandon Beckler, USN, whose thesis was on a similar topic, for the great teamwork and friendship. I am grateful we were able to work together on many projects.

Thank you to my wife, Amanda Atchley, for constantly listening to me rant and complain, for proofreading this over and over, and for her love and support. Special thanks to my daughter, Juliet, for patience and understanding while I was focused on completing this thesis and my degree.
THIS PAGE INTENTIONALLY LEFT BLANK
CHAPTER 1: Introduction

This chapter is adapted from [1], previously published by the IEEE Journal of Oceanic Engineering, ©2021 IEEE

The field of artificial intelligence (AI) is transforming every aspect of human life and has applications in numerous sectors, including finance, national security, health care, criminal justice, transportation, and smart cities [2]. A subfield of AI, machine learning (ML) is when algorithms learn from data, and it is commonly used for classification and regression tasks. As storage systems, processing speeds, and analytic techniques improve dramatically, AI algorithms are becoming increasingly sophisticated in their analysis and decision-making capabilities [2]. The U.S. Department of Defense (DOD) is aware of the technological advantage that AI will bring to future conflicts. The “2018 Department of Defense Artificial Intelligence Strategy” [3] states that “AI is poised to transform every industry and is expected to impact every corner of the Department, spanning operations, training, sustainment, force protection, recruiting, healthcare, and many others.”

Adversaries of the United States, especially China and Russia, are investing heavily in AI technology for military reasons, including applications that raise concerns about international standards and human rights; these expenditures risk eroding the United States’ technical and operational advantages [3]. To be successful in future conflicts and acquire

---

1 Reprinted, with permission, from Orescanin et al., “Active Bayesian Deep Learning With Vector Sensor for Passive Sonar Sensing of the Ocean,” IEEE Journal of Oceanic Engineering, MON 2021. This publication is a work of the U.S. government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States. IEEE will claim and protect its copyright in international jurisdictions where permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

2 In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of the Naval Postgraduate School’s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.
a strategic advantage, the United States military and allies must invest and innovate within the field of AI [3]. The DOD has adopted five ethical principles for the use of AI. These principles state that AI systems should be responsible, equitable, traceable, reliable, and governable [4]. However, China and Russia are less concerned with the legal and ethical debate on the usage of AI in warfare [2]. While having ethical requirements for the use of AI is the right thing to do, the lack of restrictions could provide China and Russia with faster automated decision making.

When ML algorithms are used in mission-critical systems, the acceptable margin of error shrinks dramatically. In these situations, it is important that there is confidence in the model’s predictions before acting on them. Understanding the uncertainty of a ML model’s predictions could provide additional confidence in taking action based on ML outputs and build more reliable systems. However, uncertainty is not captured well in the deterministic neural network models commonly used across many applications.

Using ML and AI, the DOD could develop and leverage intelligent sensors. These sensors can perform tasks such as classification, anomaly detection, and threat detection. One such application is an underwater acoustic sensor, a device that could have several applications across the DOD and the U.S. Navy, including antisubmarine warfare, intelligent mines, and harbor security.

Traditional passive monitoring of the ocean’s acoustic signals is conducted with a hydrophone, which is a pressure sensor that measures the scalar acoustic pressure and are typically omnidirectional instruments. Vector sensors, unlike hydrophones, respond to both the acoustic pressure and the vector motion of water at the same time (fluid velocity or acceleration) [5]. As a result, vector sensors provide additional information that can be used as input into a machine learning model [5]. Typically, the advantage of using a vector sensor over a hydrophone in underwater acoustic applications is the ability to estimate the direction of arrival of sound waves [6], [7]. Significant work over the last two decades was reported on vector sensor signal processing in the context of combining four input channels to maximize the target detection while mitigating the adverse effects of signal degradation in complex interfering multi-path/multi-target environments.

Bayesian deep learning (BDL) is a method of building deep neural networks with a Bayesian approach. Weights are assigned as a single value or point estimate in standard neural
networks, while in BDL, weights are interpreted as a probability distribution. However, inferring the probability distributions is intractable because of their vast parameter space found in deep neural networks but can be approximated using variational inference and Monte Carlo (MC) dropout methods [8]. One of the primary motivations for BDL is to observe and express a realistic estimate of uncertainty, which is an ability traditional neural networks lack, leading to overconfident decisions, predictions, or actions [8], [9]. There are two types of uncertainty; aleatoric and epistemic. Aleatoric uncertainty provides information about the noise inherent in the observations, such as noise in a sensor. Epistemic uncertainty accounts for uncertainty in the model parameters and can be reduced with additional data. Additionally, BDL has proven to be less prone to overfitting during training [9]. Overfitting refers to when a model performs well on the data the model was trained on but has performance issues when confronted with previously unseen data.

Active learning is an ML method that may allow a model to be trained with less data. This is done by allowing the algorithm to select the data from which it learns [10]. One of the primary uses of active learning is in scenarios where unlabeled data is abundant or readily available but training labels are difficult, time consuming, or expensive to obtain [10]. The way the algorithms select or query for data is called an acquisition function, and they often rely on uncertainty [10], [11]. Traditional deep learning models rarely capture such model uncertainty; nevertheless, recent developments in BDL with improved uncertainty estimations that have enabled active learning when using deep neural networks [11].

1.1 Research Objectives and Contributions
This work builds on Lieutenant Andrew Phau’s thesis “Multi-Label Classification of Underwater Soundscapes Using Deep Convolutional Neural Networks” [5]. Phau’s thesis demonstrated a custom convolutional neural network (CNN) model explicitly designed for underwater acoustic classification and provided an automated method of labeling data from acoustic sensors. In this thesis, the focus is on utilizing vector sensor data as an input to a neural network model for multi-class classification and studies the advantage of utilizing all four channels of the vector sensor in place of utilizing only the information from the scalar acoustic pressure. This is accomplished by benchmarking both deterministic and Bayesian configurations of deep residual network (ResNet) models [12], [13] architectures and custom CNN models. The thesis also examines the benefits of the vector sensor when used in both
Bayesian and deterministic architectures and shows the uncertainty of ship classification inference to be a distinct improvement of Bayesian architecture for underwater soundscape classification with the additional information provided by an acoustic vector sensor. Finally, the active learning enabled by BDL is evaluated using both a single and four channels from the vector sensor, and an improvement in uncertainty-based acquisition functions is shown. Specific research questions are:

• Is there an advantage of utilizing all four channels of the vector sensor in place of utilizing only the information from the scalar acoustic pressure as input to an ML model for classification for both deterministic and Bayesian models?
• Do the BDL models utilizing acoustic vector sensor data improve active learning performance? This could be a prototype for a future active learning framework for deployment on a live sensor.

Key contributions of this research include the first instance, to the author’s knowledge, of active learning on underwater acoustic signals using BDL and the usage of epistemic uncertainty as an acquisition function. With active learning using 23% of the training data, we trained within two percent of the $F^1$ score compared to the entire training data. In addition to active learning, the advantages of utilizing all four channels of the vector sensor rather than using only the information from the scalar acoustic pressure show a slight classification improvement. The best model had an $F^1$ score of .798 using single-channel data and .81 when using four-channel from the vector sensor. However, the addition of information from the four-channel signal significantly reduces predictive uncertainty, demonstrating the advantage of utilizing all four channels for passive sonar classification. This decrease in predictive uncertainty indicates more confidence in the model output leading to potentially more trust and reliability in underwater ML acoustic classification systems.
1.2 Organization

This thesis is organized into four additional chapters. Chapter 2 introduces BDL, active learning concepts, and existing relevant literature. Chapter 3 discusses the dataset and CNN models used in this thesis, the research methodology, and BDL and active learning implementation details. Chapter 4 examines the study’s outcomes, including metrics, observations, efficiency, and accuracy. Chapter 5 discusses the research conclusions, future work, and final observations.
CHAPTER 2: Background

This chapter is adapted from [1], previously published by the IEEE Journal of Oceanic Engineering. ©2021 IEEE

This chapter explores several vital technical concepts relevant to neural networks, BDL, and active learning utilized throughout this work. First, there is a brief discussion of deterministic neural networks and their limitations. Following that, there will be an introduction to BDL theory and a review of two distinct implementation techniques: variational inference and MC dropout. After that, we will discuss how to use uncertainty estimates from BDL models to accomplish active learning.

2.1 Neural Networks

Neural networks have been used to solve many complex problems in ML and AI, including regression and classification tasks. Some examples of the usage of neural networks include computer vision, business problems, and language recognition [14]. Inspired by biology, neural networks are built using many artificial neurons or nodes, with each node having an input, a weight, and bias, and an activation function that produces an output [14]. The nodes are then stacked into layers to build a network containing an input layer, a number of hidden layers, and an output layer. An artificial neural network with many hidden layers is referred to as a deep neural network or deep learning [15]. The weight and biases are the parameters that are trainable and must be optimized. The calculations required to optimize the parameters during training are well-suited for graphics processing units and customized machine learning hardware.

A common method of training a neural network is supervised learning. A training set is used in supervised learning to teach models to produce the desired output. This training dataset contains example labeled data. The loss function is used to assess the algorithm’s correctness, and it is adjusted until the error is suitably minimized. Stochastic gradient descent and the back-propagation of the loss values are used to optimize the trainable parameters of a neural network model. Increasing the number of neural layers can solve more
complex problems; however, increasing the number of layers results in more parameters, which expands both the training time and computational cost.

Traditional methods of implementing and training neural networks result in single value point estimates of the weights and bias parameters in the neural network. The use of point estimates results in deterministic models; that is, the model will produce the same output for a given input. Neural networks using point estimates will be referred to as deterministic neural networks from here on. While deterministic neural networks have been able to accomplish some impressive tasks, they are not without problems. Deterministic neural networks are often unable to accurately measure uncertainty, resulting in unnecessarily optimistic decisions from the model, and when using training data with supervised learning, overfitting is a concern. A possible improvement to deterministic neural networks is to take a Bayesian approach when training neural networks.

ML has been studied for the classification of acoustic data, including underwater soundscapes [16]. However, much of the research focuses on the classification of natural biological sounds and not the acoustic signals from ships [17]. Passive sonar classification of man-made vessels is a challenge because there is a lack of available datasets [17]. Additionally, the type and strength of signals received by a sensor are influenced by the ships’ size, speed, bearing, range, and water conditions, such as temperature, salinity, and pressure [18].

2.2 Bayesian Deep Learning

Using a probabilistic approach to deep learning allows for the estimation of uncertainty, enabling users to identify predictions that are more likely to be incorrect. Sources of uncertainty can be found in the data, such as measurement errors or label noise, or in the model, such as a lack of data for the model to learn effectively. Using a Bayesian approach, the model learns probabilistic distributions for the model parameters from which we can sample to produce an output for a given input, rather than learning specific weights and bias values (see Figure 2.1) [9]. When predictions are performed using the neural network on the right of Figure 2.1, weight values are sampled from the probability distributions that have been learned during training. This process results in an infinite number of networks, with different results for each prediction using the non-deterministic network. In cases where the network is more certain, there will be less variation in the results.
BDL is achieved by assuming distributions over the weights of a neural network architecture in contrast to a deterministic architecture in which each weight is a single scalar. We assume a supervised learning setting with $D = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{R}^d$ and $y_i \in \{1, 2, \ldots, c\}$; $N$ denotes sample size; $d$ is the dimension of the input feature space and, $c$ is the number of different classes. In Bayesian statistics, all inference about unknown quantities involves the calculation of the posterior. By placing a prior distribution $p(\omega)$ on a parameter vector $\omega \in \Omega$, where $\Omega$ represents all weights and biases of a neural network architecture, the resulting posterior distribution of the neural network weights upon training follows Bayes’ rule:

$$
p (\omega|D) = \frac{p (D|\omega) p(\omega)}{p(D)} \tag{2.1}
$$

In prediction, given a new input feature vector (e.g., test data) $x^*$ probability of the model prediction or inference, $y^*$, can be calculated with:

$$
p (y^*|x^*, D) = \int_{\Omega} p (y^*|x^*, \omega) p (\omega|D) \tag{2.2}
$$

Denoting by $f^\omega = (f_1^\omega \ldots f_c^\omega)$, the last $c$-dimensional linear output of a neural network
with parameter vector $\omega$, the predictive probability is given with:

$$p(y^* = c \mid x^*, \omega) = p(y^* = c \mid f^\omega(x)) = \text{Softmax}(f^\omega(x))$$

(2.3)

However, in practice, the inference defined by Equation 2.2 is intractable due to the calculation of the $p(\omega \mid \mathcal{D})$. Approximate inference is used instead. Two main approaches toward approximate inference are variational inference and sampling methods [19]. Variational inference methods are preferred as faster and scalable with large data over sampling methods such as Markov chain Monte Carlo [20]. Due to recent theoretical advancements in variational inference and the availability of computation, it is applied as approximate Bayes’ approach [21]. In approximate Bayes’ approach, probability densities are approximated via optimization. In general, one would specify prior distributions and approximate posterior distributions over the weights of the neural network with a tractable distribution $q_\theta(\omega)$ indexed by a variational parameter $\theta$. The main idea being that the family of densities are suggested and then, via optimization, a member of that family is found. Therefore, the object of optimization is to identify the closest density to the posterior among the family of pre-determined (target density) distributions $Q = q_\theta(\omega)$. It is common in the information-theoretic approach to measure proximity between the two densities, $q_\theta(\omega)$ and $p(\omega \mid \mathcal{D})$ via Kullback-Leibler (KL) divergence:

$$KL\{q_\theta(\omega) \mid \mid p(\omega \mid \mathcal{D})\} := \int_\Omega q_\theta(\omega) \log \frac{q_\theta(\omega)}{p(\omega \mid \mathcal{D})} d\omega \quad (2.4)$$

Minimizing the KL divergence is equivalent to minimizing the negative evidence lower bound (ELBO):

$$-\int_\Omega q_\theta(\omega) \log \frac{q_\theta(\omega)}{p(\omega \mid \mathcal{D})} d\omega + KL\{q_\theta(\omega) \mid \mid p(\omega \mid \mathcal{D})\} \quad (2.5)$$

Through variational inference, the integration problem is converted to an optimization problem with the following form of the ELBO loss:
\[ \mathcal{L}_q = \mathbb{E}_q [\log p(D | \omega)] - \text{KL}(q(\omega) \parallel p(\omega)) \] 

(2.6)

where \( E \) represents the expected value.

Given that the posterior is approximated by \( q(\omega) \) via variational inference, the quality of approximation is dependent on the choice of the family of distributions \( Q \).

Another approach is MC dropout from Gal and Gharmanani [22]. The use of dropout layers is a training approach that involves the random selection of neurons to exclude from the neural network during the training phase to avoid overfitting [23]. MC dropout introduces dropout layers in inference and is equivalent to Bernoulli approximation of the posterior \( q_\theta \) over weights.

\[ p(y = c|x) = \int p(y = c|x, \omega)p(w)d\omega \] 

(2.7)

Equation 2.7 is a Bayesian neural network with an input of \( x \). To get an output class \( c \), we must integrate over all possible weights:

\[ \approx \int p(y = c|x, \omega)q^*(w)d\omega \] 

(2.8)

Since the \( p(w) \) is unknown, it is assumed to belong to a Bernoulli distribution that is simulated by having dropout layers active during inference:

\[ \approx \frac{1}{T} \sum_{t} p(y|x, \omega_t) = \frac{1}{T} \sum_{t} p^t_c \] 

(2.9)

MC integration is used by running the sample through the network many times and summing the results. A larger \( T \) results in a better approximation.

In this work, we consider two approaches: mean field approximation via Gaussian posterior (and prior) with flipout MC estimator of KL divergence [24], and so-called MC dropout.
approach with Bernoulli prior over the weights [22]. Mean field approximation approach with normal distributions doubles the number of parameters in the model architecture due to mean and variance, making the optimization problem more challenging relative to the MC dropout. MC dropout requires minimal changes to the architecture of the network with no additional increase in the number of weights but an increase in the hyperparameter space.

2.2.1 Uncertainty Measure in Bayesian Deep Learning
Uncertainty of a neural network classifier trained using BDL techniques can be estimated with predictive entropy and total variance [25]. A standard metric for measuring uncertainty is predictive entropy, which calculates the average information encompassed by the predictive distribution and can be expressed by:

\[
\mathcal{H}_p(y \mid x) = - \sum_c p(y = c \mid x) \log p(y = c \mid x)
\]

where \(c\) represents all possible classes. Predictive probability \(p(y = c \mid x)\) is approximated by \(T\) MC samples such that for each class \(c\) the mean probability can be determined as:

\[
\bar{p}_c = \frac{1}{T} \sum_{t=1}^{T} \hat{p}_{ct}
\]

Model inputs are classified using Equation 2.11 by assigning the class with the highest mean probability:

\[
p^*_p = \max_c \bar{p}_c
\]

Total variance, the sum of the variances of individual classes, could also be used as a measure of uncertainty, given with:

\[
V_{tot}(y \mid x) = \sum_c \frac{1}{T} (\hat{p}_{ct} - \bar{p}_c)^2
\]
Figure 2.2 shows an example plot with the predicted probabilities from 100 samples from a BDL model. The left image has low variance and estimates low uncertainty in the predictions, while the example used on the right has a high estimate of uncertainty due to the larger variance.

In BDL, there are two types of uncertainty that can be estimated, aleatoric and epistemic. Kendal and Gal [26] proposed a novel approach to estimate aleatoric and epistemic uncertainties separately for regression applications. Aleatoric uncertainty provides information about the noise inherent in the observations, such as electronic noise on hydrophone input. Aleatoric error cannot be reduced even if more data were to be collected (e.g., noise on sensor input). Epistemic uncertainty, on the other hand, accounts for uncertainty in the model parameters. Given enough data, this uncertainty is explainable and can be reduced. The approach by Kendal and Gal was later revisited by Kwon et al. [27] for classification tasks where they proposed the following uncertainty estimator:

\[
\frac{1}{T} \sum_{t=1}^{T} \text{diag}(\hat{\rho}_{c_t}) - \hat{\rho}_{c_t}^2 + \frac{1}{T} \sum_{t=1}^{T} (\text{diag}(\hat{\rho}_{c_t}) - \hat{\rho}_c)^2
\]

(2.14)
2.3 Active Learning

The main idea behind active learning in machine learning applications is to query data points to train on [10], [28]. This can lead to better performance of the trained models with less training if the algorithm is allowed to select the data to train on. The cost of labeling data can be time consuming and expensive. Therefore, active learning can be used as a method to select the best data to be labeled by a human. Selection of data to train on is achieved by using acquisition functions. Given that many acquisition functions are uncertainty based [11], they are not well suited for usage with deterministic deep learning models. In this work, we focus on studying the quality and impact of different uncertainty metrics provided by BDL approaches to implement active learning using passive sonar data with deep neural networks.

2.3.1 Scenarios

In many applications, especially those pertaining to remote sensing, active learning is adopted as a strategy to help mitigate the lack of labeled data. However, given the proposed application of enhancing the sonar data with automatic identification system (AIS), the goal is different. Given that one can have very large volumes of labeled data, it is inefficient and expensive to train on all available labeled data; instead, we focus on using active learning to build a dataset that provides the best information gain in training.

In an online active learning setting with deep learning models, each sample is considered for training as it is acquired, which is called stream-based sampling [10]. However, for many applications, live stream-based sampling is impractical, and often a pool of candidate samples is considered.
The typical pool-based active learning cycle is shown in Figure 2.3. The model will first be trained on a set of labeled training data, then the algorithm will select the best data from a large pool of unlabeled data using an acquisition function, followed by querying an oracle to label the data and finally, adding the new labeled data to the training pool and repeating. The oracle can be a human or other automated system to generate labels for the data.

### 2.3.2 Acquisition Functions

This work focuses on uncertainty sampling, using uncertainty-based acquisition functions to select additional data for training. The key concept of uncertainty sampling is “that the learner can avoid querying the instances it is already confident about, and focus its attention instead on the unlabeled instances it finds confusing” [10]. Mathematically, this data sampling can be described as [11]:

\[
x^* = \arg\max_{x \in D_{pool}} a(x, \mathcal{M})
\]

where \(\mathcal{M}\) is the model, \(x\) is the input and an acquisition function \(a(x, \mathcal{M})\) that the system is using to identify where to query next. Given the classification problem setting and the use of Bayesian CNN models, we focus on the following acquisition functions:

1. Select from the sample pool data that maximizes the predictive entropy, Equation 2.10,
Max Entropy approach:

\[ x_H^* = \arg\max_{x \in \mathcal{D}_{pool}} \mathcal{H}_p(y \mid x) \] (2.16)

2. Maximize the epistemic uncertainty by selecting the samples from the pool with the highest epistemic uncertainty, Equation 2.14:

\[ x_{epistemic}^* = \arg\max_{x \in \mathcal{D}_{pool}} \frac{1}{T} \sum_{t=1}^{T} (\text{diag}(\hat{p}_{c_t}) - \bar{p}_c)^2 \] (2.17)

3. Maximize the mutual information between the predictions and the model posterior via Bayesian Active Learning by Disagreement (BALD) [11], [29]:

\[ x_B^* = \arg\max_{x \in \mathcal{D}_{pool}} \mathcal{H}_p(y \mid x, \mathcal{D}_{train}) - \mathbb{E}_{p(\omega \mid \mathcal{D}_{train})} [\mathcal{H}(y \mid x, \omega)] \]

\[ \approx -\sum_c \left( \frac{1}{T} \sum_{t=1}^{T} \hat{p}_{c_t} \right) \log \left( \frac{1}{T} \sum_{t=1}^{T} \hat{p}_{c_t} \right) \]

\[ + \frac{1}{T} \sum_{t=1}^{T} \hat{p}_{c_t} \log \hat{p}_{c_t} \] (2.18)

where the goal is to maximize the decrease in the expected posterior entropy.

4. Query by random sampling as a baseline [10], [11], [30]:

\[ x_{RS}^* = \arg\max_{x \in \mathcal{D}_{pool}} \text{unif}() \] (2.19)

where the function \text{unif}() is a uniform distribution in interval [0, 1]. This is equivalent to sampling uniformly a point at random from the pool.
2.4 Previous Work in Active Bayesian Deep Learning

There has been little work done using active learning in deep learning and neural networks; this is because, as used in an active learning environment, deep learning presents several challenges. Active learning methods depend on the ability to learn and upgrade models from limited quantities of data, deep learning is known for its reliance on massive volumes of data, and active learning acquisition functions require uncertainty from the model [11].

Gal et al. [11] demonstrated active learning with an MC dropout model for image data classification using both the MNIST dataset and skin cancer diagnostic images. Their work looked at four acquisition functions: BALD, variation ratios, standard deviation, and random. The results can be seen in Figure 2.4. These results achieved an error rate similar to other published semi-supervised learning techniques that use a limited amount of labeled data. Additionally, they compared the results of the Bayesian model to a deterministic neural network and showed that the additional uncertainty provided resulted in better performance in an active learning scenario. The authors reset the model and trained it to convergence with each active learning loop and left the question of the possibility of not resetting the model for future work.

Figure 2.4. MNIST test accuracy with different acquisition functions as a function of number of acquired images. Source: [11].
Dayoub et al. [31] examined different strategies of updating a neural network during active learning, including resetting or continuing to train. They found that incremental updates to the network and a final training session after active learning resulted in the best performance using the CIFAR-10 dataset. The results also showed that they could achieve similar or better results with a smaller portion of the data selected by active learning than using the entire dataset.

Concerning previous work regarding active learning with underwater acoustic data, Kriminger et al. [28] presented an online active learning strategy for automatically labeling novel items that were not in the training set. This is performed by sending the most uncertain samples to an outlier bin, from which new classification models can be constructed. The approach does not use neural networks, but the detection of novel classes by using the uncertainty from BDL is a task this work examines.

Bayesian Active Learning, also called BaaL, [32] is an active learning library for Python. BaaL can perform active learning with scikit-learn and PyTorch. It implements MC dropout neural networks along with several acquisition functions. A custom framework was implemented for this work due to the deterministic models, and an existing data pipeline for the passive sonar data was already built for TensorFlow.
CHAPTER 3: Methodology

This chapter is adapted from [1], previously published by the IEEE Journal of Oceanic Engineering, ©2021 IEEE

This chapter discusses the passive sonar dataset, CNN models used in this thesis, and the research methodology. Additionally, BDL and active learning implementation details are discussed.

3.1 Datasets
The deep-sea passive acoustic recordings were collected in the Monterey Bay from May thru August 2019 via a vector sensor, GeoSpectrum M20-105 by GeoSpectrum Technologies Inc. GeoSpectrum describes the M20-105 as “a hydrophone designed to measure 3-D particle velocity and acoustic pressure in a broad range of environments — from within 1 m of an air gun, to quiet ocean environments. The M20-105 includes a 3-dimensional orientation sensor, 3 dipole sensors, and one omnidirectional acoustic pressure sensor” [33]. The measurements contain a four-channel recording of the pressure scalar field and three directional particle velocity fields. Additionally, the sensor is wide-band capturing frequencies of up to 3kHz. This sensor is deployed on the Monterey Accelerated Research System (MARS) cabled observatory at the depth of 890 meters within Monterey Bay National Marine Sanctuary (MBNMS, California, United States) [34], [35]. The overall acquired dataset totaled 2257 hours. From now on, in this work, this dataset will be referred to as the MARS dataset.
In parallel to MARS data, we used AIS messages to label the dataset [5], [17]. We follow the approach of Beckler et al. [17], Santos-Domínguez et al. [36] and adopt a five-class ontology for ship sound classification, see Table 3.1. Having a no-ship class avoids utilizing a detector of ship presence followed by the ship-class classification algorithm. The MARS dataset contains multi-class and multi-labeled datasets; this work only examined the multi-class classification for BDL and active learning.

Table 3.1. Dataset Class List. Adapted from [5].

<table>
<thead>
<tr>
<th>Class</th>
<th>Ship Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Fishing Vessel, Tug, Towing Vessel</td>
</tr>
<tr>
<td>B</td>
<td>Pleasure Craft, Sailboat, Pilot</td>
</tr>
<tr>
<td>C</td>
<td>Passenger Ship, Cruise Ship</td>
</tr>
<tr>
<td>D</td>
<td>Tanker, Container Ship, Military Ship, Bulk Carrier</td>
</tr>
<tr>
<td>E</td>
<td>No ship present, background noise</td>
</tr>
</tbody>
</table>
Following the data labeling approach by Beckler et al. [17] for multi-class classification, the overall dataset was first standardized to 4 kHz sample rate and 30 second clips. These were then fused with AIS labels. Given the proximity of the harbor and the occasional lack of AIS signal from small boats and pleasure crafts, we recognize that some of the labels might be noisy. Overall, this approach yielded 1250 hours of labeled 30 second data clips. Each of the labeled 30-second clips contained all four channels of the sensor recordings, which enabled us to study both single-channel and four-channel model performance.

To generate data representations from 30-second labeled clips, we use low-level acoustic signal representations to generate mel-log spectrograms. Mel-log spectrograms are extensively used for deep learning audio-classification and are related to the Short Time Fourier Transform (STFT) magnitude [37]. They are constructed by applying a mel-filterbank over the magnitude of the complex output of the STFT. This effectively summarizes the frequency content with fewer dimensions [38]. Doing so reduces the input feature space to the input of the neural network, reducing the model capacity. One additional advantage of utilizing mel-filterbank over STFT magnitude is in the filterbank emphasis of details in the lower frequencies. Lower frequencies were proven to be important in underwater soundscape classification. On the other hand, the higher frequency content is deemphasized with the mel-filterbank, which generally is not needed for classification [17], [18].

Specifically, similarly to [5], [17], we compute STFT of 30-second labeled samples with a 500 ms frame size, 125ms frame hop, and a Hann window function. A 128 band mel-filterbank is further applied to STFT magnitude followed by log-compression of the signal [38] to produce mel-log spectrogram, see Figure 3.2.

![Figure 3.2. Mel-Log Spectrograms. The y axis is frequency and x axis is time. Sound intensity is represented by pixel color, with red being the most intense and blue being the least intense.](Image)
3.2 Deep Learning Models

This work uses a custom CNN model developed by Pfau [5] and the ResNet architecture to compare BDL using a vector sensor as input into a CNN. The custom CNN model was selected for use in the active learning experiments due to its better performance metrics and faster training time. For both the custom model and the ResNet model, MC dropout and variation inference models were created. For the MC dropout, additional dropout layers were added in between each convolution block of the custom model and the residual blocks of ResNet.

3.2.1 Custom Model

The custom CNN model uses a rectangular kernel, unlike image classification, which commonly uses a square kernel of size 3x3 or 5x5 [5]. Spectrograms, unlike images, have axes with different scales; the time axes are linear and frequency axes are logarithmic. This work uses the proposed 10x5 kernel size to apply the convolution operation on time vs. frequency mel-log spectrograms. Normalizing the input mel-log spectrograms is done with a batch normalization layer. Every layer of the network has a fixed kernel size. An increasing number of filters are being used throughout the network. Each subsequent set of filters adds 16 to the initial set of 16 filters. A max-pooling layer with a stride of 2 by 2 is used after each block of the two convolutional layers to cut the input size for the next block in half. Overfitting is prevented using L2-regularization on CNN layers and a dropout layer before the final output layer.
3.2.2 Residual Networks ResNet

In addition to the custom CNN model, the ResNet model was introduced by He et al. to avoid vanishing gradients and mitigate the problem, where increasing the depth of a model can result in increased training error [12]. To solve these problems, the authors introduced a concept called residual blocks that use short cut connections. These “shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers” [12]. The benefit of including residual blocks is, if any layer that hurts the architecture performance, it will be skipped by regularization. He et al. improved on ResNet with version 2 with a new residual unit that added an additional activation function to the identity mapping and showed improved performance on deep networks [13].

Figure 3.3. Custom CNN Model Architecture: Each block is described by number of filters, filter shape. Source: [5].
Choice of ResNet architecture was driven by a hyperparameter search over architectures, using single-channel data and deterministic configuration. Nine different models each were trained for 50 epochs and compared using validation accuracy to select the best model (see Table 3.2). By the end of the 50 epochs, the deterministic models were overfitting and additional training would not have improved the results. ResNet32v1 and ResNet44v1 performed similarly, but ResNet32v1 was selected due to it being a less complex model and faster training time.

Table 3.2. ResNet Size Search

<table>
<thead>
<tr>
<th>Model</th>
<th>Validation Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet8v1</td>
<td>.65</td>
</tr>
<tr>
<td>ResNet14v1</td>
<td>.70</td>
</tr>
<tr>
<td>ResNet20v1</td>
<td>.70</td>
</tr>
<tr>
<td>ResNet26v1</td>
<td>.71</td>
</tr>
<tr>
<td>ResNet32v1</td>
<td>.72</td>
</tr>
<tr>
<td>ResNet44v1</td>
<td>.72</td>
</tr>
<tr>
<td>ResNet11v2</td>
<td>.66</td>
</tr>
<tr>
<td>ResNet20v2</td>
<td>.67</td>
</tr>
<tr>
<td>ResNet29v2</td>
<td>.69</td>
</tr>
</tbody>
</table>
This work also tested ResNet models using the same 10x5 kernel size used in the custom model and found only a small increase in model performance.

### 3.3 Bayesian Deep Learning Implementation

For this work, BDL versions of the custom model and ResNet models were implemented using variational inference and MC dropout. The TensorFlow Probability (TFP) library includes Bayesian implementations of the standard TensorFlow neural network layers using both reparameterization and flipout techniques. We found that on the MARS data, the flipout layers performed better than the reparameterization implementation. The flipout layers employ the flipout estimator, which integrates across the kernel and bias in a MC approximation of the distribution. Flipout has nearly double the number of floating point operations as the reparameterization estimator, but it has a lower variance [24].

Converting the model to TFP requires replacing the default model layers with TFP layers. Implementing a KL divergence function that scales the loss once per epoch. In this work, the loss function categorical cross-entropy was used for multi-class classification. The Keras API will then automatically add the KL divergence contained on the TFP layers of the model to the cross-entropy loss, effectively calculating the ELBO. The below code shows the comparison of a deterministic model and TFP model using the sequential API:

```python
# deterministic model
conv1 = layers.Conv2D(NUM_FILTERS, kernel_size=KERNEL,
    strides=1, padding='same',
    use_bias=False, kernel_regularizer=regularizer,
    name='conv1_block_' + str(i))(x)
...

# TFP model

def kernel_divergence_fn(q, p, _):
    kernel_divergence = tfp.distributions.kl_divergence(q, p) /
    tf.cast(train_size, tf.float32)
    return w * kernel_divergence
...
```

25
MC dropout was implemented by using dropout layers that are always on in both training and inference. The PermaDropout layer was inserted between each convolution block of the custom model (see Figure 3.3) and between each residual block of the ResNet model. Below is the code from [39] that implements a custom dropout layer that is on during training and inference:

```python
from tensorflow.keras.layers import Lambda
from tensorflow.keras import backend as K

def PermaDropout(rate):
    return Lambda(lambda x: K.dropout(x, level=rate))
```

A common implementation of MC dropout is to set the model to training during inference. Caution should be used with this technique, as some other layers besides dropout have different behavior during training and testing such as batch normalization, and having the model set to training during inference could have unexpected results.

### 3.3.1 KL Annealing

KL annealing techniques can be used to improve the performance of the variational inference models proposed by Bowman et al. for use in variational autoencoders [40]. At training time, a variable weight is added to the KL term in the cost function. We set that weight to zero at the start of training so that the model can learn from the data. We gradually raise this weight as the training progresses, encouraging the model to smooth out its parameters and estimate the prior. This weight is increased until it reaches 1, at which point the weighted
cost function is equal to the ELBO. Figure 3.5 shows the loss graph when training the MARS dataset with KL annealing using a five epoch silent period, and annealing the KL weight from zero to one at 50 epochs.

![Figure 3.5. KL Annealing Loss Graph.]

We also evaluate a second strategy, Blundell et al. [9], where gain function follows per-batch dependence that is repeated in every epoch. That batch dependence is given with:

$$\alpha^i_{batch} = \frac{2^{M-i}}{2^M - 1}$$

(3.1)

where $M$ represents the number of batches in the epoch and index $i \in [1, M]$. This approach heavily weights the first few mini-batches by the complexity cost, while the latter mini-batches are more influenced by data. Similar strategies of per epoch annealing were previously proposed for controlling the learning parameter of the optimization schema [41]. Figure 3.6 shows the KL weight across 100 batches, with the KL weight starting high and quickly decreasing.
3.4 Performance Metrics

To compare the performance of one-channel and four-channel BDL models for passive sonar classification, we evaluate performance using accuracy (Acc), precision (Prec), recall (Rec) or true positive rate (TPR) and \( F^1 \) score [42]:

\[
Acc = \frac{TP + TF}{TP + TF + FP + FN}
\]  
(3.2)

\[
Prec = \frac{TP}{TP + FP}
\]  
(3.3)

\[
Rec/TPR = \frac{TP}{TP + FN}
\]  
(3.4)

\[
F^1 = 2 \cdot \frac{Prec \cdot Rec}{Prec + Rec}
\]  
(3.5)
where TP and TN are true-positives and true-negatives, while FP and FN are false-positives and false-negatives. The $F^1$ score can be thought of as a weighted average of precision and recall, with the best value being 1 and the worst being 0. We utilize accuracy and weighted $F^1$ scores for both evaluation of single-channel over four-channels using the full dataset and during the active learning test. Weighted $F^1$ was selected due to the unbalance between ship classes in the MARS dataset. To compare various BDL uncertainty measures, we investigate utilizing high uncertainty to filter data and calculate the weighted $F^1$ score against the retained samples. The scikit-learn library was used for all metric calculations [43].

### 3.5 Experiment Methodology

BDL and active learning experiments on the MARS datasets were conducted using Python 3.6, TensorFlow 2.2, and TensorFlow probability 0.11. Models were trained using an NVIDIA Quadro RTX 8000.

#### 3.5.1 Full Data

Models were trained using the full MARS dataset to use as a baseline for comparison using both one-channel and four-channels from the vector sensor. The custom model and ResNet model each had a traditional deterministic, MC dropout, and variational inference model trained. The dataset used in all of these studies was divided into three sets: training, testing, and validation. The training set comprised 80 percent of the total dataset, with testing and validation accounting for 10 percent each. Each model was trained for 500 epochs using the Adam optimizer and a learning rate of 0.0001 and the ReduceLROnPlateau Keras call back to lower the learning rate when needed. The deterministic and MC dropout models could have trained with significantly fewer epochs, but it was found that the models built using the TFP variational inference layers required more training time.

#### 3.5.2 Active Learning

The active learning experiments in this thesis implemented the below Bayesian Active Learning Algorithm:
Bayesian Active Learning (BAL) - General Algorithm

**Algorithm 1:** Bayesian Active Learning (BAL) - General Algorithm

**procedure** BAL($\mathcal{D}^e_{\text{train}}, \mathcal{D}^e_{\text{pool}}, N, AF, M$);

$\mathcal{D}^e_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{l}, \epsilon = 1 \rightarrow$ initial training set;

$\mathcal{D}^e_{\text{pool}} = \{(x_i, y_i)\}_{i=l+1}^{N}, \epsilon = 1 \rightarrow$ pool of candidates;

(pool set), $N \rightarrow$ number of samples to add at each iteration;

AF, acquisition function;

$M$, neural network model architecture;

**foreach** $t = 1, 2, \ldots$ **do**

$M = \text{train}(\mathcal{D}^e_{\text{train}})$;

**foreach** $j = 1, 2, \ldots, N$ **do**

select $x^* \in \mathcal{D}^e_{\text{pool}}$, the most uncertain instance according to the model $M$ and chosen AF;

add $\langle x^*, y^* \rangle$ to $\mathcal{D}^e_{\text{train}}$;

remove $\langle x^*, y^* \rangle$ from $\mathcal{D}^e_{\text{pool}}$;

**end foreach**

$\epsilon \rightarrow \epsilon + 1$;

**end foreach**

Our strategy for training was to use an initial training pool, 400 random samples from each class, train for 50 epochs, and use the best checkpoint as a starting place for testing various acquisition functions. In [11], the models were reset and trained to convergence during each active learning iteration. This work did not reset the model, but continued training from the best checkpoint obtained during the previous iteration. This was done because a possible future use of this work is active learning with deployed sensors, and it would not be feasible to reset the model during each active learning iteration. After each iteration, the weighted $F^1$ and accuracy were recorded on the testing dataset, and every 10 iterations the histogram plots of entropy and variance were saved to view how the uncertainty of the test data changed over time. In [11], 20 samples were selected per iteration, but this was found to be slow for the MARS data. Sixty-four samples worked well, and the batch size was set to match, so with each iteration, the number of batches increased by one.
For the initial active learning tests, each acquisition function was tested with 100 iterations for both single and four channels. Selecting sixty-four samples for 100 iterations resulted in the models only being trained on seven percent of the total training data. The best performing acquisition functions were selected to train on a larger portion of the data using both single-channel and four-channel to determine the best model that could be trained using minimal amount of data.
CHAPTER 4:
Results

This chapter is adapted from [1], previously published by the IEEE Journal of Oceanic Engineering, ©2021 IEEE.

This chapter presents the results of experiments using BDL and active learning utilizing all four channels of a vector sensor compared to single-channel information. Results are presented in two main sections. The first section uses the complete training data from the MARS dataset, examining performance metrics, and uncertainty information provided by Bayesian models. Next, active learning results are presented, comparing the acquisition functions using single and four-channel data with an active learning algorithm.

4.1 Full Dataset Results

First, we examined the performance of traditional deterministic non-Bayesian, MC dropout, and flipout versions of both our custom CNN and ResNet32v1 models in both single-channel and four-channel configurations using the entire training dataset, see Table 4.1 and Table 4.2:

Table 4.1. MARS Full Data Custom Model

<table>
<thead>
<tr>
<th>Model</th>
<th>Prec</th>
<th>Rec</th>
<th>$F^1$</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic 1 Ch</td>
<td>.80</td>
<td>.79</td>
<td>.79</td>
<td>.794</td>
</tr>
<tr>
<td>Deterministic 4 Ch</td>
<td>.81</td>
<td>.80</td>
<td>.80</td>
<td>.805</td>
</tr>
<tr>
<td>Dropout 1 Ch</td>
<td>.80</td>
<td>.80</td>
<td>.80</td>
<td>.796</td>
</tr>
<tr>
<td>Dropout 4 Ch</td>
<td>.82</td>
<td>.81</td>
<td>.81</td>
<td>.806</td>
</tr>
<tr>
<td>Flipout 1 Ch</td>
<td>.81</td>
<td>.80</td>
<td>.80</td>
<td>.798</td>
</tr>
<tr>
<td>Flipout 4 Ch</td>
<td>.82</td>
<td>.81</td>
<td>.81</td>
<td>.810</td>
</tr>
</tbody>
</table>
Custom model architecture outperforms ResNet32v1 model for all six model configurations in every metric. Overall, the models developed on all four channels of data achieved comparable performance to the single channel versions.

The difference between four-channel and single-channel performance is less noticeable with the custom model, see Table 4.1, and is within one percent in all metrics. This, however, is not the case for the ResNet style architecture where the difference is on average within three percent across the metrics and architectures.

In terms of $F^1$ and accuracy, the BDL models outperformed the deterministic models by a small margin. The best-performing model was the flipout model in both architectures. In addition to a slight improvement in performance metrics, BDL has the additional benefit of being able to capture additional uncertainty information compared to deterministic models.

### 4.1.1 Uncertainty Results

This next section examines the uncertainty estimates of the best-performing model, flipout custom CNN, to evaluate if the additional information provided by the four channels improves the uncertainty of the model. Uncertainty metric results are compared using the entropy, variance, epistemic, and aleatoric uncertainties of the test dataset. Displayed in Figure 4.1 is the mean of each metric.
The four-channel model resulted in decreased uncertainty across each metric tested. The results indicated that the four-channel data could provide predictions with reduced uncertainty. Due to the additional information provided by the four-channel vector sensor data, the more considerable decrease in epistemic compared to aleatoric uncertainty is an expected result since additional training data can reduce epistemic uncertainty.

The uncertainty estimates can help us understand more about the models. Figure 4.2 shows the histogram of the epistemic and aleatoric uncertainties calculated using Equation 2.14 for the custom CNN model. The plot shows that most of the uncertainty is due to aleatoric uncertainty resulting from sensor noise. It can be hypothesized that a reasonably good model was trained based on the available data. However, the limit of performance was reached due to not being able to overcome the sensor noise.
4.1.2 Uncertainty Filtering to Improve Reliability

When ML algorithms are part of mission-critical systems, the acceptable margin of error shrinks dramatically. In these situations, there must be confidence in the model’s predictions. In this section, we examine using the uncertainty provided by BDL to improve the reliability of the model by setting aside or filtering predictions with high uncertainty for review by humans before any automated decision making. For example, suppose an ML passive sonar system was deployed on a U.S. Navy ship. In the example case, the sonar operators could focus on the samples the model was uncertain about and have a higher degree of trust in predictions marked as low uncertainty.

Filtering was accomplished by selecting the most certain samples with an uncertainty metric and calculating the weighted $F^1$ score for the retained data. The process was repeated across the entire test set. Figure 4.3 shows the results for single-channel and four-channel configurations using the custom flipout model.
The results showed that total variance performed significantly worse than both entropy, and aleatoric combined with epistemic uncertainty. However, entropy, and aleatoric plus epistemic uncertainty performed similarly. Both single-channel and four-channel models show a significant improvement that can be achieved with uncertainty filtering.

Single-channel and four-channel comparisons of the results for entropy, and aleatoric plus epistemic uncertainty are presented in Figure 4.4. It demonstrates the advantage of utilizing all four channels when using uncertainty to filter predictions.
Unfortunately, this filtering requires saving some samples for a human expert or discarding them as unreliable depending on the application. Next, we consider a mission critical system that requires relatively high performance and compare the single and four-channel performance. Specifically, we will examine the percentage of data retained with results of .98 and .90 weighted $F^1$ scores using uncertainty filtering and the results are presented in Table 4.3.
Table 4.3. BDL Uncertainty Filtering Use Case

<table>
<thead>
<tr>
<th>Uncertainty Measure</th>
<th>$F^1 = 0.98$ Data Retained</th>
<th>$F^1 = 0.90$ Data Retained</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy 1 Ch</td>
<td>.62</td>
<td>.75</td>
</tr>
<tr>
<td>Entropy 4 Ch</td>
<td>.66</td>
<td>.79</td>
</tr>
<tr>
<td>Aleatoric + Epistemic 1 Ch</td>
<td>.63</td>
<td>.77</td>
</tr>
<tr>
<td>Aleatoric + Epistemic 4 Ch</td>
<td>.66</td>
<td>.79</td>
</tr>
</tbody>
</table>

The four-channel data would result in three to four percent less data having to be examined by a human expert, potentially saving time and money.

### 4.2 Case Study: Ship Track Predictions

Using a similar study with a single ship AIS track and the same code to generate plots as in [17] to compare one-channel and four-channel predictions, a ship was selected and the classification results made by both one-channel and four-channel versions of the model were examined. This case study used the custom MC dropout models and selected from the MARS dataset a pleasure craft (Class B) that transited near the MARS sensor on May 5, 2019, for about 7 hours. The AIS information was used to plot a range versus time graph and compare the differences between one-channel and four-channel predictions along a 50 km track, see Figures 4.5 and 4.6. This resulted in 852 30-second audio samples and with an accuracy of 0.9930 for one-channel and 0.9977 for four-channel. Uncertainty was calculated using normalized predictive entropy, Equation 2.10, and is displayed with gray transparent circles corresponding to the relative range of $H_p$. The incorrect classified predictions are shown to be coming from areas of higher uncertainty and the four-channel model displays a significant reduction in uncertainty values. Figures 4.7 and 4.8 are the ship’s AIS position and the predictions plotted on a map.
Figure 4.5. MARS Range Versus Time: One Channel
Figure 4.6. MARS Range Versus Time: Four Channel
Figure 4.7. Geographic AIS Location Plot: One Channel
When closer to the sensor, the predictions for both models became more certain, while incorrect classification instances appeared further away from the sensor.

### 4.3 Active Learning Results

For active learning results, we used the MC dropout model due to the smaller model size for improved training time and added simplicity of not having to optimize KL divergence and annealing as the training data size grows. The first active learning experiment tested four acquisition functions for 100 iterations using both single and four channels of the model. The results are shown in Figure 4.9.
All uncertainty-based acquisition functions performed better than the random baseline, indicating the algorithm selects the best sample to use for training. After 100 iterations, the active learning algorithm has trained on approximately seven percent of the total training data. Using the single-channel configuration, BALD performed the best with a weighted $F^1$ score of 0.6983. Epistemic uncertainty performed better in the four-channel model than the others, with a weighted $F^1$ score of 0.7142. However, all uncertainty-based acquisition functions performed similarly.

The following results examine active learning after an extended period of 400 iterations, focusing on comparing single and four channels. Figure 4.10 displays the weighted $F^1$ scores for BALD and epistemic uncertainty, comparing single-channel and four-channel results.
Figure 4.10. Active Learning: 400 Iterations. The value for the full data is from the four-channel MC dropout model.
These results demonstrate that at 400 iterations, or 23 percent of the training data, active learning could train the model with relatively close performance compared to the full training data. In all cases, four channels performed better.

<table>
<thead>
<tr>
<th>Acquisition Function</th>
<th>1 Channel</th>
<th>4 Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>BALD</td>
<td>0.771</td>
<td>0.795</td>
</tr>
<tr>
<td>Epistemic</td>
<td>0.760</td>
<td>0.789</td>
</tr>
<tr>
<td>Entropy</td>
<td>0.763</td>
<td>0.771</td>
</tr>
</tbody>
</table>

The active learning algorithm using the BALD acquisition function was able to train a model to .795 weighted $F^1$ score using 23 percent of the training data. This experiment did not require a human oracle to label data. However, the results demonstrates a benefit to active Bayesian deep learning for the DOD when there exists large datasets requiring labeling to build machine learning systems. Active learning can be used to select the best data for a human expert to label saving time and money.
CHAPTER 5:
Conclusion

This chapter is adapted from [1], previously published by the IEEE Journal of Oceanic Engineering, ©2021 IEEE.

The classification of underwater acoustic signals is a challenging problem that is difficult for human operators as well as machine learning algorithms. This work demonstrated the advantages of using a vector sensor over an omnidirectional hydrophone for surface ship classification. Typically, the benefit of using a vector sensor over a hydrophone in underwater acoustic applications is the ability to estimate the direction of arrival of sound waves. Classification results from our experiments showed minor improvements in $F^1$ scores when utilizing all four channels of the vector sensor relative to a single channel (comparable to a hydrophone) as input into neural networks. The addition of information from the four-channel signal, however, significantly reduces the amount of predictive uncertainty when using BDL techniques.

Averaged aleatoric uncertainty over the dataset is similar in value for both one-channel and four-channel evaluations, indicating the noise level on the sensor input was comparable in value. In contrast, the epistemic error shows a significant reduction between the two evaluations. This finding suggests the four-channel model is significantly improved over the one-channel version. This reduces the bias of the model relative to the true underlying distribution, given the reduction in epistemic uncertainty (of about 40%, see Figure 4.1). This contrast in epistemic uncertainty relative to little to no contrast in aleatoric uncertainty further reinforces the assumption that epistemic error indicates more information gain about the model performance on new data, which can be used in active learning over utilizing a bulk uncertainty metric like entropy.

The value of aleatoric and bulk uncertainty is in filtering out uncertain samples due to the noise in the input. This can help achieve higher model performance, demonstrating that we do produce calibrated uncertainties and higher sensor noise in specific samples. Understanding model uncertainty can assist the DOD with building trustworthy machine learning systems. The demonstrated uncertainty filtering can be combined with human and
AI teaming. That is, samples with high uncertainty are sent to a human expert, and the system is allowed to process the remaining data. When using the four-channel input, 3-4 percent fewer samples would require human intervention.

Previous work on Bayesian active learning used well known curated datasets, this is likely the first study on active learning with Bayesian models on passive sonar data. The active Bayesian deep learning experiments demonstrated that uncertainty-based acquisition functions had an increase in performance using four-channel input data over one-channel. While using a small percentage of the training data (e.g., 23% see Figure 4.10), active learning trained a model within two percent of the $F^1$ score compared to when using the entire training data. The experiments did not allow a better model to be trained with fewer data samples. Still, there are many applications in the realm of underwater sound that lack an automated labeling system. Some examples include submerged adversary submarines, cases when AIS was spoofed, and aquatic sea life. Using Bayesian active learning can save a significant amount of time and money often consumed when manually labeling data by allowing the active learning algorithm to select the data with the highest information gain to be labeled. This same strategy has potential in other deep learning applications in the DOD where there is a significant amount of unlabeled data.

This work proposes using the epistemic portion of Equation 2.14 as an uncertainty based acquisition function and demonstrates similar performance to existing acquisition functions.

### 5.1 Future Work

There is still more work to be done in studying model architectures and hyperparameters, including testing rectangular kernel sizes on other well-known image classification models for underwater sound classification improvements. Additionally, hyperparameter tuning would likely improve the model performance.

Another possible area of future work is understanding the ranges at which one-channel and four-channel models perform best. In areas of significant differences in performance, this understanding would allow dynamic selection of the best model based on range and bearing estimates from a regression model.
A significant area of future work is implementing live Bayesian active learning with passive sonar data. Specifically, monitoring a live stream from a deployed sensor and selecting the most uncertain samples to save for periodic training to improve the model continuously. Bayesian active learning could potentially allow models to be improved after deployment to production as the algorithm discovers new samples with additional information gain.
List of References


Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California