NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

GAUSSIAN PROCESSES FOR SATELLITE DATA

by

David W. Martin

September 2023

Thesis Advisor: Marko Orescanin
Second Reader: Scott Powell

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC, 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2023	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
GAUSSIAN PROCESSES FOR SATELLITE DATA

5. FUNDING NUMBERS

6. AUTHOR(S) David W. Martin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Using convolutional neural networks (CNNs) on satellite data sets outperforms previous meteorological methods for classifying weather events from space-based sensor data. However, CNNs alone lack the ability to quantify the uncertainty about their prediction given the input to the model. Gaussian processes (GPs) are a mathematical technique for making predictions while providing an estimate of the functional uncertainty about the underlying model; however, they are more computationally expensive to train than traditional CNNs. We train a ResNet50 CNN augmented with GPs against a large satellite dataset but test the model across not only held out data from the training dataset but also two other large datasets of a tropical cyclone and mesoscale convective system to test model performance across a variety of weather events. We compare the accuracy of a ResNet50 CNN augmented with GPs for the output layer of the neural network to the Goddard Profiling Algorithm, a differential equation-based approach that is not based on neural networks, a fully-deterministic neural network, as well as several Bayesian neural networks (BNNs) and find that Gaussian Processes outperform the GPROF and deterministic approaches but fall short of the top-end BNN results. Additionally, more work is needed to compare if the uncertainty estimates from the GPs are better calibrated than the uncertainty estimates from the BNNs.

14. SUBJECT TERMS machine learning, artificial intelligence, atmospheric science, Gaussian processes, GP, convolutional neural network, CNN, Bayesian neural network, BNN	**15. NUMBER OF PAGES** 69
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

GAUSSIAN PROCESSES FOR SATELLITE DATA

David W. Martin
Major, United States Marine Corps
BS, University of Texas, Austin, 2010

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2023

Approved by: Marko Orescanin
Advisor

Scott Powell
Second Reader

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Using convolutional neural networks (CNNs) on satellite data sets outperforms previous meteorological methods for classifying weather events from space-based sensor data. However, CNNs alone lack the ability to quantify the uncertainty about their prediction given the input to the model. Gaussian processes (GPs) are a mathematical technique for making predictions while providing an estimate of the functional uncertainty about the underlying model; however, they are more computationally expensive to train than traditional CNNs. We train a ResNet50 CNN augmented with GPs against a large satellite dataset but test the model across not only held out data from the training dataset but also two other large datasets of a tropical cyclone and mesoscale convective system to test model performance across a variety of weather events. We compare the accuracy of a ResNet50 CNN augmented with GPs for the output layer of the neural network to the Goddard Profiling Algorithm, a differential equation-based approach that is not based on neural networks, a fully-deterministic neural network, as well as several Bayesian neural networks (BNNs) and find that Gaussian Processes outperform the GPROF and deterministic approaches but fall short of the top-end BNN results. Additionally, more work is needed to compare if the uncertainty estimates from the GPs are better calibrated than the uncertainty estimates from the BNNs.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

ABI Advanced Baseline Imager

AI Artificial Intelligence

BDL Bayesian Deep Learning

BNN Bayesian Neural Network

CNN Convolutional Neural Networks

CPU Central Processing Unit

DCNN Deep Convolutional Neural Networks

DET Detection Error Tradeoff

DON Department of Navy

DPR Dual-Frequency Precipitation Radar

ELBO Evidence Lower Bound

FLOPS Floating-point Operations Per Second

GMI Global Precipitation Measurement (GPM) Microwave Imager

GP Gaussian Processes

GPM Global Precipitation Measurement

GPROF Goddard Profiling Algorithm

GPU Graphical Processing Units

IAS Intelligent Autonomous Systems

KL Kullback-Leibler

LOE Line of Effort

MC Monte Carlo

MCD Monte Carlo (MC) Dropout

MCS Mesoscale Convective System

ML Machine Learning

MNLL Mean Negative Log-Likelihood

NASA National Aeronautics and Space Administration

NN Neural Networks

NLL Negative Log-Likelihoods

PMW Passive-Microwave

RAM Random Access Memory

RBF Radial Basis Function

ReLU Rectified Linear Unit

ResNet Residual Network

RGB Red-Green-Blue

RMS Root Mean Square

RMSP Root Mean Square Propagation

ROC Receiver Operator Characteristic

SGD Stochastic Gradient Descent

Acknowledgments

I would like to thank my thesis advisor Dr. Marko Orescanin for his patience and mentorship throughout this process.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

In recent years breakthroughs in Graphical Processing Units (GPU) have made it possible to train substantially more capable Artificial Intelligence (AI)/Machine Learning (ML) models, leading to a growth of applications in remote sensing. Most of these models, however, lack the ability to quantify uncertainty which could be important for downstream applications. In this work we focus on understanding uncertainty aware deep learning models. Specifically, we focus on Gaussian Processes (GP) which can provide functional uncertainty information. We showcase this on a remote sensing problem of estimating convective versus stratiform precipitation, an important problem from meteorology.

The Advanced Baseline Imager (ABI) is a satellite that images the Earth across 16 separate spectral bands, both visible and non-visible. These passive images provide imagery and radiometric data on both ocean and land surface, meteorological, and atmospheric conditions across the entire planet [7].

The Global Precipitation Measurement (GPM) Microwave Imager (GMI) is an active scanning radiometer from National Aeronautics and Space Administration (NASA) that measures across 13 microwave channels to predict precipitation levels in the atmosphere around the planet [8]. It is a satellite-based sensor that covers most of the surface of the planet and takes around two weeks to fully sample [6]. The spatial resolution varies depending on the specific band recorded but ranges from 6km x 4km to 32km x 19km.

1.1 Research Objectives and Contributions

The objective of this study is to determine if augmenting deep neural networks with Gaussian Processes will allow for quantifying the functional uncertainty of the prediction of the model in a classification problem using meteorological satellites. Essentially, we are looking at two separate research tasks. The first is whether or not augmenting conventional Convolutional Neural Networks (CNN) with GPs is effective at categorizing precipitation as convective or stratiform using the satellite data we've collected. The second is benchmarking this technique against deterministic and Bayesian Neural Network (BNN) models for comparison.

Table 1.1. GMI Channels. Source: [6]

Band [GHz]	Polarization	Spatial Resolution (3-dB footprint size) [km x km]
10.65	V,H	32 x 19
18.7	V,H	18 x 11
23.8	V	16 x 10
36.5	V,H	15 x 9
89.0	V,H	7 x 4
165.5	V,H	6 x 4
183.31+/-3	V	6 x 4
183.31+/-7	V	6 x 4

Research questions being pursued in this work include:

1. How well do GPs classify precipitation type from satellite data?
2. How effective are GPs at providing an uncertainty estimate to output?
3. How do GPs compare to CNNs in training time and resources?
4. How do GPs compare to BNNs in training time and resources?
5. How does inference from GPs compare to traditional variational methods?

In order to answer these questions, this thesis will develop a deterministic CNN and a novel GP augmented CNN to compare it against. These results will then be compared against previous work on the same dataset [9]. The functional uncertainty found by the GPs will be compared with the aleatoric and epistemic uncertainty found by the previous work using BNNs [5].

1.2 Organization

The thesis begins with an overview of GP research, deep learning techniques, as well as BDL techniques and additional techniques for quantifying uncertainty, then continue on to how the approach in this thesis builds on those foundations. We will then cover the mathematics that these approaches are derived from, as well as the origin of the dataset used for training the models. The next section will cover the methodology used, the model architectures implemented and metrics and model benchmarks. After that follows an analysis of results, finished with research conclusions and potential avenues for future work.

1.3 Navy Relevance

With climate change taking the forefront in societal conversations about where to invest resources, the Department of Navy (DON) has made climate action a key priority in its plan looking towards 2030. Effective meteorological models are an important aspect of this plan, and artificial intelligence offers the potential for improving these models or creating new ways of interpreting observed data. Atmospheric precipitation is one of the ways in which climate effects can be measured, as energy absorbed into water sources causes that water to move into the atmosphere as precipitate. Measuring and categorizing this precipitation from satellite collection sources gives information about the amount of energy passing through the atmosphere to the surface. Additionally, meteorology is already a field which makes heavy use of computer and mathematical modeling techniques for predictions and forecasts, making it a good candidate for comparison to ML models.

AI, and specifically ML, are a collection of emerging technologies with recognized applications to DON priorities, in the context of weather models due to their abilities to automatically ingest large amounts of data to make predictions. One drawback of many current AI techniques however is their inability to provide effective estimates of the certainty of the model about its predictions, which can make it more difficult for human decision-makers to trust the output of the AI model.

A popular method of quantifying the uncertainty with the predictions in ML models is using Bayesian Deep Learning (BDL) techniques, such as Monte Carlo (MC) Dropout (MCD) and Flipout. This allows for a mathematical formulation of the uncertainty of the model, as well as breaking down the total uncertainty into both its aleatoric and epistemic sub-components [10].

GPs are one technique for quantifying the functional uncertainty of the prediction of a model, however they are not frequently used with real-world datasets due to the computational complexity of training them.

From a Navy perspective, the benefit of probabilistic models generally is that they are able to capture the uncertainty of the prediction, which enables a human decision-maker more context when using the prediction of the model as to how much trust to put into the output.

Aleatoric uncertainty is the uncertainty that is inherent in the process generating the data itself, better models and more data are unable to further reduce aleatoric uncertainty. In a military setting, this could be compared to the accuracy of a single shot from a rifle. We can predict the placement of the shot to within the accuracy of the firing weapon and the skill of the shooter themselves, but we are unable to determine the exact point that a bullet will impact beyond a certain tolerance. That tolerance is aleatoric uncertainty.

Epistemic uncertainty comes from our lack of knowledge about our model itself. This uncertainty can be reduced by either improving the complexity of the model itself, or observing more data with which to better fit the model architecture that we already have [11]. In a military setting, this is analogous to the fog of war, where we have a model of how we expect the conflict to play out, but lack of information causes uncertainty in our predictions. With more information this uncertainty is further reducible.

Functional uncertainty deals with the uncertainty in the underlying function that generates data. For a military perspective, as we begin using artificial intelligence to aid commanders in strategy versus human opponents, it is unlikely that we are going to be able to perfectly capture warfare into a system of equations. Functional uncertainty allows us to express the incompleteness of the understanding we have between the systems of equations we are able to generate and how effectively they are capturing the reality we are observing.

Deep Convolutional Neural Networks (DCNN)s are effective at automated classification tasks, where a complex input is sorted into one of several different categories. With satellite precipitation data, we use convective and stratiform categories as the two categories for comparison.

THIS PAGE INTENTIONALLY LEFT BLANK

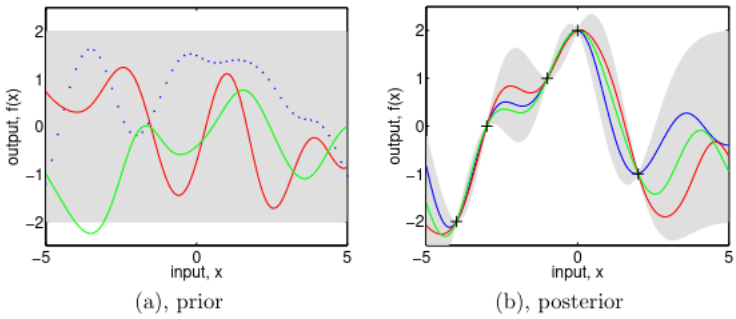
CHAPTER 2:
Background

Gaussian Processes are generalizations of Gaussian distributions. Whereas a distribution describes the properties of a random scalar, or in the multivariate case a random vector, a process describes the characteristics of random functions. A Gaussian Process provides us with a way to formalize our belief in the characteristics of a function by tying in with the Bayesian framework to allow us to define a prior in functional space, observe a finite number of data points, then update our beliefs to reflect our prior and this observed data.

We establish a prior belief about the functions that will describe the data we expect to see, and express that function in terms of a multivariate Gaussian process, with mean function $m(\mathbf{x})$ and covariance or kernel function $k(\mathbf{x}, \mathbf{x}')$. This expresses our view about the relative likelihood of each of the infinite functions that could describe the phenomena we are trying to model.

We then observe outputs of the unknown function at various points in the functional space, in machine learning terms these points are the training data, in Bayesian terms these are our evidence.

Figure 2.1. GPML. Source: [1]



(a), prior (b), posterior

Once we have observed our evidence, we are able to update our beliefs on the unobserved function to find our posterior. This posterior has an updated mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ that corresponds to our beliefs about the function in light of

the observed data. The benefits this provides is that the $m(\mathbf{x})$ will match the observed data points, and around these observed points we should see a relatively low covariance function $k(\mathbf{x}, \mathbf{x}')$, whereas farther away from observed data points the covariance function $k(\mathbf{x}, \mathbf{x}')$ expands to capture our greater uncertainty. This is both mathematically correct and matches our intuition as we would expect that our model confidence would improve as we are predicting near our observed data and lessen as we predict in areas we have not observed data in. Figure 2.1 graphically depicts this cycle. We can see the wide range of possible functions, depicted by the grey shading with some example functions depicted by the lines inside of the shaded area. Once points are chosen, there is substantially less uncertainty, however as we get further from the points the uncertainty grows.

The motivation for applying GPs as a strategy for satellite datasets is to compare it to more traditional forms of BNN, as well as against deterministic DCNNs. Most popular ML frameworks allow conversion of deterministic models to MCD, Flipout, or Local Reparameterization by only changing a few lines of code. Additionally, a large amount of effort has been put in to increasing the performance of these probabilistic augmentations, whereas GPs lack native support in most ML frameworks and are substantially less trivial to incorporate into models in popular ML frameworks. Comparing the GP approach to other forms of BNN shows us whether or not it may be a fruitful approach for future BNN applications.

2.1 Squared Exponential Kernel

We used the Squared Exponential Kernel, frequently referred to as the Radial Basis Function (RBF) or the Gaussian kernel. Sometimes referred to in the literature as the Exponentiated Quadratic as well, it is a common kernel for Gaussian Process models.

The Squared Exponential Kernel is defined as:

$$k_{\text{SE}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{(\mathbf{x} - \mathbf{x}')^2}{2\ell^2}\right) \tag{2.1}$$

where σ is the standard deviation and l is the lengthscale.

Figure 2.2. The Squared Exponential Kernel. Source: [2]

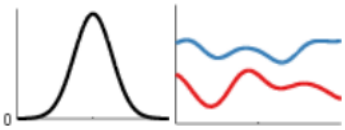


Figure 2.2 shows the visualization of the RBF on the left, and the shape of some example functions that can be described by the RBF kernel. Generally speaking, RBF kernels are universal and can be used to describe any potential function, the parameters of the RBF kernel describe how much the function's average distance varies from the mean and how far from the data to extrapolate from.

2.2 Existing Work

Using Gaussian Processes to augment Neural Networks (NN) on GMI data to categorize precipitation is a novel idea, however there is existing work in using deterministic NNs and using probabilistic BNNs to quantify the uncertainty around the predictions.

Deterministic models based on neural networks have been studied since the 1960s, although limited compute power made early implementations difficult to use for non-trivial problems. [12] Development continued on deterministic neural networks, with AlexNet [13] ushering in a new era of development on neural networks by showing the benefits gained by using deeper networks and GPUs to train them more quickly. CNNs have also been applied to satellite datasets in the past, and have been shown effective at classification tasks [14]. Additionally, CNNs have been used with meteorological data observed from satellites for classification, finding a high degree of accuracy relative to alternatives [15].

BNNs began being formally studied in the 1990s, with big advances corresponding to the use of GPUs to train the probabilistic equivalents to deterministic models. Bayesian Deep Learning, or BDL, is using these approaches with larger neural networks, which require a greater amount of time and resources to train.

BNNs typically follow one of two different approaches, either using variational inference, where a closed form approximation of the posterior function is used for inference. The

variational approximation is learned during the training process, but typically the variational inference is able to be executed more quickly due to closed form solutions being present.

The second approach is MC approximation, where the actual posterior function is learned using numerical approximation. During inference, several runs are made then averaged together, which allows for more accuracy if more computed is able to be used.

The trade-off between the two techniques is between increased numerical accuracy and speed of inference, where a variational technique is going to have substantially faster inference but be a less accurate answer, and an MC technique is going to be able to be tuned to be arbitrarily accurate but increased accuracy is going to come at substantial computational cost.

Work on Gaussian Processes for regression was popular in the 1990s and 2000s, however the computational difficulty relative to neural networks has made them less popular as GPU-trained neural networks have gained in popularity.

While both BNNs and GPs are more computationally expensive than deterministic NNs, BNNs are able to be parallelized on GPU hardware more readily than GPs, leading to better training performance.

Metrics for comparing probabilistic machine learning techniques differ from those used in deterministic machine learning. Typical metrics for deterministic machine learning techniques are accuracy or Receiver Operator Characteristic (ROC) curves, whereas with probabilistic techniques comparison of Negative Log-Likelihoods (NLL) are more expressive considering the problem domain.

Using deterministic and probabilistic machine learning techniques on GMI data has been used before, specifically using deterministic ResNet50s and probabilistic equivalents using MC Dropout, Flipout, and Local Reparameterization.

Deep learning models were able to outperform the Goddard Profiling Algorithm (GPROF) approach, which was previously the state of the art in meteorology for this type of task. The GPROF approach followed a traditional Bayesian approach, approximating a best estimate of the amount of rainfall given a set of observations using a discrete approximation of the continuous function.

2.3 Neural Network

Artificial neural networks are mathematical systems that are inspired by biological neurons and are capable of approximating complex non-linear functions. They are capable of both regression and classification problems, with classification problems requiring the use of a softmax function on the output to convert the class predictions into probabilities.

Neural Network:

$$f(x, \theta) = \sigma(Wx + b) \tag{2.2}$$

where W is the weight matrix and b is the bias.

The networks are trained using backpropogation, a technique where the network is initialized with random weights, a prediction is made during a forward pass, and the results of that prediction are compared to the actual output for the given input [16]. The gradient of the loss function with respect to the input data is then taken, and the weights are updated to get incrementally closer to the expected value.

2.4 Softmax Activation Function

The softmax function is a generalization of the logistic function into multiple dimensions and normalizes the output of a network into a probability distribution over predicted output classes [17].

Logistic function:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

Softmax function:

$$y_i = \frac{e_i^x}{\sum_{j=1}^{c} e_j^x} \tag{2.4}$$

The neural network provides weights corresponding to each category on its output layer, however these weights are not normalized into probabilities. Applying a softmax activation function over this output vector allows us to consider the outputs of the neural network as a probability distribution. For a multiclass classification problem, this allows us to more easily interpret the output of a neural network, as otherwise we would have an unscaled

output vector. Additionally, converting to probabilities preserves the relative order of the output classes, so applying an arg min or an arg max function still returns the same result. Effectively, this lets us convert a model that outputs a range of values and convert those values directly into a classification.

2.5 He Initialization

To avoid the problems of vanishing or exploding gradients, we initialize the weights of the neural network using He initialization [18]. He initialization ensures that we have the same variance of activations across all layers, and we achieve it by sampling our weights from a zero-centered Gaussian with standard deviation as follows:

He Initialization:

$$w_l \sim \mathcal{N}(0, 2/n_l) \tag{2.5}$$

where n_l is the width of the layer l. He et al. showed that using this form of initialization caused faster convergence of the model during training.

2.6 Bayesian Neural Networks

BNNs are a composition of a probabilistic model and a neural network that allow for combining the benefits of stochastic modeling with neural networks. Deterministic neural networks are unable to express uncertainty about their predictions, which is an even larger problem when they are used for inference on data outside of their initial training dataset [19]. Neural networks are able to approximate complex functions, and the stochastic models are able to capture the uncertainty within the approximation. Using Bayesian techniques allows for quantifying the posterior distribution given a prior distribution, neural network architecture, and observed data. Specifically, neural networks find a maximum likelihood estimate for θ, the unobserved parameters that maximize the probability of the observed data. Using BNNs allows us to instead learn the distribution of θ according to our prior beliefs and the observed data.

Mathematically, with a BNN we are attempting to learn:

$$p(y|D,x) = \int p(y|w,x)p(w|D)dw \tag{2.6}$$

where $p(y|D,x)$ is the probability of the class label y given an input x and dataset D.

2.7 Bayes Rule

Bayes Rule provides a method for breaking down the conditional probability into its constituent parts, allowing for solving the individual terms with respect to the others analytically [20].

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)} \tag{2.7}$$

Overall, the different techniques for Bayesian Neural Networks can be thought of as a tradeoff between mathematical accuracy and computational tractability. That is to say, given infinite time, most of the techniques will converge to approximately the same answer. However, how much computation is required to get them to that answer can vary widely between the techniques, and for complex problems with large datasets the realities of the differing techniques must be traded off with the available time and hardware.

2.8 MC Dropout

One popular technique to approximate distributions for θ is Monte Carlo Dropout, or MC Dropout. MC Dropout samples from a Bernoulli process parameterized with p_{drop} to turn the weights of the model on or off, similar to the dropout regularization technique [21], but also samples from the Bernoulli process during inference to provide multiple inferences to sample from and average. MC Dropout is computationally less expensive than other techniques while still providing an estimate of uncertainty [22].

Mathematically, the Bayesian form of the posterior distribution from a neural network is given as:

$$p(\theta|x, Y) = \int p(\theta|x, w) \cdot p(w|Y)dw \tag{2.8}$$

Which we approximate using the predictive distribution:

$$p(\theta|x, Y) = \sum_i p(\theta|x, w_i) \cdot p(w_i|Y) \tag{2.9}$$

When using MC Dropout, we continue to use the dropout layers during testing and average the resulting predictions, giving us a Monte Carlo predictive distribution:

$$p(\theta|x, Y) = \frac{1}{T} \sum_{t=1}^{T} p(\theta|x, w_t) \tag{2.10}$$

where w are the weights, x, Y are the observed datapoints, and T is the number of Monte Carlo inference runs we are performing.

2.9 Local Reparameterization

Another method of estimating the uncertainty of a neural network prediction is using local reparameterization. Local reparameterization transfers the sampling of noise from the intermediate steps to the end of the network, sampling the noise from the posterior approximation [23]. This reduces the number of samples that need to occur and additionally is easier to perform in parallel, making it even more efficient on a GPU. It also reduces the variance of the estimator.

2.10 Flipout

Flipout is a technique that reintroduces some of the variance from local reparameterization and takes a comparable amount of computation [24]. While both flipout and local reparameterization will converge to the same result over infinite samples, flipout typically is able

to converge over a shorter time period than reparameterization. Flipout takes advantage of the fact that Gaussian distributions are symmetrical, and converts the sampling of the noise from:

$$\theta_i = \mu_i + \sigma_i \epsilon_i, \ \epsilon_i \sim \mathcal{N}(0, 1) \tag{2.11}$$

Taking advantage of:

$$p(\epsilon_i) = p(-\epsilon_i), \text{ as Gaussians are symmetrical} \tag{2.12}$$

To instead sample from:

$$\theta_i = \mu_i + \sigma_i \epsilon_i r_i, \ \epsilon_i \sim \mathcal{N}(0, 1), \ r_i \in \{-1, 1\} \tag{2.13}$$

where μ_i is the mean, σ_i is the variance, r_i flips the sign of the distribution between positive and negative, and ϵ_i is Gaussian distributed random noise.

2.11 Variational Bayesian Methods

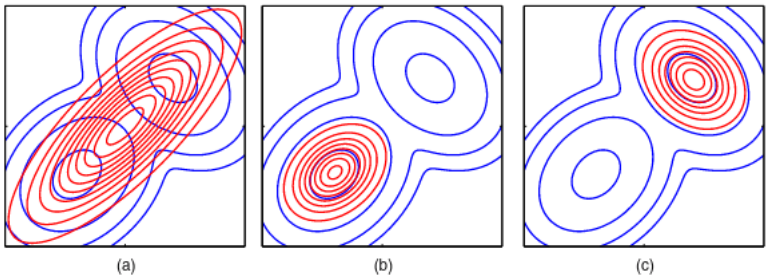
2.11.1 Kullback-Leibler Divergence

Kullback-Leibler (KL) Divergence is a statistical distance that measures how difference a probability distribution p_1 is from a separate probability distribution p_2 [25]. It is a distance, but not a metric, as $KL[p_1|p_2] \neq KL[p_2|p_1]$. For discrete probability distributions, such as for a classification problem, KL divergence is defined as:

$$KL[p_1|p_2] = \sum_{x \in \mathcal{X}} p(x) \log \left(\frac{p_1(x)}{p_2(x)} \right) \tag{2.14}$$

Figure 2.3 shows the intuition behind the KL divergence. We have a bimodal distribution denoted by the blue contour lines and we are attempting to fit a single Gaussian distribution to this bimodal distribution as closely as possible by minimizing the KL divergence. Figure 2.3 (a) shows the single Gaussian that maximizes the best fit of minimizing $KL[p_1|p_2]$, whereas (b) and (c) show the two local minima that are found by minimizing $KL[p_2|p_1]$, demonstrating that the KL divergence is not a metric as different distributions are found

Figure 2.3. KL Divergence. Source: [3]



depending on how the distance is computed.

2.11.2 Evidence Lower Bound

Evidence Lower Bound (ELBO) is a lower bound on the log-likelihood of observed data [26]. That is, given the "evidence" $p_\theta(X)$, it follows the inequality:

$$\ln p_\theta(x) \geq \mathbb{E}_{z\sim q_\phi}\left[\ln \frac{p_\theta(x,z)}{q_\phi(z)}\right] \tag{2.15}$$

where the left-hand side is the log of the evidence (the name comes from the classic Bayes formula) and the right-hand side is the lower-bound, or ELBO.

Overall, the different methods of applying probabilistic models to neural networks were used on a popular neural network architecture known as the Residual Network (ResNet).

2.11.3 Laplace Approximation

The Laplace approximation relies on doing a second order Taylor expansion of $\log p(\mathbf{f}|X,\mathbf{y})$ around the maximum of the posterior [27], giving the Gaussian approximation:

$$q(\mathbf{f}|X,\mathbf{y}) = \mathcal{N}(\mathbf{f}|\hat{\mathbf{f}}, A^{-1}) \propto \exp\left(-\frac{1}{2}(\mathbf{f}-\hat{\mathbf{f}})^T A(\mathbf{f}-\hat{\mathbf{f}})\right) \tag{2.16}$$

Effectively, the Laplace approximation is finding the maximum of the posterior, using a partial Taylor series expansion to extract the local curvature information from around the

maximum, and using that maximum and curvature to define a Gaussian distribution fitted to that point.

Figure 2.4. Laplace Approximation. Source: [3]

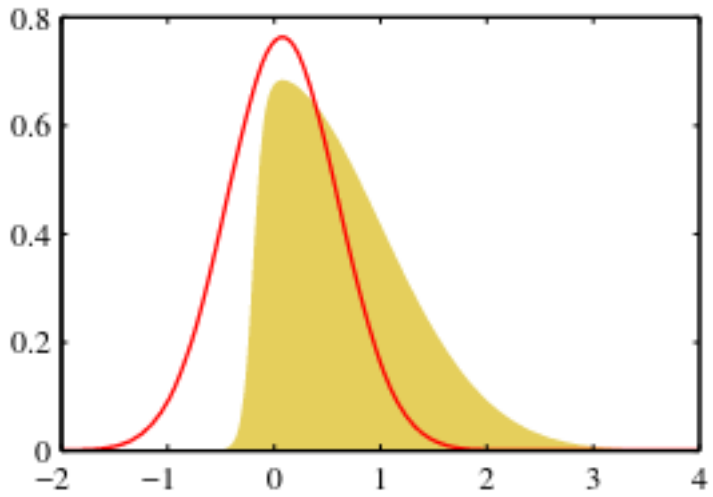


Figure 2.4 shows an example Laplace approximation, where the yellow shaded area is the posterior distribution and the red Gaussian distribution is derived by using the Laplace approximation.

2.12 ResNet Architecture

Deeper neural networks allow for learning more complicated non-linear functions than shallower neural networks, however the trade-off comes in the additional time required to train them. Additionally, deeper networks suffer from the "vanishing gradient" problem, where for sufficiently deep networks the gradient trends to zero, causing learning to stop and complicating back-propagation [28]. One solution to this problem is using residual networks, where each layer is learning residual functions with reference to the layer inputs instead of learning unreferenced functions [4]. This allows for the network to be deeper than would be possible with the same computational resources and provides a higher level of accuracy for classification and regression tasks.

2.13 Aleatoric Uncertainty

Aleatoric uncertainty is the uncertainty that is inherent in the stochastic process that is generating the data [29]. That is, if we were to run an experiment multiple times, we would be unable to further reduce the aleatoric uncertainty even if we had more data. An example of this would be flipping a fair coin. Prior to conducting the first flip, the best prediction we could make would be that both heads and tails were equally likely to occur from a given flip. Even with more information, in this case flipping the coin multiple times and observing the results, we would still be unable to predict a future coin flip with more accuracy than that heads and tails are both equally likely to occur.

2.14 Epistemic Uncertainty

Epistemic uncertainty is uncertainty which is reducible with more information [29]. Following the example in the preceding section, if we were instead using a biased coin but were unsure of the bias between heads and tails, conducting further flips would allow us to better quantify the bias between the two sides and would reduce the epistemic uncertainty. Once we had determined the bias between the two sides, we would have eliminated the epistemic uncertainty and only be left with aleatoric uncertainty.

Aleatoric and epistemic uncertainty are able to be broken out of total predictive uncertainty via the following equation [30]:

$$\frac{1}{T}\sum_{t=1}^{T} diag(\hat{p}_t) - \hat{p}_t^{\otimes 2} + \frac{1}{T}\sum_{t=1}^{T}(\hat{p}_t - \bar{p}_t)^{\otimes 2} \tag{2.17}$$

where the first term represents aleatoric uncertainty, and the second term represents the epistemic uncertainty.

2.15 Functional Uncertainty

Functional uncertainty moves the uncertainty into function-space, where instead of dealing a probability distribution over scalars or vectors, we are dealing with a stochastic process, or a generalization of a distribution over functions [1]. Colloquially, we can think of functional uncertainty as an expression of our uncertainty about the underlying mathematical function

that is generating the data that we are observing. Whereas with aleatoric and epistemic uncertainty we are assuming that we have a correct model and our uncertainty is just a function of the underlying data or the hyperparameters of our model, with functional uncertainty we are not assuming that the underlying function is correct, but only more generally assuming a general "shape" of the function. Again extending the example in the previous section, we could model a sporting event between two teams using the same concept as before, looking at the wins and losses of previous meetings between those teams. However, while the outcome of a single coin-flip is a Bernoulli process, and a Beta distribution is the expression of uncertainty about the parameter p of a Bernoulli process, that does not accurately represent the mathematical process that describes the winner of a sports match. The distance between $\beta(\alpha, \beta)$, and the true function $f(x, \theta)$, is the functional uncertainty.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3: Methodology

This chapter discusses the GMI dataset which was used as the baseline for the different models, as well as the preprocessing that occurred to prepare the dataset for training. The ResNet architecture is also covered, as well as how the Gaussian Processes are integrated into the larger model for the Bayesian training.

3.1 Dataset

The dataset was observed by the Global Precipitation Mission using Dual-Frequency Precipitation Radar (DPR) and Passive-Microwave (PMW) instruments from an international mission headed by NASA.

> The Global Precipitation Measurement (GPM) Microwave Imager (GMI) instrument is a multi-channel, conical- scanning, microwave radiometer serving an essential role in the near-global-coverage and frequent-revisit-time requirements of GPM. The instrumentation enables the Core spacecraft to serve as both a precipitation standard and as a radiometric standard for the other GPM constellation members.
>
> The GMI is characterized by thirteen microwave channels ranging in frequency from 10 GHz to 183 GHz. In addition to carrying channels similar to those on the Tropical Rainfall Measuring Mission (TRMM) Microwave Imager (TMI), the GMI carries four high frequency, millimeter-wave, channels near 166 GHz and 183 GHz.
>
> With a 1.2 m diameter antenna, the GMI provides significantly improved spatial resolution over TMI. [8]

The GPM mission also carries an advanced DPR [31] system, which collect multi-spectral measurements across the entire atmospheric column and build links between PMW brightness temperatures and radar-derived precipitation rates. These links are then assessed to estimate surface precipitation across targeted swaths throughout the planet.

This approach provides accurate global estimates of precipitation over long periods of time, however region-specific biases in precipitation rate arise due to differences between convective and stratiform rainfall. Convective precipitation has heavier rainfall and more vertical motion than stratiform precipitation, and this vertical structure and horizontal radar reflectivity can be used by space-borne radar sensors to classify the precipitation by the vertical columns of DPR echo [32].

3.2 Data Collection

The dataset was the same used in Orescanin et al. [9], the methodology is reprinted below:

> The performance of deep learning methods, mainly accuracy and ability to generalize to new inputs, is driven by the quality and quantity of the dataset. The approach of Petković et al. is followed with some adjustments. First, two independent 12-month periods of GMI and DPR co-located observations were created. Model training and validation relied on data collected during 2017, while performance tests of the trained model were performed on 2018 observations. The GMI brightness temperatures (Tbs) observed at 13 microwave channels (10.65 H/V, 18.7 H/V, 23.8 V, 36.5 H/V, 89.0 H/V, 166 V/H, and 183.3 ± 3/7 V GHz) stored in publicly available (e.g., https://storm.pps.eosdis.nasa.gov) GPM level-1 standard product [GPM_BASEGPMGMI_XCAL - V05; GPM Science Team 2016] were used to construct model training features. The current study chooses the GMI product to define the atmospheric state (i.e., an observation vector) over an area of approximately 125 km × 125 km centered on the observing field of view (FOV, hereinafter referred to as pixel). Given the GMI's scanning geometry, such an area corresponds to a patch of 25×9 individual pixels. Collecting Tbs at each of 13 GMI channels, the resulting training feature elements are stored into 9×25×13 arrays, where the three dimensions reflect the number of GMI scans, pixels, and channels, respectively, to form an input dataset for the model. The input dataset was normalized by scaling each Tbs with its channel's maximum value, subtracting the channel's mean and dividing by the channel's standard deviation, a procedure known as a z-score scaling. A segment of the dataset, covering a GPM orbit over the Atlantic Ocean on

August 11, 2018, was used for a case study to demonstrate performance over a continuous spatial region and demonstrate concepts around using uncertainty. This specific part of the dataset was not used in the training nor testing phases of the presented results.

The data were labeled using the output from the DPR radar, specifically the GPM_2ADPR standard product and its precipitation rate and type flag. Two precipitation categories, convective and stratiform, were considered when defining the label for each individual GMI pixel. Using DPR observations falling within GMI's 18-GHz channel FOV, a convective fraction is calculated by applying Gaussian weighting to DPR-observed precipitation rates. This has ensured accurate matching between DPR- and GMI-viewing geometry. Once available, the convective fraction of precipitation was used to assign a label to each individual GMI pixel. A convective flag was assigned to all pixels with the fraction of 50% or more; otherwise, the pixel was labeled as stratiform. Observations containing any missing or nonclassified data (less than 5% of total data) were excluded from the training dataset to ensure minimal noise. Upon labeling, the dataset used for model development was balanced so that an equal representation of both precipitation classes is preserved. All ~14 million samples were further split into training/validation/test data with an 80/10/10 ratio, respectively.

3.3 ResNet Architecture

The baseline deterministic model for this project was a ResNet 50 architecture with 50 2-dimensional convolutional layers.

An adjustment needed to be made to the traditional ResNet 50 architecture, as the baseline ResNet was designed for 299 x 299 x 3 inputs to support the 3-channel RBG ImageNet input images.

The dataset was adjusted to use 9 x 25 pixels for each image, with 13-channels and preprocessing to provide pixel-level TBS (brightness temperatures) corresponding to 125 x 125 km spatial area. This required adjusting the input layer of the ResNet implementation to support 9 x 25 x 13 instead of 299 x 299 x 3 as well as adjust the average pooling from 8

x 8 in the traditional ResNet to 2 x 2 in the deterministic ResNet model to compensate for the new input dimensions.

The architecture was also extended by adding a Gaussian Process layer immediately prior to the output layer to learn the Gaussian Process in the feature space of the output. This allowed the CNN to reduce the input space to a much smaller dimensionality before applying the GP approximation, enabling the GP to be applied against a much less complex targets. This also allowed for the CNN to learn the more non-linear portions of the problem space.

The CNN was implemented using Tensorflow [33], a machine learning framework in Python that allows for high-level specification of the model architecture, as well as the training and inference pipeline.

Figure 3.1. Basic ResNet Architecture. Source: [4]

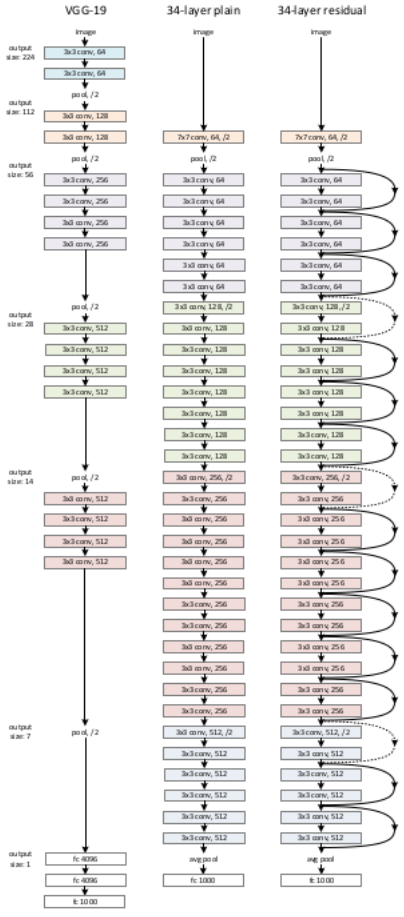


Figure 3.1 shows a comparison between the VGG-19 architecture and the 34-layer ResNet architecture both with and without residual connections. The residual connections, denoted by the arrows on the right-hand side of the figure, are skip connections where the output of one layer is allowed to skip the next layer, continually through the end of the network, which allows the output from earlier stages of the network to pass through to the end of the network unchanged.

Figure 3.2. Basic ResNet Block. Source: [4]

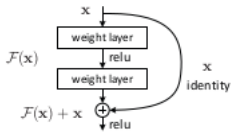


Figure 3.2 shows an individual building block of the ResNet, capturing that the block can either allow the input from the previous block to pass through an additional block, or bypass that block to proceed directly into the next block.

3.4 Gaussian Process Architecture

The GP layer was added at the end of the ResNet model, replacing the final Dense Layer with a Gaussian Process layer implemented in Edward2 [34]. Replacing the Dense Layer rather than extending it with an additional layer makes it a more fair comparison, as the probabilistic CNN is no deeper than the deterministic CNN, so any additional predictive ability comes from the GP layer rather than just additional depth of network.

3.5 Negative Log-Likelihood Loss

NLL is the function used to measure the distance between the predicted results and the actual results represented in the data. For a two-class classification problem, such as convective versus stratiform, binary cross entropy is the negative log-likelihood, computed as:

$$\text{NLL} = -(p * \log(q) + (1 - p) * \log(1 - q)) \tag{3.1}$$

where p and q represent the respective probability of each category as output by the model.

Our optimizer tries to minimize the NLL, as smaller values of the loss equate to more similarity between the expected results and the predicted results. Additionally, the loss function is asymmetric, where if the model makes a high confidence prediction that turns out to be wrong, that is penalized more highly than a low confidence prediction that turns out to be wrong, which is a desirable quality in a loss function. Also, the loss function is always greater than or equal to zero, negative values are not possible.

3.6 Stochastic Gradient Descent

Stochastic gradient descent is a variant of the Gradient Descent Algorithm that is more effective when used with large models and large amounts of training data [35]. In traditional gradient descent, the gradient of the loss function with respect to the entire training dataset is taken, then an average is taken and the model parameters are updated.

The equation for gradient descent is:

$$x_{n+1} = x_n - \eta \nabla f(x) \tag{3.2}$$

Where ν is the learning rate, a scaleless measure of how quickly we intend on updating based on the gradient. This also shows the danger of getting trapped in a local minima, as once $\nabla f(x)$ reaches zero x_{n+1} ceases to update.

Stochastic gradient descent instead computes the average gradient after processing randomly selected small batches of the training data. The benefits of this are that with sufficiently large datasets it becomes impossible to fit all the data into Random Access Memory (RAM) which causes a significant slowdown if the data needs to be shuttled back and forth between the RAM and hard drive. Gradient descent also has a tendency to become trapped in local minima, where the gradient becomes zero at a local minima that is not also the global minima, causing the algorithm to stop updating. Stochastic gradient descent introduces some noise into the updates, which allows it to more efficiently traverse the loss landscape in search of the global minima.

Mathematically, it is:

$$x_{n+1} = x_n - \eta \nabla f_i(x) \tag{3.3}$$

where $f_i(x)$ represents that we are updating on each individual subset of the data that we are computing the gradient with, rather than the entire set of data. This also shows how it is much harder to get trapped in a local minima, as even if $\nabla f_i(x)$ is zero, once the next step is updated a new $f_i(x)$ will be chosen randomly, making it extremely unlikely that each $\nabla f_i(x)$ is zero unless the true global minimum has been found.

3.7 Training and Validation Methodology

80% of the data was used for training the neural network with 10% being used to validate the neural network. This means that the training data was available for use by the optimizer during backpropogation, however the validation data was only used during forward passes to assess whether or not the neural network was learning generalizable skill or merely "memorizing" the training data. Every epoch that the validation loss function improved, the model weights were saved as a checkpoint for use during testing. Training was done on a single GPU, using the TensorFlow Stochastic Gradient Descent (SGD) optimizer with a Binary Cross-Entropy loss function. After 75 epochs of training, with each epoch consisting of the model being updated against the entire testing dataset, there was no further improvement in the validation loss. Conducting 75 epochs of training took just over 14 days of computing resources.

3.8 Testing Methodology

10% of the original dataset was held out for testing purposes. As the model trained, model checkpoints were saved every time the validation loss improved, allowing us to access the model weights independently of the code and data used to train them. A separate script was written that evaluated the cross-entropy loss and accuracy of the model weights against the held-out test dataset, ensuring that no test data could leak into the model training. This technique of separately evaluating the model against the test data also ensures that no "look-ahead" occurs, where the training is stopped prematurely or continued longer than necessary

do to understanding how the training is affecting the performance against the testing data. Additionally, two separate datasets were prepared that separated from the testing dataset both spatially and temporally. The first was hurricane off the coast of Hawaii, and the second was a mesoscale convective system. These separate testing datasets were used to ascertain if the model was generalizable to different kinds of weather than that used to train the model itself.

3.9 Benchmarks

To benchmark the model, we compared it with the deterministic ResNet50 model, as well as the previously trained BNN models using the same dataset. Accuracy and cross-entropy loss was compared to show the relative differences between the different Bayesian approaches, the deterministic approach, and the Gaussian Processes approach. Additionally, the uncertainty quantified by each of the models was compared to note if the uncertainty was higher in mis-categorizations than it was in correct categorizations.

3.10 Summary

To summarize, the same data was used to train and validate deterministic, Bayesian, and Gaussian Process augmented convolutional neural networks. The Gaussian Process augmented convolutional neural networks were trained using SGD, with Binary Cross-Entropy as the loss function. And finally, the deterministic, Bayesian, and Gaussian Process augmented CNNs were evaluated on the same test dataset that had not previously been exposed to any of the neural networks during training.

CHAPTER 4:
Results

This chapter introduces the results of the GP model, and provides a comparison between the GP model, the GPROF algorithm results, and previous BNN work that has been done in prior studies. The benchmarks consist of the test dataset, which is the held out data from the training and validation data and consists of 1.4 million feature vectors. Additionally, a dataset containing a portion of Hurricane Lane from the vicinity of Hawaii in August 2018 containing 1800 feature vectors is provided for comparison. And finally, a Mesoscale Convective System (MCS) dataset is used for additional testing data in a different type of precipitation system. Testing the model across this wide variety of weather systems and across different regions of the planet has the benefit of showing that the model is robust and able to generalize beyond just the weather it was shown during training.

The GP model was initially trained with both the mean and covariance matrix of the model to attempt and obtain the uncertainty from the model predictions for comparison to other BDL techniques. However, the GP covariance matrices required more RAM then was possible to obtain on the compute resources available, so the model was then trained with strictly the mean to evaluate whether or not using the GP was still an effective way to make predictions relative to traditional deterministic techniques. This answered one of the research questions on the ease of use for GP models, highlighting that these models are substantially more RAM intensive than deterministic models or more traditional BNN techniques, which illustrates why they have fallen out of favor in real-world applications. We attempted switching the training from GPU to Central Processing Unit (CPU) to have access to more RAM, however even with 512GB of CPU RAM we were unable to get a full epoch of training with the training dataset. As such, we are comparing the results

Table 4.1. GP Model Parameter Summary

Total parameters	2,463,650
Trainable parameters	1,665,794
Non-Trainable parameters	797,856

of the GP model with the deterministic model as well as the comparison of the overall accuracy of the BNN models, that is, the models without additional filtering based on the uncertainty. We do compare the GP model to the filtered models, however that is not a straightforward comparison as there was no uncertainty filtering from the GP model. Future work on comparing the uncertainty filtering from the GP models is recommended.

4.0.1 Testing Dataset

Table 4.2. Model Test Confusion Matrix

		True Category		
		Positive	Negative	Total
GP Model Prediction	Positive	6370	880	7250
	Negative	935	6315	7250
	Total	7305	7195	14,500

Table 4.3. GPROF Test Confusion Matrix

		True Category		
		Positive	Negative	Total
GPROF Prediction	Positive	7250	0	7250
	Negative	3547	3703	7250
	Total	10,797	3703	14,500

Confusion matrices are diagnostic tools that show the predictions of a given model, as well as what the accurate class for the prediction should be. In other words, a confusion matrix gives the true positives, false positives, true negatives, and false negatives of a given model in table form, making it easy to see not only if the model is correct in its predictions, but also if there is a trend in where a model makes mistakes.

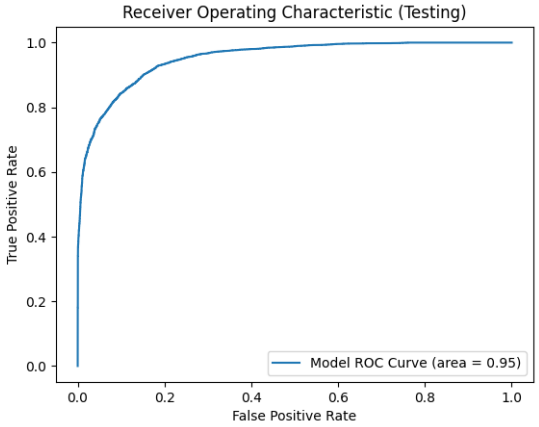
The model had significantly higher accuracy on the test dataset than the GPROF algorithm, with a model accuracy of 87.48% and a GPROF accuracy of 74.3%. The GP model also outperformed all of the BNNs prior to filtering for uncertainty, with the exception of the Flipout and Local Reparameterization networks that had the KL Divergence term zeroed out, as seen in Table 4.4. The GPROF algorithm did not have any false positives as shown in

Table 4.4. Comparison Between Different Models on Test Dataset. Adapted from [5]

GPROF	0.743
ResNet38 V2	0.868
Flipout, KL = 0	0.927
Reparam., KL = 0	0.920
Flipout, KL RW	0.866
Reparam., KL RW	0.864
MC Dropout	0.860
ResNet with GP	0.875

Table 4.3, however it had a significant number of false negatives which lowered the overall accuracy to below that of the GP model. This can be seen in the relatively larger number of values in the main diagonal of the GP model confusion matrices in Tables 4.3 and 4.2.

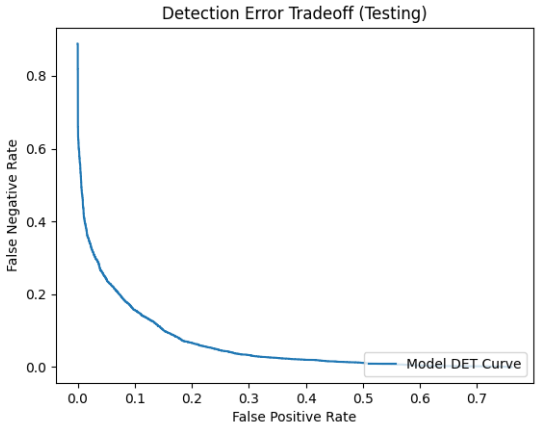
Figure 4.1. Receiver Operator Curve Comparison



The Receiver Operating Characteristic (ROC) Curve is a tool for comparing the true positive rate to the false positive rate as we vary the classification threshold. It is a graphical measure of the sensitivity of the binary classifier to where we set the decision threshold. A random classifier would have a ROC curve that would be a 45 degree line from the lower left to the upper right, and a perfect classifier would climb immediately to the top axis then be a straight line to the upper right. In general, better classifiers are more concave, with the best

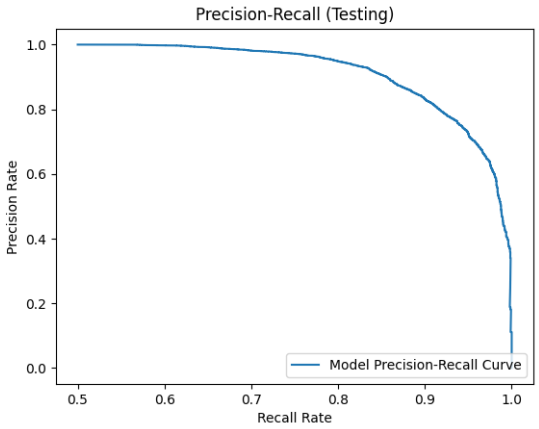
classifiers being bowed in towards that upper left corner. Across all three datasets, shown in Figures 4.1, 4.9, and 4.13, the GP model shows a high ROC area of 0.86 or higher, showing that the model is very effective at classifying convective versus stratiform precipitation. We were unable to compare this curve to the GPROF algorithm, as our GPROF model provides a classification, but no logits we can compare different classification thresholds against.

Figure 4.2. DET Curve Comparison



The Detection Error Tradeoff (DET) Curve is a diagnostic tool for comparing false rejection versus false acceptance. The ideal shape for these curves is concave, with curves that are closer in to the origin better than ones that lie closer to the 45 degree line sloping from the upper left of the graph to the lower right. Figures 4.2, 4.11, and 4.15 all also show that the GP model is an effective binary classifier, with curves that pass close to the origin.

Figure 4.3. Precision-Recall Curve (Test Dataset)



The Precision-Recall Curve is a comparison of how the precision and the recall of a model vary as we adjust the classification threshold on the underlying logits. Precision is defined as:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{4.1}$$

with recall defined as:

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{4.2}$$

Intuitively, we can think of precision as the likelihood of a model to not label a negative sample a positive. Recall is the opposite, the likelihood of a model to find all positive samples. These measures are helpful whenever we are dealing with a classifier on a dataset that is not perfectly balanced, that is, a dataset that has a higher proportion of one category than the other. A classifier that is working in a space where the underlying data is not perfectly balanced may end up favoring the more frequently observed category when in doubt, without actually developing skill as a classifier.

Plotting the precision-recall curve helps us verify that the classifier is not doing that. Figures 4.3, 4.10, and 4.14 all show high areas under the precision-recall curve, which indicates that the model is not simply favoring one category over the other.

Figure 4.4. Testing Data Global Map

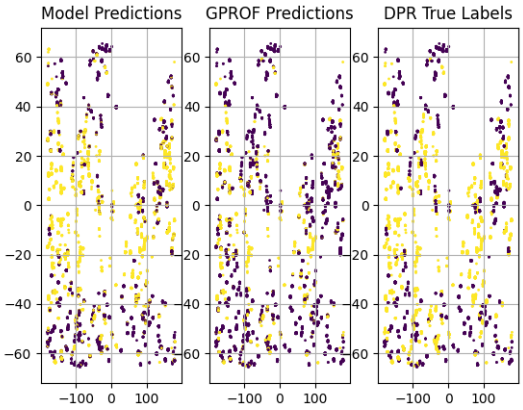


Figure 4.4 shows a visualization of the locations of points in the test dataset. The GP model predictions and GPROF model predictions are on the left and middle respectively, with the DPR derived true labels displayed on the rightmost plot.

The Testing Dataset is just the held out data from the same set of observations that built the training and testing dataset. This data is geographically dispersed, making it more difficult to see a clear pattern of classification, however even without the aid of the confusion matrix and accuracy data it is clear that the GP model is more consistently matching the true classification than the GPROF model.

4.0.2 Learning Curve Analysis

The learning curve in Figure 4.5 shows that there is not a significant difference between the training and validation accuracy, showing that we do not have evidence of overfitting. The GP model appears to have good regularization, as otherwise we would expect to see training accuracy continuing to climb while validation accuracy plateaued at a lower level.

Similarly, the loss values in Figure 4.6 confirm what we are seeing in the learning curves for accuracy. Training and validation loss both quickly drop then both plateau at approximately

Figure 4.5. Training vs. Validation Accuracy

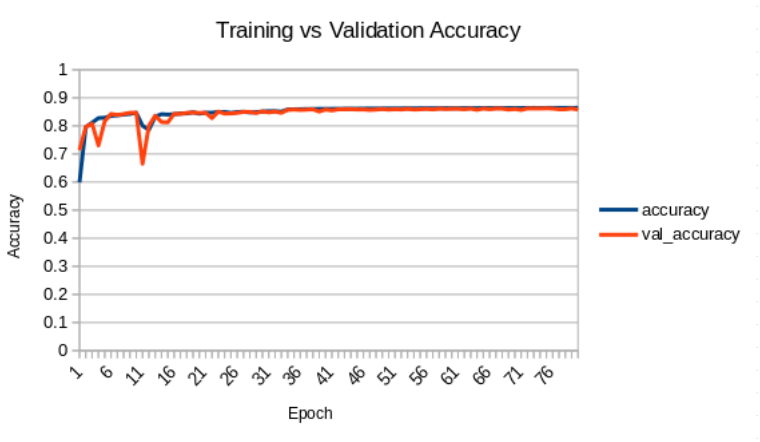


Figure 4.6. Training vs. Validation Loss

the same level, with training accuracy not continuing to drop to a lower level which would tend to suggest a risk of overfitting.

4.0.3 Hurricane Dataset Analysis

Interestingly, for the Hurricane Lane dataset the GP model and the GPROF predictions both had approximately the same number of false positives. Visually inspecting the classification map in Figure 4.7, it appears as though both models consistently misclassified pixels on the western edge of the dataset between 13 and 14 degree latitude and -143 and -144 degrees longitude, although the GP model still had a higher accuracy of 82.28% compared to a

Table 4.5. Comparison Between Models on Hurricane Dataset

GPROF	0.677
ResNet with GP	0.823

GPROF accuracy of 67.68%.

Figure 4.7. Plots of the Northeast Section of Hurricane Lane Using the ResNet50 with Gaussian Processes Model

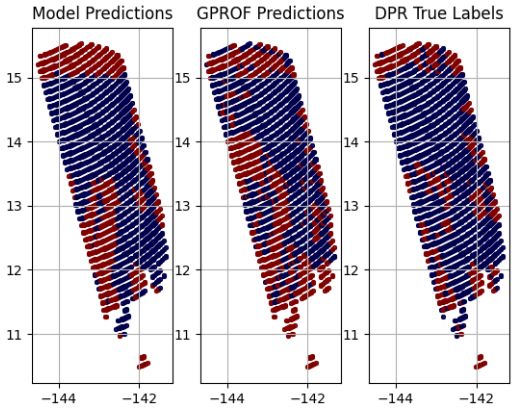


Figure 4.7 depicts the GP model and GPROF model on the left and middle respectively compared to the DPR derived true labels on the rightmost plot.

Ortiz et al. [5] analyzed the hurricane dataset as well, using BNNs to generate their predictions. Figure 4.8 is the visualization of their results, and a comparison with Figure 4.7 shows the difference between the GP based model and the traditional BNNs used in their research. The top row of Figure 4.8 shows the uncertainty metrics for variance, aleatoric, and epistemic uncertainty and the bottom row shows GPROF predictions, DPR-derived true labels, and Flipout model predictions with uncertainty annotations in the shading.

4.0.4 MCS Dataset Analysis

The MCS dataset results are consistent with the held-out test dataset and the hurricane dataset results with the GP model outperforming the GPROF model. The GP model had a

Figure 4.8. Plots of the Northeast Section of Hurricane Lane Observed by the GMI Instrument. Source: [5]

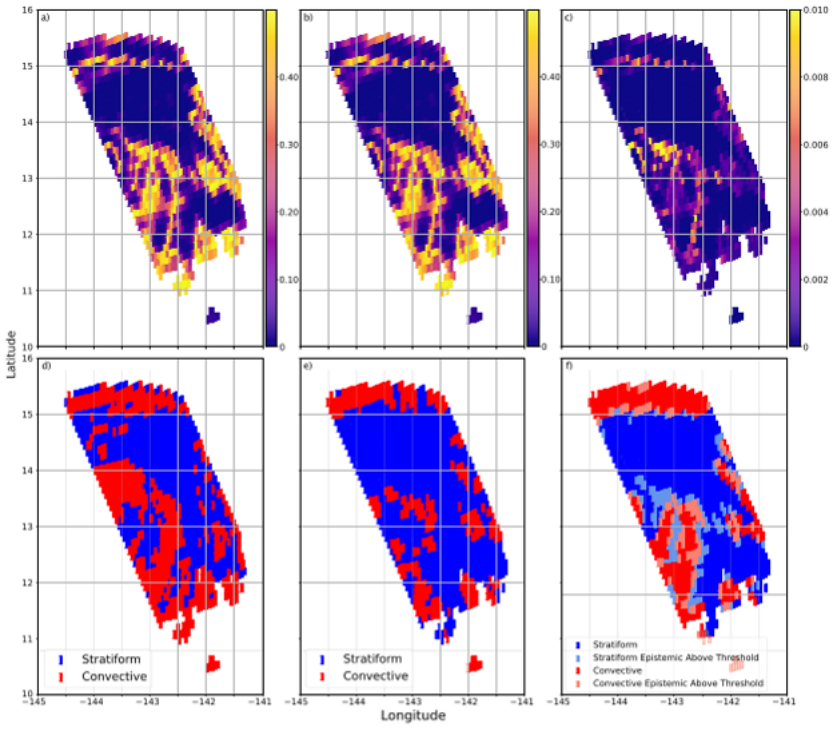


Table 4.6. Model Hurricane Confusion Matrix

		True Category		
		Positive	Negative	Total
GP Model Prediction	Positive	802	176	978
	Negative	47	262	309
	Total	849	438	1285

80.13% accuracy while the GPROF model was only 55.75% accurate.

4.1 Summary

In general, the GP model substantially improved upon the GPROF approach, and maintained that margin across all tested weather datasets. While uncertainty information was unable to be recovered from the GP models, GP models still performed better relative to deterministic

Table 4.7. GPROF Hurricane Confusion Matrix

		True Category		
		Positive	Negative	Total
GPROF Prediction	Positive	806	172	978
	Negative	56	253	309
	Total	10,797	3703	1285

Figure 4.9. Receiver Operator Curve (Hurricane Dataset)

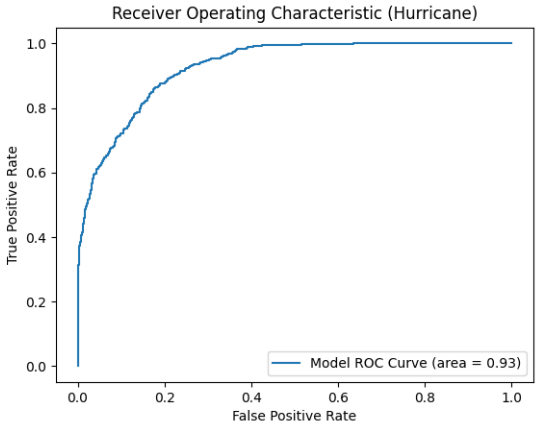


Figure 4.10. Precision-Recall Curve (Hurricane Dataset)

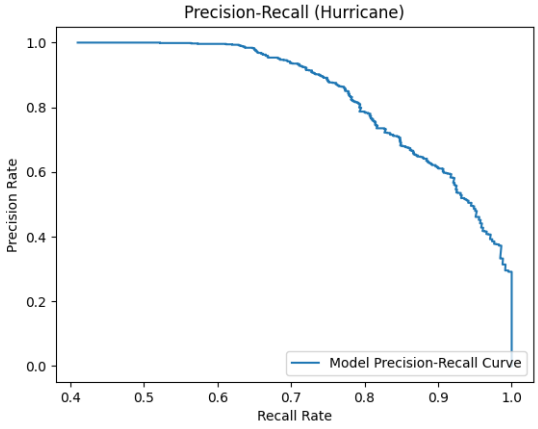


Figure 4.11. DET Curve (Hurricane Dataset)

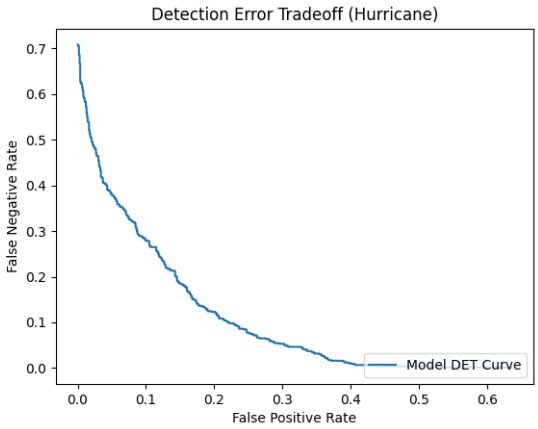
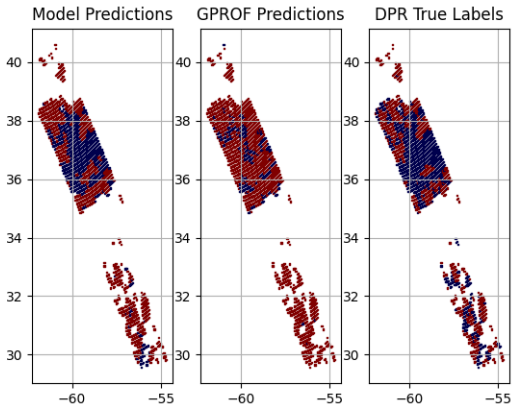


Table 4.8. Comparison Between Models on MCS Dataset

GPROF	0.558
ResNet with GP	0.801

Figure 4.12. MCS Map



ResNet50 models, and also performed slightly better than BNN models without filtering their output based on their uncertainty estimate. Only once the BNN models excluded samples from the test dataset that the model had low certainty about were the BNN models

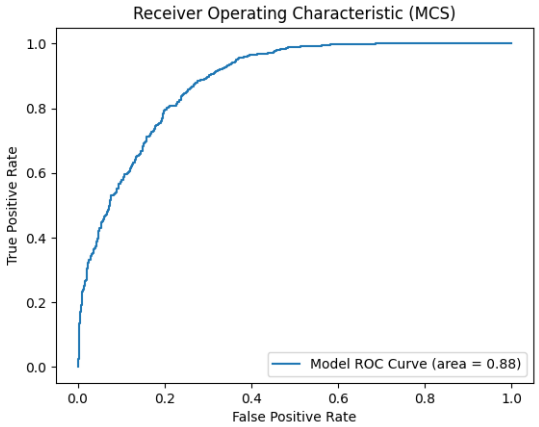
Table 4.9. Model MCS Confusion Matrix

		True Category		
		Positive	Negative	Total
GP Model Prediction	Positive	586	267	853
	Negative	54	709	763
	Total	654	962	1616

Table 4.10. GPROF MCS Confusion Matrix

		True Category		
		Positive	Negative	Total
GPROF Prediction	Positive	181	672	853
	Negative	43	720	763
	Total	224	1392	1616

Figure 4.13. Receiver Operator Curve (MCS Dataset)



able to outperform the GP model. This shows promise for GPs as an ML technique, as there was a small drop in performance from the deterministic models to the BNN equivalent before applying the uncertainty filter. If GPs are also able to provide well-calibrated uncertainty estimates then they would be able to offer outperformance of equivalent deterministic models for both uncertainty filtered and unfiltered predictions.

Figure 4.14. Precision-Recall Curve (MCS Dataset)

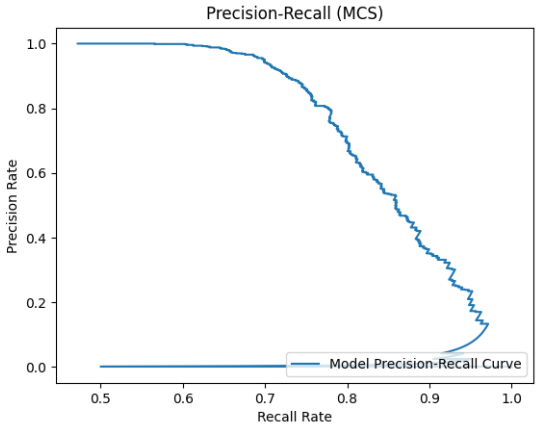
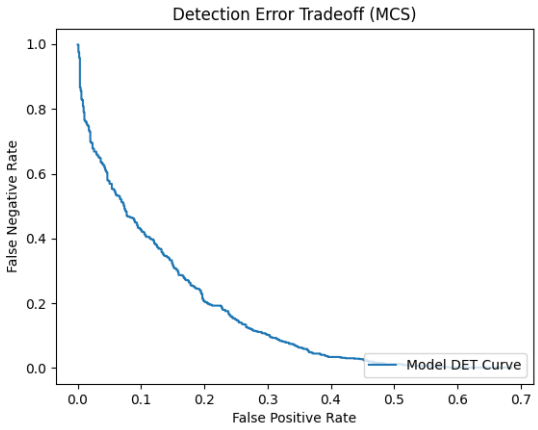


Figure 4.15. DET Curve (MCS Dataset)



THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusion

This chapter summarizes insights from the research, Department of the Navy relevance of the research, as well as recommends future work.

5.1 Summary Results

The following objectives were researched in this report:

1. How well do GPs classify precipitation type from satellite data?
2. How effective are GPs at providing an uncertainty estimate to output?
3. How do GPs compare to CNNs in training time and resources?
4. How do GPs compare to BNNs in training time and resources?
5. How does inference from GPs compare to traditional variational methods?

We will now examine each of these questions in turn based upon the results found earlier in this thesis.

Gaussian Processes had a classification accuracy of 87.45%, with a ROC AUC score of 0.872. This compared to the GPROF algorithm accuracy of 74.3%, deterministic model, and BNN models.

This classification accuracy outperforms the classification accuracy of the equivalent deterministic model, and substantially outperforms the accuracy of the differential equation based physics model, GPROF. Unfortunately, the covariance matrices were unable to compared for this dataset to the uncertainty produced by the BNN techniques. The amount of RAM required to train the covariance matrices was larger than the amount of RAM available in the GPU cluster. After switching to training on CPUs and providing 512GB of RAM, the covariance matrices were still unable to be trained due to lack of RAM. A comparison was made of the means with no additional information provided by the covariances, and further research would be needed to improve the variational algorithms for computing the covariance matrix to use on big data problems, such as presented by satellite data.

Compared to Monte Carlo techniques, GP's have an advantage in that only a single inference run is required, as opposed to multiple output runs required for each inference of a Monte Carlo model. This results in substantially lower compute required for inference from a GP model relative to an MCD model.

5.2 Impact on Naval Services

This research is highly relevant to the Navy, both through its focus on meteorological effects and its focus on quantifying uncertainty.

Line of Effort (LOE) 1 in the Department of the Navy's recent Climate Action 2030 strategy is climate-informed decision-making, which requires that leaders throughout the enterprise consider "climate change impacts, risks, and opportunities for adaptation, mitigation, and resilience benefits" [36]. This research provides enhancements on numerical meteorological techniques for predicting precipitation from satellite imagery, which is one of the best ways for estimating latent heat at planetary scale.

Additionally, the DON Strategy for Intelligent Autonomous Systems (IAS) lays out the Naval IAS Vision as the seamless integration of IAS as trusted members of the naval enterprise [37]. One of the most common critiques of AI/ML systems is that they present a black-box to users, requiring the user to trust a system that they are unable to understand that provides predictions without context. This research lays out an alternate path for AI/ML systems, one where the system's predictions are contextualized by estimates of the uncertainty of the prediction, estimates that not only provide context, but also break down the type of uncertainty allowing for decision-makers to make better decisions given input from an AI/ML system. This context is a valuable addition to a prediction in a human-machine team, and one that will help substantially in moving towards the Naval IAS Vision.

Gaussian processes add a level of explainability to neural networks that is unavailable with simple deterministic predictions. Even though the addition of Gaussian processes nominally increases the complexity of a network, that increase is offset by the simple interpret-ability of a mean and standard deviation in the feature space of the neural network. Even without a background in machine learning, that additional information can aid in military decision-makers understanding the context of the prediction and responding accordingly.

Machine learning based models also scale more efficiently than finite element modeling software from a Floating-point Operations Per Second (FLOPS) per watt perspective, providing more compute for a given amount of power input than is possible with CPU-based algorithms. This has downstream effects for meeting the Climate Action 2030 goal of reduction of power requirements, and also allows for larger models to be staged closer to the edge, for example on ships where there is constrained power budget.

5.3 Future Work

Recovering the covariance matrices produced by a GP model would be the next step to continue this research. We found here that relative to deterministic DCNNs and BNNs GP models are substantially slower to train, however the actual predictive results are similar. Comparing the uncertainty estimate of the model to those of traditional BNN techniques would be able to validate whether or not they still provide any value for that additional cost, even if the actual prediction from them does not outperform those other models.

One research direction that was unable to be explored in more detail is if the covariances found by the Gaussian processes have an explanation in the physics of either the satellite data-source or the meteorology of the observed area. As this research focus was confined to the machine learning and mathematical domain, no work was done on whether or not this uncovers any previously unknown terms in the predictive systems already in use by meteorologists. One of the benefits of Gaussian processes as a technique is that the processes are both fully described by their second-order statistics, mean and covariance, as well as human interpret-able as those statistics translate directly to intuitive meanings. As such, the existence or non-existence of covariances found by the Gaussian processes could highlight underlying correlations between either sensor spectrums on the satellite, or correlations with the climate systems occurring in the atmosphere.

The dataset used in this research has now been evaluated with deterministic ResNets as well as Bayesian techniques and Gaussian processes, however there are additional neural network architectures that have not been attempted to compare to these other techniques due to time and resource availability. In addition to the neural network architectures, there is a rich history of climate modeling that predates current machine learning trends and uses differential equation solvers to model the climate, and those have not been compared to

these techniques on this dataset. Further research using traditional finite element modeling, stochastic process equivalents to those models, and other neural network architectures may find improvements over the work that has been done so far.

5.4 Conclusion

Gaussian Processes are one of many potential methods for exploring the uncertainty of predictions in a neural network. They benefit from being relatively intuitive for a non-mathematician due to their full parameterization via the mean and covariance, while also providing effectiveness at machine classification. In situations where the additional compute is available for augmenting traditional deep learning techniques with them, they are an effective means to providing human decision-makers with a much needed estimate of uncertainty.

List of References

[1] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005. Available: https://doi.org/10.7551/mitpress/3206.001.0001

[2] D. Duvenaud, "Automatic model construction with Gaussian processes," 2014. Available: https://doi.org/10.17863/CAM.14087

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. Cambridge, UK: Springer, 2006.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. Available: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[5] P. Ortiz, M. Orescanin, V. Petković, S. W. Powell, and B. Marsh, "Decomposing satellite-based classification uncertainties in large earth science datasets," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2022.

[6] T. M. J. S. K. H. Wentz, F.J., "Remote Sensing Systems GPM GMI," 2015. Accessed: 2023-07-25. Available: https://www.remss.com/missions/gmi/

[7] NASA, "Instruments: Advanced Baseline Imager (ABI)." Accessed: 2023-07-25. Available: https://www.goes-r.gov/spacesegment/abi.html

[8] NASA, "GPM Microwave Imager (GMI)." Accessed: 2023-07-25. Available: https://gpm.nasa.gov/missions/GPM/GMI

[9] M. Orescanin, V. Petković, S. W. Powell, B. R. Marsh, and S. C. Heslin, "Bayesian Deep Learning for Passive Microwave Precipitation Type Detection," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022. Available: https://doi.org/10.1109/LGRS.2021.3090743

[10] J. Fischer, M. Orescanin, P. Leary, and K. B. Smith, "Active bayesian deep learning with vector sensor for passive sonar sensing of the ocean," *IEEE Journal of Oceanic Engineering*, vol. 48, no. 3, pp. 837–852, 2023. Available: https://doi.org/10.1109/JOE.2023.3252624

[11] U. Sahlin, I. Helle, and D. Perepolkin, ""this is what we don't know": Treating epistemic uncertainty in bayesian networks for risk assessment," *Integrated Environmental Assessment and Management*, vol. 17, no. 1, pp. 221–232, 2021. Available: https://doi.org/https://doi.org/10.1002/ieam.4367

[12] S. Haykin, *Neural Networks and Learning Machines* (Neural networks and learning machines). Prentice Hall, 2009, no. v. 10. Available: https://books.google.com/books?id=K7P36lKzI_QC

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, may 2017. Available: https://doi.org/10.1145/3065386

[14] M. D. Pritt and G. Chern, "Satellite Image Classification with Deep Learning," *2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pp. 1–7, 2017. Available: https://api.semanticscholar.org/CorpusID:52192868

[15] V. Petković, M. Orescanin, P. Kirstetter, C. Kummerow, and R. Ferraro, "Enhancing PMW Satellite Precipitation Estimation: Detecting Convective Class," *Journal of Atmospheric and Oceanic Technology*, vol. 36, no. 12, pp. 2349–2363, Dec. 2019. Available: https://doi.org/10.1175/jtech-d-19-0008.1

[16] Y. LeCun *et al.*, "Handwritten Digit Recognition with a Back–Propagation Network," *Advances in Neural Information Processing Systems*, vol. 2, 1989 [Online]. Available: https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf

[17] K. P. Murphy, *Probabilistic Machine Learning: An Introduction*. Cambridge, MA, USA: MIT press, 2022.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034. Available: https://doi.org/10.1109/ICCV.2015.123

[19] O. Durr, B. Sick, and E. Murina, *Probabilistic deep learning*. New York, NY: Manning Publications, Feb. 2021.

[20] G. I. Webb, *Bayes Rule*. Boston, MA: Springer US, 2010, pp. 74–75. Available: https://doi.org/10.1007/978-0-387-30164-8_62

[21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. Available: http://jmlr.org/papers/v15/srivastava14a.html

[22] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," *arXiv e-prints*, 2015.

[23] D. Kingma, T. Salimans, and M. Welling, "Variational Dropout and the Local Reparameterization Trick," *arXiv e-prints*, 2015.

[24] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, "Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches," *arXiv e-prints*, 2018.

[25] M. E. E. Khan, P. Baque, F. Fleuret, and P. Fua, "Kullback-Leibler Proximal Variational Inference," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, vol. 28. Available: https://proceedings.neurips.cc/paper_files/paper/2015/file/3214a6d842cc69597f9edf26df552e43-Paper.pdf

[26] M. N. Bernstein, "The Evidence Lower Bound (ELBO)," 2020. Accessed: 2023-07-26. Available: https://mbernste.github.io/posts/elbo/

[27] R. Kass, L. Tierney, and J. Kadane, "Laplace's method in Bayesian analysis," pp. 89–135, 1991. Available: https://doi.org/10.1090/conm/115/07

[28] S. Hochreiter, "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions," *Int. J. Uncertain. Fuzziness Knowl. Based Syst.*, vol. 6, pp. 107–116, 1998. Available: https://api.semanticscholar.org/CorpusID:18452318

[29] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," *Machine Learning*, vol. 110, no. 3, pp. 457–506, Mar. 2021. Available: https://doi.org/10.1007/s10994-021-05946-3

[30] Y. Kwon, J.-H. Won, B. J. Kim, and M. C. Paik, "Uncertainty quantification using Bayesian neural networks in classification: Application to biomedical image segmentation," *Computational Statistics Data Analysis*, vol. 142, p. 106816, 2020. Available: https://doi.org/https://doi.org/10.1016/j.csda.2019.106816

[31] S. S. N. Y. J. A. T. K. Robert Meneghini, Toshio Iguchi. "GPM/DPR Level 2 Algorithm Theoretical Basis Document (ATBD)." 2021.

[32] S. Powell, R. Jr, and S. Brodzik, "Rainfall-Type Categorization of Radar Echoes Using Polar Coordinate Reflectivity Data," *Journal of Atmospheric and Oceanic Technology*, vol. 33, 01 2016. Available: https://doi.org/10.1175/JTECH-D-15-0135.1

[33] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org. Available: https://www.tensorflow.org/

[34] D. Tran, M. Dusenberry, M. van der Wilk, and D. Hafner, "Bayesian layers: A module for neural network uncertainty," *Advances in Neural Information Processing Systems*, pp. 14 660–14 672, 2019.

[35] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, Y. Lechevallier and G. Saporta, Eds. Heidelberg: Physica-Verlag HD, 2010, pp. 177–186.

[36] D. of the Navy, "Climate Action 2030," 2022. Available: https://www.navy.mil/Portals/1/Documents/Department%20of%20the%20Navy%20Climate%20Action%202030%20220531.pdf?ver=3Q7ynB4Z0qUzlFg_2uKnYw%3d%3d&timestamp=1654016322287

[37] O. of Navy Research, "Science Technology Strategy for Intelligent Autonomous Systems," 2021. Available: https://www.nationaldefensemagazine.org/-/media/sites/magazine/2021-dist-a-don-st-strategy-for-intelligent-autonomous-systems-2-jul-2021.ashx

Initial Distribution List

1. Defense Technical Information Center
   Fort Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California