Advances in Intuitive Priors and Scalable Algorithms for Bayesian Deep Neural Network Models in Scientific Applications

by

Jeremiah M. A. Hauth

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in the University of Michigan
2024

Doctoral Committee:

Assistant Professor Xun Huan, Chair
Assistant Professor Peng Chen, Georgia Institute of Technology
Professor Kathleen Sienko
Associate Professor Jenna Wiens

Jeremiah M. A. Hauth
hauthj@umich.edu
ORCID iD: 0009-0009-3755-2398

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Xun Huan, whose technical expertise, guidance, and unwavering support have been instrumental throughout the entire journey of this dissertation. Your encouragement, constructive feedback, and empathy have shaped this work and my academic growth.

I would like to thank my long-time research collaborators and committee members, Kathleen Sienko and Jenna Wiens for the tremendous amount of insight, feedback, and guidance you have provided me over the last several years in our loss of balance, precision health, and M3X research projects. I would also like to thank my committee member, Peng Chen, whose algorithmic research contributions substantially influenced the direction of this dissertation. I also thank collaborators Beckett Zhou, Nikolai Banovic, and Reese Jones for your research mentorship across the many projects that I have worked on.

I would like to thank several of my long-time graduate student collaborators Safa Jabri, Jamie Ferris, and Snehal Prabhudesai for the countless hours and late nights you spent helping me to review code, analyze results, prepare for research meetings, and draft papers. I would also like to thank the entire Huan UQ group for all of your help and support over the years!

My deepest gratitude goes to my family for their unconditional love. I am profoundly grateful for all the encouragement that you have provided throughout my life to get me to where I am today.

Thank you to all my amazing friends, in Michigan and abroad— Marshall and Kai for having my back long before I came to Michigan and having my back long after leave. Alex, Sammy, Matti, Max, Patsy, Katie, Travis, Erin, and Tom for being inspiring teammates and competitors in beach volleyball, kickball, triathlon, cycling, relay race-running, and orienteering and for being such fun, silly, chaotic friends. I always look forward to our next adventure! And lastly, Tessa for being my dearest friend and closest companion. I can't wait to see what trails we blaze next.

Thank you to my bicycle for the countless hours of therapy, stress reduction, and convenient transportation it has provided me over the years and thank you to my dog Gil for being none of those things, but being worthy of love nonetheless.

In conclusion, this dissertation would not have been possible without the invaluable contributions of my advisor, fellow researchers, family, friends, bike, and dog. Each of you has played a crucial role in shaping this academic achievement, and for that, I am sincerely thankful.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

LIST OF ACRONYMS

AI artificial intelligence

BNN Bayesian neural network

CAA computational aeroacoustics

CDF cumulative distribution function

CDSS clinical decision support system

CFD computational fluid dynamics

CNN convolutional neural network

DL deep learning

DNN deep neural network

ELBO evidence lower bound

FCVI full covariance variational inference

GCNN graph convolutional neural network

GPU graphics processing unit

GRU gated recurrent unit

HD Hamiltonian dynamics

HMC Hamiltonian Monte Carlo

IMU inertial measurement unit

KL Kullback-Leibler [divergence]

KSS Kolmogorov-Smirnov statistic

LSTM long short-term memory

MAP maximum *a posteriori*

MCMC Markov chain Monte Carlo

MFVI mean-field variational inference

MH Metropolis-Hastings

MSE mean squared error

ML machine learning

MRI magnetic resonance imaging

NN neural network

pBNN partially Bayesian neural network

PDF probability density function

PINN physics-informed neural network

pSVGD projected Stein variational gradient descent

PT physical therapist

QoI quantity of interest

SciML scientific machine learning

SI sensitivity index

SVGD Stein variational gradient descent

UQ uncertainty quantification

VI variational inference

ABSTRACT

In recent years, deep learning (DL) algorithms have gained widespread use in scientific and engineering fields, promising insights into complex trends within extensive datasets. However, these models typically lack transparency and interpretability and the quantification of uncertainty in DL models, especially related to the understanding derived from the quality and quantity of training data, remains an underexplored research area. This dissertation addresses this gap by advancing Bayesian uncertainty quantification (UQ) methods in large-scale deep neural networks (DNNs), specifically constructing Bayesian neural networks (BNNs).

This dissertation proposes two methodological improvements to BNNs: the first is a parameter subselection procedure that leverages gradient based sensitivity analysis to select only the most impactful DL parameters for Bayesian inference; the second contribution is a prior selection methodology that weighs both expert knowledge of the predictive space alongside as well as desirable regularizing effects in the weight space.

This dissertation goes on to implement Bayesian neural networks and these novel methodologies in four unique scientific machine learning case studies, two related to physics simulations and two related to real-world health data. These case studies include: a novel framework for remotely detecting ice accumulation on helicopter rotor blades and assessing flight performance degradation; an investigation on the temporal evolution of uncertainty in Bayesian graph convolutional neural networks when predicting stress response in polycrystalline materials; an investigation of the uncertainty in the state-of-the-art U-NET model for brain tumor segmentation; and a novel framework for automatically assessing physical therapy patient performance on balance training exercises, along with preliminary approaches for future exercise recommendation.

By drawing new insights into model uncertainty across diverse science and engineering applications, this research aims to provide greater understanding of uncertainty in Bayesian neural networks, to help mitigate the consequences of model overconfidence, and to provide critical metrics for decision-making and data collection.

CHAPTER 1

Introduction

1.1 Opportunities

1.1.1 Scientific Machine Learning

Scientific machine learning (SciML) [1] is an interdisciplinary field that brings together the principles of machine learning (ML) and scientific computing. The goals of SciML are to develop models and algorithms that can learn from scientific data and simulations, and to use that knowledge to improve our understanding of complex systems and make predictions about their behavior [2]. The targeted scientific applications often involve multiscale and multiphysical dynamics under diverse sources of uncertainty and noise, and where decisions carry high consequences. As pointed out in [1], it is therefore critical to strengthen the scale, rigor, robustness, and reliability of SciML methods.

SciML is particularly useful in scientific domains where traditional modeling and simulation techniques may be computationally expensive, or where the underlying physical laws are poorly understood or unknown [3]. In these cases, ML can provide advantages in extracting patterns and relationships from datasets, and to create predictive models that can be used to guide scientific inquiry. In particular, deep learning (DL) is a subset of ML that utilizes highly parameterized deep neural network (DNN) models to learn and express complex, nonlinear interactions. For example, DL-based models have recently had tremendous success in applications in medical image segmentation, where DNNs with tens of millions of parameters have been used to parse through magnetic resonance imaging (MRI) scans containing millions of voxels to section out dangerous anomalies in just a few seconds [4]. Accuracy is comparable or even superior to a trained radiologist that might require several hours to complete the same task [4]. This task will be explored more in Chapter 6. Other DL successes in scientific domains include protein folding [5], molecular discovery [6], particle physics [7], fluid dynamics [8], solid mechanics [9], material science [10], and many more. SciML has the potential to transform many scientific domains, however it also

presents several challenges, including the need for large amounts of high-quality data, for robust validation and uncertainty quantification, and for interpretability and transparency [3].

It should be noted that within SciML, there is a very active field of research often referred as *physics-informed ML* [11, 12, 13] that aims to combine known physical principles with data-driven constructs. For example, physics-informed neural networks (PINNs) [14] find solution mappings by injecting loss terms based on violation residual of differential equations and initial boundary conditions that should obeyed, while neural operators [15] learn from data the rate-of-change of quantities that can generalize over time. While all of these are exciting research directions, they are not the focus of this dissertation. Instead, I will work with purely data-driven methods and investigate their roles in a range of scientific applications.

1.1.2 Uncertainty Quantification

Uncertainty quantification (UQ) is the process of assessing and characterizing uncertainty in scientific models, simulations, and measurements. UQ tasks involve representing and propagating uncertainties from their sources through the model to produce a range of possible prediction outcomes, and updating these uncertainties when new evidence (e.g., observation data) becomes available. This range of outcomes is often represented using probability distributions or other statistical measures, such as confidence intervals. UQ is a challenging and computationally intensive task, as it often requires running multiple simulations or models with different inputs or parameters. However, it is an essential aspect of scientific inquiry, as it allows researchers to understand the limitations of their models and the range of possible outcomes, so to make more informed decisions based on these results [16].

Within UQ, sources of uncertainty typically fall within one of two categories, epistemic and aleatoric uncertainty. Epistemic uncertainty comes from a lack of knowledge. This commonly comes from a lack of data, and can be reduced as data becomes less sparse. This uncertainty can also come from modeling assumptions, like unknown or neglected physics. In a data-driven approach, the type of model used (linear, random forest, deep neural network, etc) and how it is parameterized comes with limitations and assumptions that induce epistemic uncertainty. Aleatoric uncertainty comes from random noise or chance that cannot be reduced without a different data collection process. Sensors, whether analog or digital, have limitations to their accuracy, which is a type of aleatoric uncertainty. Some physical phenomenon, such as sub-atomic particle motion, is inherently probabilistic. Other processes, such as the jitter of stock prices on a second by second basis, is so complex that it becomes effectively random. These too are considered aleatoric uncertainty. In ML models, aleatoric uncertainty is often present in both features and labels.

In DL, UQ is particularly useful for conveying trust and transparency to otherwise uninter-

pretable, blackbox models and their predictions. At the minimum, UQ provides a means for the DL model to indicate when it is confident and when it is not. However, the notion of trust in ML is highly complex, and enabling uncertainty would still be only a small step towards achieving trustworthiness. Furthermore, UQ needs to be done carefully, since wrong predictions with low uncertainty would actually erode trust for the users of the ML model. UQ is also useful for guiding decision-making based on those predictions [17]. This need is further elevated in science and engineering domains where data can be limited, expensive, and often noisy and indirectly observed. Uncertainty can be used as an important metric to guide new data collection, such as in the cases of active learning and experimental design. While differing slightly in flavor, both approaches employ principles to target new data that explore the model space with high uncertainty or high potential for uncertainty reduction. Finally, UQ is essential for risk assessment and decision-making. In science and engineering, model predictions will often lead to real-world implications and consequences, for example, that involve people's health and safety. It is thus essential for end-users of these models to understand both the likelihood of different scenarios that could take place as well of the consequences of these outcomes [18].

While there are different mathematical frameworks to quantifying uncertainty, we will focus specifically on Bayesian UQ. By deriving the rules of probability from an extension of symbolic logic [19], Bayes' theorem provides a systematic approach to update the distribution of plausible outcomes through conditioning on data, taking into account their quality and quantity.

Bayesian inference starts by forming a prior probability distribution about the possible values of model parameters of interest, which may entail physical quantities (e.g., gravitational constant, Young's modulus) or abstract variables (e.g., weight parameters of a DNN). In the subjective Bayesian interpretation, these prior distribution can be formed in an informative manner based on the existing knowledge of the user. Alternatively, following the objective Bayesian interpretation, prior distributions can be selected for their ability to regularize a model and induce better generalizability on unseen data. In fact, under uncertain regression conditions, distributions such as the Laplace distribution and Gaussian distribution are Bayesian analogs to L1 and L2 regularization in frequentist settings. Other times, priors are selected for their uninformativeness or ability to not skew the posterior, as in Jeffrey's prior [20], the maximum entropy prior [21], or in DNN settings, wide Gaussian priors. How to select priors on DNN weights remains and open question, and concerns have been raised regarding the misspecification of DNN priors, particularly with the most common weakly-informative Gaussian priors that are so commonly used [22, 23].

A major component of Bayesian inference that distinguishes it from frequentist methods is this incorporation of prior information via the prior distribution. In the case of selecting priors on DNN weights, it poses a challenge due to the lack of interpretability in the abstract weight space. That is to say that while assigning prior distributions to weights is trivial, understanding the way in which

these priors encode information, particularly its effect on the output space, is non-trivial.

As more data relevant to that parameters is acquired, a user's belief state on the value of that parameter is also updated into the posterior distribution—that is, the conditional distribution of the parameters given the new data. This Bayesian approach to uncertainty is very powerful, as it not only follows directly from laws of logic and probability, but it also intuitively mirrors the way that many humans understand the world—starting with an initial (prior) hypothesis of how things work, and then updating this hypothesis in light of new information that support or refute different possibilities. One major challenge of Bayesian UQ is then to calculate this posterior probability distribution or, more commonly, to sample from it. While sometimes analytical posterior distributions are available, there is typically not a close-form solution. This necessitates the work of computational Bayes that focuses on developing and studying numerical methods for performing Bayesian inference.

This thesis will focus on enabling Bayesian UQ for DNN-based models. The Bayesian treatment of DNNs is also known as **Bayesian neural networks (BNNs)**.

1.2 Background

The thesis will take a two-pronged approach, making advanced in both the algorithmic and application sides of Bayesian UQ in SciML. A brief algorithmic background, with a focus on existing algorithms, is presented below first, followed by short introductions of the science and engineering application problems to be tackled.

1.2.1 Algorithms for Bayesian Inference

1.2.1.1 Markov Chain Monte Carlo (MCMC)

The workhorse methods for Bayesian inference is perhaps the Markov chain Monte Carlo (MCMC) class of algorithms. MCMC works by generating a sequence of random samples by constructing a Markov chain whose equilibrium distribution is the target distribution (which is set to the posterior distribution in the context of Bayesian inference) [24, 25, 26, 27, 28]. Roughly, this is done by drawing a new candidate sample based on a proposal distribution, evaluating the target probability density function (PDF) at the current and candidate samples, and then accepting or rejecting the proposed candidate based on the ratio in their PDF scores.

One of the key advantages of MCMC methods is that they can be used to sample from general probability distributions that may be difficult or impractical to sample using traditional methods such as rejection sampling or importance sampling. Additionally, the target distribution need not be normalized, eliminating the challenge of dealing with the difficult model evidence (marginal likelihood) term that arises in Bayes' theorem. Lastly, MCMC is considered to be an "exact"

algorithm with ergodic theorems guaranteeing its convergence in the asymptotic (infinite steps) limit. Despite their advantages, MCMC methods are not without limitations.

One major challenge is ensuring that the Markov chain has sufficiently converged to the desired equilibrium distribution under finite steps, which can be difficult to assess in practice. In addition, MCMC methods can be computationally intensive, especially when dealing with high-dimensional parameter spaces. This is partially addressed by using more advanced proposal distributions that more intelligently explore the target distribution. Whereas early MCMC methods (e.g., Metropolis-Hastings) use fixed Gaussian proposals that can lead to extremely low acceptance rates in high dimensions, modern MCMC (e.g., Metropolis-Adjusted Langevan Dynamics (MALA) [29], Hamiltonian Monte Carlo (HMC) [30], and No-U-Turn Samplers (NUTS) [31]) have adopted gradient information to guide proposals of new samples with lower rejection rates. Nevertheless, the number of samples needed to reasonably capture the equilibrium distribution grows substantially in higher dimensions as a result of the geometric limitation from the "typical set" [32], making MCMC unsuitable for very large scale inference problems.

1.2.1.2 Variational Inference

Variational inference (VI) [33, 34, 35] adopts a different strategy than MCMC: instead of sampling from the posterior distribution, VI seeks to directly approximates the posterior in a parametric form. This entails first selecting a parametric family of distributions, and then optimizing in this parametric space to find the specific distribution that is closest to the true posterior distribution according to some distance measure. Most commonly, this measure is the Kullback-Leibler (KL) divergence from the true posterior to the approximate posterior represented by the variational distribution. This is sometimes referred to as the 'reverse' KL divergence, to distinguish it from the 'forward' KL divergence that is from the approximate posterior to the true posterior. Unless otherwise specified, the KL divergence used throughout this dissertation will refer to the 'reverse' formulation. The KL divergence is typically selected for its computationally tractability and for its significance in information theory, however other metrics, such as the Wasserstein distance, have also been explored. The best approximate distribution is when the KL divergence is minimized. In practice, this KL term can be transformed to the evidence lower bound (ELBO), which can be estimated efficiently using Monte Carlo sampling.

One of the main advantages of VI is that, by effectively transforming the inference problem into an optimization one, it is much more scalable to high-dimensional parameter spaces and to handle complex dependencies. The major drawback of VI is that its approximation capacity is constrained by the variational family, and it is no longer an exact algorithm like MCMC. For example, while simple to use, mean-field Gaussian VI are unable to capture correlation and non-Gaussian structures; however, the variational space can be enriched by advanced parameterizations

such as normalizing flows. Furthermore, VI does not carry any convergence guarantees, even as the approximating families become more complex.

1.2.1.3 Stein Variational Gradient Descent

Stein variational gradient descent (SVGD) is a particle-based algorithms used in ML and Bayesian inference to approximate complex probability distributions, first proposed by [36]. The basic idea behind SVGD is to transform the target distribution into a set of particles, each of which is updated using a gradient-based optimization algorithm. The update rule is designed to move the particles towards regions of high probability density, via the Stein operator, while also encouraging diversity (variance) among the particles.

A key innovation of SVGD is the use of a kernel function to define a distance metric between the particles, which is used to compute the gradient of the optimization objective. This allows the algorithm to effectively explore the high-dimensional parameter space and converge to a high-quality approximation of the target distribution. Importantly, SVGD does not require explicit knowledge of the target distribution and does not require an explicit variational family of approximate posteriors.

One of the main advantages of SVGD is its ability to scale to large datasets and high-dimensional models, which can be difficult or impossible to handle using MCMC or VI. In addition, SVGD can be easily parallelized and can be applied to a wide range of models, including those with complex dependencies and non-standard distributions. However, the quality of the approximation depends on the choice of the kernel function, the number of particles used. Like most optimization algorithms, this algorithm is sensitive to initial conditions and the exact choice of optimizer and may require careful tuning and parameter selection. One major shortcoming with SVGD is the tendency for the posterior to "collapse" in high-dimensions [37]. That is to say, without enough particles or with a poor choice of kernel, the kernel might lack the expressivity to adequately explore both high and low density regions of the target posterior distribution. As such, particles will clusters around the mode of the posterior and underestimate posterior variance. While increasing the number of particles can alleviate this issue, this might not be computationally feasible in very high dimensional problems.

1.2.1.4 Projected Stein Variational Gradient Descent

Projected SVGD (pSVGD), proposed by [38], advances on the original SVGD algorithm to greatly alleviate the tendency of posterior collapse. The key innovation of pSVGD is to project the posterior onto a strategically selected low-dimensional subspace such that the number of particles is not dwarfed by the number of dimensions. That is, the number of particles per dimension is effectively increased. The choice of subspace is a likelihood-informed active subspace that is

most-informed by the data. The complementary inactive subspace is minimally informed by the data and can be frozen to remain at the prior.

Much more details of these algorithms, including their mathematical formulations, will be presented in Chapter 2.

1.2.2 Simulation-based Physics Applications

In addition to algorithmic investigations, this thesis also seeks to bridge theory to application, by implementing and exercising the computational methods on complex problems encountered in real-world science and engineering. To date, the have been few large-scale implementations of BNNs, particularly with promising Bayesian inference algorithms such as SVGD and pSVGD. Recently, [39] demonstrated a deep convolutional autoencoder with 241,164 parameters and 20 SVGD particles for a SciML task of modeling fluid permeability in random porous media. However there is a large research gap in identifying suitable real-world applications for BNN. There are also a number of unique, domain-specific adaptions needed to effectively implement BNN in scientific applications, particularly as an uncertainty-aware surrogate model. Uncertainty is an important component in physics models, and even more-so in surrogate models. Typically, advanced physics models are used to aid design and design of physical experiments. Surrogate models are approximations of higher fidelity models that are designed to be much cheaper to run, but may lack the accuracy, resolution, or even convergence guarantees of high fidelity numerical modeling techniques. UQ is also an important tool for guiding additional data collection and is also valuable for understanding the limitations of surrogate models. This section presents background information on two BNN applications as a computational physics surrogate models, each of which is developed further in Chapters 3 and 4.

1.2.2.1 Helicopter Icing Performance Evaluation

In-flight ice formation poses a major danger for rotorcraft (e.g., helicopter) operations, and detection of ice formation is paramount for operational safety. Direct detection methods such as infrared thermography [40], optical reflectometry [41] and ultrasonic wave techniques [42] have received the most scientific interest but have either struggled with implementation on moving parts or with commercial viability. Indirect methods compare established aircraft states with potentially iced states to deduce the presence of ice. This work focuses on developing an indirect data-driven method of mapping acoustic signature to flight performance techniques. I accomplish this mapping with Bayesian dense neural networks.

In an offline phase of the work, the acoustic signature of glaze and rime ice shapes on an oscillating wing are computed via computational fluid dynamic simulations. In addition, aerodynamic

performance indicators corresponding to the ice shapes are also monitored. These performance indicators include the lift, drag, and moment coefficients. These data pairs of high-dimensional acoustic signatures and performance indicators are well suited for mapping via DNNs. In addition, the health and safety nature of monitoring icing and its performance impact necessitates a strong understanding of the model limitations and the degree of uncertainty in its assessments.

A BNN is subsequently trained using pSVGD to create a mapping from the acoustic signature generated by the iced wings to predict their performance indicators along with quantified uncertainty that is highly important for time- and safety-critical decision-making scenarios. While the training is carried out fully offline, usage of the BNN to make predictions can be conducted rapidly online allowing for an ice detection system that may be used in real-time and in-flight.

DNNs have previously been utilized in identifying rotorcraft based on acoustic signature [43]. DNNs have also been used to diagnose icing based on visual data of airfoils but this approach fails to directly quantify the impact on flight performance[41, 40]. There is a substantial discrepancy in flight performance impact between more mild rime ice and more severe glaze ice that is not accounted for in simply identifying the presence or absence of ice. My approach is the first to directly assess the impact of ice and the first to do so based on acoustic signature. Furthermore, mine is the first approach to also provide uncertainty information with these predictions.

This use-case highlights the ability of even a simple, dense neural network to accurately map high-dimensional aeroacoustic data to flight performance indicators. Furthermore, I was able to demonstrate the effectiveness of SVGD and pSVGD in quantifying uncertainty in these neural networks. This uncertainty is valuable in conveying risk and transparency to end users. One line of future work will focus on adapting the BNN approach to recurrent networks, to allow for real-time performance evaluation without the need for frequency binning of the aeroacoustic data. Another line of future work lies in using the uncertainty in the acquisition function of an active learning framework. This work will be presented in Chapter 3.

1.2.2.2 Polycrystal Stress Response

The material response functions of many real-world materials depend on the microstructure, which includes variations in the phase and orientation of constituent crystals. To give just one example, different grades of steel are primarily differentiated by their microstructure. Depending on how the steel has been tempered, quenched, and the exact ratio of iron to carbon and other trace elements, the small crystals that make up the grain of the metal might form pearlite, ferrite, cementite, austenite, and other phases and each crystal takes on a unique spatial orientation. Their exact arrangement contributes to the unique yield strength, elastic modulus, brittleness, and ductility of the bulk material and differentiates that particular grade of steel from other grades. While monocrystalline materials do exist, most metals, alloys, and ceramics, as well as polymers and composite materials

are made up of a polycrystalline microstructure.

To predict the mean response of a microstructure sample to external loading, homogenization is necessary for developing subgrid models and exploring structure-property relationships. However, using the graph induced by the segmentation of microstructural fields to learn response functions directly can be difficult as this representation does not encode all the information of the full field.

To address this challenge, [44] proposed a DL method based on hidden features on a reduced graph that utilizes the native discretization and segmentation of the input field. The reduced graph represents regions as nodes that are associated with features, and a subsequent multi-level graph convolutional neural network (GCNN) model is used as the basis for analysis. Reducing the graph before processing it with convolutional layers has several benefits, such as interpretable features and efficiency on large meshes.

However, one drawback of this approach is that the GCNN only predicts the mean response to external loading, without capturing the non-negligible variance of the response. In order for the GCNN surrogate model to be useful for design applications, uncertainty quantification must be implemented in order to determine the potential for deviation from higher fidelity methods.

The work presented in Chapter 4 investigates scalable Bayesian methods for quantifying the variance that is innate to the microstructure response predictions. Open questions involve identifying and evaluating scalable Bayesian inference algorithms for this task as well as evaluating suitable time marching approaches, such as architectures with long short-term memory (LSTM) and neural ordinary differential equations (NODEs).

1.2.3 Real-world Health Applications

In addition to investigations centered around building surrogate models for higher fidelity physics simulations, this thesis also seeks to explore real-world applications in the health sciences. These real-world applications carry unique challenges. The first is that the data is particularly small and noisy. Data can be sensitive to sensor accuracy, sensor calibration, and the potential for human error in data collection. And while physics models typically have a "true" underlying model governed by known laws of physics, medical data can feature a complexity that is well beyond what can be modeled using governing equations. As such, any data-driven model that is derived from medical data typically cannot be compared to a "true" model, in that a "true" model likely does not exist. Going back to the concept of aleatoric uncertainty presented earlier in the chapter, there is a limit to what can be known. This makes UQ particularly necessary in the health science domain.

Furthermore, modeling in the health domain is intrinsically linked to the concept of risk. Model errors and misclassifications in health modeling could lead to improper treatment and even physical injury. If risk is conceptualized as the intersection of consequences and uncertainty [45], and the

potential consequences in this domain are innately high, then it is imperative that uncertainties be well understood.

Of course, understanding all sources of uncertainty in a given domain is challenging. This work is limited in its scope in that it only tries to address epistemic uncertainty in model parameters and aleatoric noise in data labels. Other sources of uncertainty such as model misspecification, feature noise, and sensor accuracy are not considered.

Chapter 6 investigates a Bayesian implementation of the popular U-Net segmentation architecture [4] for the purpose of identifying tumors in MRI scans. Specifically, these scans are mapped to radiologist-annotated boundaries of the tumors using BNN. The U-Net model also poses a challenge due to its very large model size, and this challenge is partly addressed by my novel methodological contribution of *partially* Bayesian neural networks.

Chapter 7 investigates a Bayesian ML tool for automatically assessing a patients balance ability in a precision health balance rehabilitation setting. Kinematic information of a balance exercise from an inertial measurement unit (IMU) sensor is mapped using BNN directly to a rating assigned by a physical therapist (PT) observing the exercise. With the end goal of this project being to recommend future exercises to a patient, it is imperative that recommendations be accurate, which in turn depends strong on the assessment of current exercises. Injury should an exercise be too difficult and stagnated recovery should an exercise be too easy mean that the consequences of poor prediction are high. As such, it is again imperative that uncertainty be well understood, and it forms an important component to advancing to clinical implementation of a balance exercise recommendation tool.

1.2.3.1 Bayesian Medical Image Segmentation

The work presented in Chapter 6 focuses on the task of segmenting brain tumors in MRI modalities. Tumors are clusters of cells with abnormal growth patterns can can be either benign (non-cancerous) or malignant (cancerous). Both can interfere with normal function of the brain, though cancerous are especially dangerous as they can spread to other tissues and other organs.

Presence of tumors are typically confirmed using magnetic resonance imaging (MRI), which can produce images of organs inside the body. Different modalities are produced by changing the pulse and response time of the MRI radiowaves, and also by introducing contrast enhancement agents. MRI scans are invaluable to planning treatment, as accurately removing the tumor and avoiding healthy tissue is challenging. Typically, this process is aided by radiologist-annotated scans indicating the boundaries of the tumor. This is a labor-intensive process and radiologists require many years of training. In recent years, ML image segmentation algorithms have shown to be a promising approach for automating this expensive annotation process.

Broadly speaking, image segmentation in the context of ML is the process of automatically

parsing an image into labeled segments for the purpose of identifying objects and their boundaries within the image frame. This is of particular value in medical imaging, as it poses a means to automatically identifying structures of interest in X-ray images, CT scans, and MRI scans. To date, one of the most successful DL implementations of medical image segmentation is the U-Net architecture [4]. Through a series of cascading convolutional layers interconnected with a series of cascading deconvolutional layers, this architecture excels at image-to-image segmentation.

Typically, ML models for segmentation, including U-Net, represent pixels as Bernoulli distributions (or multinomial distributions in the case of multiclass problems) that express the probability of the presence of a structure of interest, such as a tumor. While these probabilities can be calibrated to the incidence of correct / incorrect predicted labels, these AI-based clinical decision support systems (CDSS) that produce single-valued (i.e. deterministic) predictions do not reflect the inherent uncertainty in medicine [46], and encourage impressions of "superhuman" ability of AI [47] without mechanisms to contest these claims [48].

I address some of these shortcomings by adapting the U-Net architecture to represent trainable weights and biases through mean-field VI (MFVI), using independent Gaussian distributions to approximate the true posterior. However, even MFVI struggles when the number of DNN parameters (i.e., the dimension of Bayesian inference) goes to millions. Chapter 5 will thus introduce a new technique that uses local gradients as a low-cost heuristic to focus MFVI on a subset of the most impactful weights and biases, thus inducing a 'partially Bayesian' neural network (pBNN).

The work in Chapter 6 develops a computationally efficient, Bayesian adaptation of the state-of-the-art U-Net medical image segmentation architecture. This chapter demonstrates a strong negative correlation between the normalized gradient of a single layer of weights and the KL divergence between the push-forward posterior distributions of the partial Bayesian U-Net and fully Bayesian U-Net. pBNNs identified with this heuristic are able to capture the majority of the Bayesian uncertainty of a full BNN but with a much smaller memory footprint and with much lower training times.

1.2.3.2 Precision Health Balance Assessments

Losses of balance and falls are a consistent concern for much of the population, particularly older adults and those with inner ear disorders [49, 50]. Balance training via physical therapy can be an effective treatment for poor balance. To date, the most effective physical therapist programs require in-person supervision from a physical therapist (PT) [51, 52] with remote supervision often lacking the real-time feedback and personalization [53, 54].

Recent progress has been made in remote supervision by utilizing DL tools to map motion data collected from patients via inertial measurement units (IMU) to corresponding balance assessments made by PTs [55, 56]. Such DL tools can provide accurate and near-instantaneous automatic balance

assessments, which can hasten in-home balance recovery. Furthermore, such tools can leverage the assessments of multiple PTs, potentially mitigating the biases of individual PTs. Using ML in physical therapy, like many other fields of medicine, requires great care to ensure that the model can be trusted and that risk is mitigated to any potential patient. The consequences of poorly assessed and prescribed exercises can be quite severe and as such, mitigating risk in an ML expert system requires quantifying the uncertainty of any predictions.

DNNs often fail to sufficiently convey appropriate uncertainty information, which is critical to both medical professionals and patients alike. I quantify uncertainty in DNN models by performing Bayesian inference for their weight parameters, i.e., creating BNNs. With parameter dimension often reaching hundreds of thousands for DNNs, I conduct approximate inference using SVGD. Parameter dimensionality is further reduced by projected parameters onto Hessian-informed active subspaces, via projected SVGD. Prior distributions for the DNN weights also tend to be very abstract, and at times arbitrary. I illustrate that priors on DNN parameters can in fact be used to create intuitive prior predictions while also inducing prediction-enhancing regularization. Furthermore, I also explore a hierarchical Bayesian framework, in which the model can learn the quality of data and carefully weigh the impact of the data on modifying prior.

This use-case highlights the ability to adapt the SVGD-BNN approach to scale to larger and more complex models. This work is presented in Chapter 7. Furthermore, this work demonstrates the effectiveness of the intuitive prior heuristic, described in Chapter 5, on noisy, real-world data. Future work will focus on developing an exercise progression algorithm that makes use of these exercise assessments to suggest new exercises for a PT patient to work on.

1.3 Thesis Objectives

The main objectives and contributions of this thesis are summarized as follows.

1. **Enable large scale Bayesian neural networks**
   The first objective of this work is to enable large scale BNNs. Part of this will be accomplished through a substantial literature review to understand current scalable Bayesian inference algorithms, particularly in the variational and Stein variational classes of algorithms. A subset of these scalable algorithms will be implemented and validated against state-of-the-art sampling algorithms, such as Hamiltonian Monte Carlo in order to determine the Bayesian inference algorithms most well-suited for enabling BNN. This work also seeks to develop a general purpose approach to rapid scalability of BNN via Bayesian parameter reduction methods.

2. **Develop predictive-informed prior selection**

The second objective is to develop a method for intuitive predictive-informed prior selection. These methods are particularly valuable for bringing interpretation and rationale for prior selection for non-physical parameters such as DNN weights. This new approach will combine the performance-enhancing regularization effects with prior predictive distributions that accurately encode prior information about the model predictive.

3. **Demonstrate real-world applications**
   The third objective of this dissertation is to identify novel and appropriate applications for BNN in the domain of science and engineering. This work will adapt BNN architectures and prior selections for the specific applications outlined above. These include a novel technique for using acoustic signature of iced rotorcraft airfoils to directly predict their flight performance indicators via dense neural networks; assessing the time evolution of crystalline structure strength profiles via graph neural networks; identifying potential tumor sites in MRI scans through a partially Bayesian U-Net; and assessing precision health balance performance scores based on kinematic information obtained from non-invasive IMU via convolutional neural networks.

1.4 Dissertation Overview

This dissertation presents novel approaches to scalability and intuitive prior selection in Bayesian UQ of DNNs. It further develops these approaches along with domain-specific insights in a number of science and engineering applications. The organization of these developments follows below, as well as in Figure 1.1.

Chapter 2 provides background information and a discussion of current approaches to scalable Bayesian inference, namely the MCMC, variational inference, and Stein classes of algorithms.

Chapter 3 details the implementation of Bayesian dense neural networks via SVGD and pSVGD algorithms to create a novel rotorcraft ice detection framework based on data collected from high-fidelity physics simulations. This framework enables rotorcraft to monitor flight performance metrics impacted by ice accretion as well as model uncertainty in real time.

Chapter 4 takes a similar methodological approach to developing BNNs based on physics simulation data as chapter 3, but for the purpose of predicting poly-crystalline stress response to time-evolving forcing functions. SVGD and pSVGD algorithms are demonstrated here to state-of-the-art GCNN and GRU architectures.

Chapter 5 introduces two novel methodologies for scalability and prior selection, which will be implemented in Chapter 6 and Chapter 7 respectively. The first methodology is a novel heuristic for reducing the Bayesian dimensionality of a DNN via targeted inference guided by a gradient-based

sensitivity index. The second methodology is a novel framework for selecting abstract, high-dimensional Bayesian parameter priors that encode tangible, low-dimensional predictive priors.

Chapter 6 implements the novel, pBNN heuristic developed in Chapter 5 on the state-of-the-art U-Net architecture for the purpose of creating a BNN model to identify and segment the presence of tumors based on multi-modalities of MRI scans.

Chapter 7 implements the novel, intuitive prior selection approach also developed in Chapter 5 for the novel purpose of automatically assessing physical therapy patient balance performance based on kinematic data collected from a single IMU sensor.

Chapter 8 summarizes key advancements and research themes from the preceding chapters and includes recommendations for future work in these research areas.

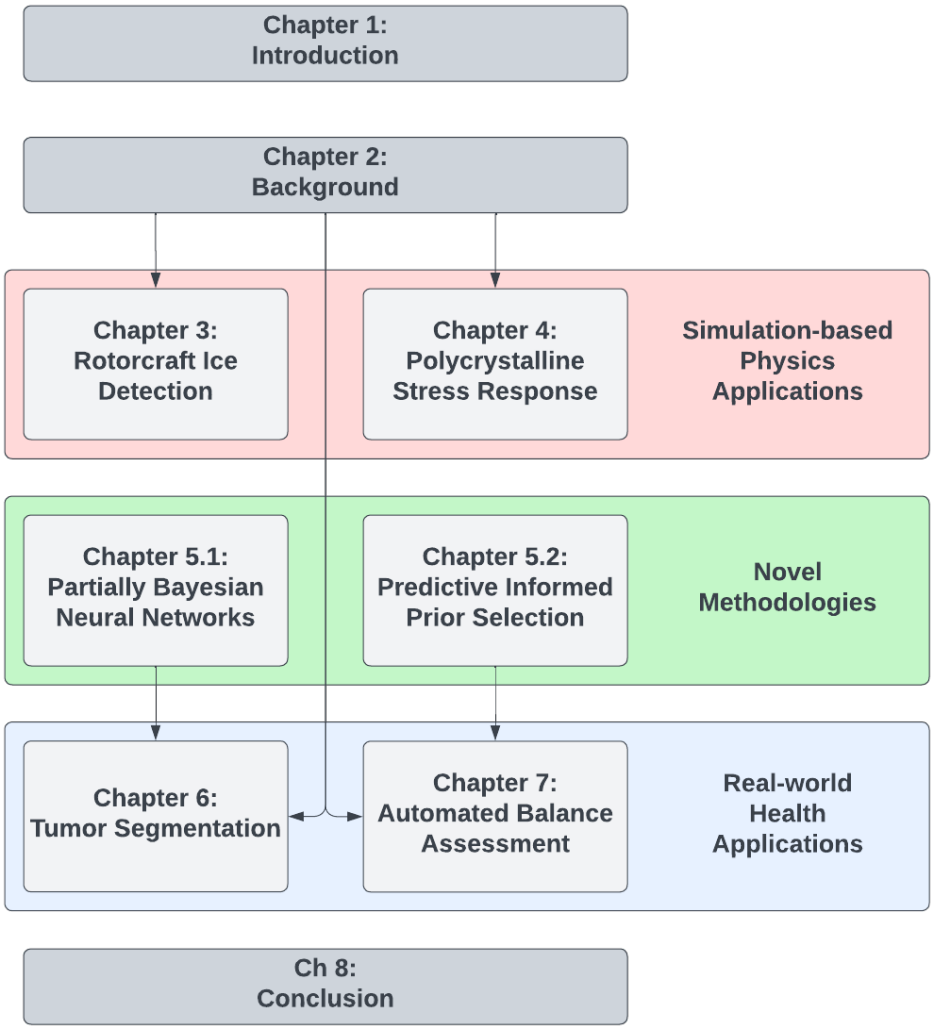


Figure 1.1: Organizational flowchart

CHAPTER 2

Background

This chapter presents a brief survey of Bayesian inference algorithms and assess their suitability for applications in Bayesian neural networks (BNNs). Notably, Stein variational gradient descent (SVGD) and projected Stein variational gradient descent (pSVGD) show particular promise for BNN implementation. This chapters also conducts several numerical experiments to demonstrate the performance of these two algorithms and validate their performance against a state-of-the-art Markov chain Monte Carlo (MCMC) algorithm: Hamiltonian Monte Carlo (HMC).

2.1 Introduction

Bayesian inference algorithms have developed substantially since the introduction of the Metropolis-Hastings method [24, 25]. However, the "curse of dimensionality" remains inevitable where even the most modern MCMC algorithms struggle [57]. Furthermore, the state-of-the-art neural network architectures used for scientific machine learning (SciML) are growing ever larger, with parameter spaces frequently surpassing millions of dimension. As such, the Bayesian inference algorithms that enable BNNs must be highly scalable.

The variational inference (VI) class of algorithms re-frames the sampling approach characteristic of rejection sampling, importance sampling, and MCMC as an optimization problem. Instead of trying to sample from the true target distribution (the true posterior in this case), an approximate posterior is sought. This approximate posterior takes the form of a parametrized family of distributions that is otherwise tractable and easy to sample. An optimization problem is then created where the variation parameters of the approximate distribution can be tuned such that the discrepancy between the approximate distribution and the true target distribution is minimized. Several algorithms under VI will be introduced below and adopted in this thesis work for building BNNs.

2.2 Background

2.2.1 Neural Networks

A neural network (NN) is a function that maps input x to output $\hat{y}$, that is–

$$\hat{y} = f(x) \tag{2.1}$$

where the hat in $\hat{y}$ denotes the NN prediction. The architecture of a given NN involves passing and processing information through successive layers of the network. For example, a densely connected feed-forward NN is organized into layers of different nodes, where each node performs calculations on output values from the previous layer before passing the result onto the next layer. Mathematically, the computation for node j in layer ℓ is

$$a_{\ell,j} = \sigma_\ell \left(\sum_i W_{\ell,ij} a_{\ell-1,i} + b_{\ell,j} \right) \tag{2.2}$$

where $a_{\ell,j}$ is the node output, σ_ℓ is an activation function (e.g., rectified linear units (ReLU), sigmoid function, hyperbolic tangent (tanh), etc.), $a_{\ell-1,i}$ is the ith nodal input (which is the output from the ith node in the previous layer), $W_{\ell,ij}$ is the weight placed on the ith nodal input, and $b_{\ell,j}$ is a biasing term. Definitions vary as to what qualifies a neural network to be "deep", though generally a DNN consists of two or more hidden layers (i.e., layers that are not the input or output layer).

Convolutional neural networks (CNNs) build upon this concept of dense neural networks, though rather than using matrix multiplication, cross-correlation[1] is used to produce outputs from input vectors and weight matrices. Mathematically, the computation for node j in layer ℓ is

$$a_{\ell,j} = \sigma_\ell \left(\sum_i W_{\ell,ij} * a_{\ell-1,i} + b_{\ell,j} \right), \tag{2.3}$$

where $*$ is the cross-correlation operation. CNNs excel at processing spatially or temporally correlated feature spaces, such as those found in time-series data, images, and video frames. 1D (for time-series) and 2D (for flat images) CNNs are most common, though higher order CNNs are possible.

[1]Technically, the operation being performed in CNNs is cross-correlation, which is very similar though not identical to convolution.

2.2.2 Bayesian Neural Networks

BNNs treat the parameters of a NN, w, as random variables with associated probability density functions (PDFs) representing the uncertainty or belief state on w. When training data become available, these PDFs are updated via Bayes' theorem:

$$p(w|X_T, y_T) = \frac{p(w)p(y_T|X_T, w)}{p(y_T|X_T)}, \tag{2.4}$$

where $p(w)$ is the prior PDF on the weight parameters and I also assumed $p(w|X_T) = p(w)$ (i.e., the prior uncertainty should not change from knowing only the input values of the training data), $p(y_T|X_T, w)$ is the likelihood function, $p(w|X_T, y_T)$ is the posterior PDF, and $p(y_T|X_T)$ is the Bayesian evidence. Solving the Bayesian inference problem then entails computing the posterior $p(w|X_T, y_T)$—that is, our updated uncertainty on w given the training dataset (X_T, y_T). For ease of notation, the training dataset will also be interchangeably denoted by $\mathcal{D} = (X_T, y_T)$.

2.2.3 Prior and Likelihood Functions

Prior distributions are a defining component of Bayesian analysis. Prior distributions are most typically used to encode an beliefs of a system, before these beliefs are influenced by data. In the absence of any tangible prior beliefs, many Bayesian statisticians might select priors that have little to no influence on the posterior distribution, also known as weak priors and uninformative priors, respectively [21]. Selecting priors for BNNs can be particularly challenging, as the parameter space can be very abstract and uninterpretable. Many BNN users simply resort to using weak Gaussian priors without consideration for other distributions [58].

The purpose of the Bayesian likelihood function is to assess the probability of observed data as a function of the parameters of a given model. The likelihood modifies the prior distribution in accordance with Bayes' theorem in order to arrive at the data-informed posterior distribution. The exact likelihood distribution used to infer parameter distributions for a given statistical model is often a matter of user selection. Typically for regression-type problems, a normal distribution is selected to model the discrepancy between the model predictions and observed data. This is analogous to mean squared error (MSE) in a frequentist setting.

Models demonstrated in this dissertation typically assume additive noise in the form:

$$y_T = f(w, X_T) + \sigma_\epsilon$$

For a regression-type problem, this implies a likelihood function based on the normal distribution

and the error between the model, f and the observed training data X_T and y_T:

$$\mathcal{N}(0, \sigma_\epsilon^2; y_T - f(w, X_T)).$$

Selection of the noise parameter, σ_ϵ, should be based on observed noise in the data. This value can be selected directly or can itself be learned through Bayesian or maximum likelihood methods as part of a hierarchical Bayesian formulation [59].

2.2.4 Hamiltonian Monte Carlo

HMC [60, 61] is a well-known and well-established MCMC sampling method [57]. HMC frames the sampling of the posterior in terms of Hamiltonian dynamics (HD) of the evolution of the weights coupled with periodic Metropolis-Hastings (MH) steps [57]. The HD allows the weights to transition away from the previous accepted state while staying in regions of high posterior probability due the phase volume-preserving aspects of HD. The usual MH acceptance criterion can then be applied to these decorrelated samples resulted from HD. The overall algorithm has some similarity to the thermostatted dynamics of single particle in a high dimensional space.

For HMC, the Hamiltonian is comprised of a potential and a kinetic energy

$$H = \Phi + \frac{1}{2}\mathsf{p} \cdot \mathsf{M}\mathsf{p}. \tag{2.5}$$

The posterior $p(\omega \mid \mathcal{D})$ is associated with the potential Φ

$$\Phi = -\log p(\omega \mid \mathcal{D}) \tag{2.6}$$

and the (fictitious) momentum p is combined with a selected mass matrix M to form a quadratic kinetic energy. Choice of form of kinetic energy is relatively free since the main requirement is that the probability distribution on phase space marginalizes to the target distribution. The choice of the mass matrix effectively transforms the target parameter space, by changing the constant H level sets and the correlation structure of the target distribution.

Computationally, the HD is implemented with a sympletic (reversible, phase volume preserving) integrator such as leapfrog Verlet

$$\begin{aligned}
\mathsf{p}_k((n+1/2)\Delta t) &= \mathsf{p}_k(n\Delta t) - \frac{1}{2}\Delta t \nabla \Phi(\omega_k) \\
\omega_k((n+1)\Delta t) &= \omega_k(n\Delta t) + \Delta t \mathsf{M}^{-1}\mathsf{p}_k((n+1/2)\Delta t) \\
\mathsf{p}_k((n+1)\Delta t) &= \mathsf{p}_k((n+1/2)\Delta t) - \frac{1}{2}\Delta t \nabla \Phi(\omega_k((n+1)\Delta t)
\end{aligned} \tag{2.7}$$

which requires the gradient of the log posterior with respect to the weights. The effect of the mass matrix M is evident from how it enters (2.7), i.e., by transforming p_k and the effect of $\nabla\Phi$ on ω_k. The pseudo-time step Δt is chosen to maintain dynamic stability while promoting efficient decorrelation of states. Here k indexes stages of the HMC algorithm and samples, while n indexes pseudo-time. The momentum provides a stochastic aspect, where the momentum is initialized with a blend of the momentum of the previous stage of HD and a random component:

$$\mathsf{p}_k(0) = \beta \mathsf{p}_{k-1} + (1-\beta)\boldsymbol{\varepsilon} \text{ with } \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathsf{M}) \tag{2.8}$$

where β is a hyperparameter and $\mathsf{p}_0 = \boldsymbol{\varepsilon}$ for the first stage.

The MH step is performed every N HD steps

$$\omega_{n+1}|\omega_n = \begin{cases} \omega_n(N\Delta t) & \text{with probability } \alpha \\ \omega_n(0) & \text{else} \end{cases} \tag{2.9}$$

where

$$\alpha = \min\left(1, \frac{\exp(-H(N\Delta t)}{\exp(-H(0))}\right). \tag{2.10}$$

The MH and the momentum initialization allow the overall chain to jump to new level sets of the Hamiltonian.

2.2.5 Variational Inference

Conventional Bayesian solutions largely rely on MCMC [25, 26, 27, 28] to *sample* the posterior distribution. While MCMC provably converges to the exact posterior that may be highly non-Gaussian, it converges slowly in high-dimensional settings due to the onset of measure concentration to the so-called *typical set* [32]. HMC [62, 31, 32], one of the more scalable MCMC variants, has been used for Bayesian inference for up to hundreds of parameters, but remains orders of magnitude short of the large neural networks (as many as millions of parameters) that are becoming common in SciML applications.

VI [33, 34, 35] provides much better scalability by finding the best approximation to the true posterior from a parametric family of distributions (e.g., all independent Gaussians), thereby turning the sampling task into an optimization one:

$$\theta^* = \arg\min_{\theta} \; D_{\mathrm{KL}}[q(w;\theta) \parallel p(w|x_T, y_T)], \tag{2.11}$$

where $q(w;\theta)$ is the approximating posterior PDF parameterized by θ, and the Kullback-Leibler

(KL) divergence D_{KL} quantifies the farness from the true posterior $p(w|x_T, y_T)$ to $q(w;\theta)$. One can further show that θ^* is also the maximizer of the well-known *evidence lower bound (ELBO)*:

$$\theta^* = \arg\max_{\theta} \ \mathbb{E}_q \left[\ln p(y_T|x_T, w) + \ln p(w) - \ln q(w;\theta)\right] = \arg\max_{\theta} \ \mathrm{ELBO}(\theta) \tag{2.12}$$

which can be estimated through Monte Carlo that samples $w^{(m)}$ from $q(w;\theta)$:

$$\mathrm{ELBO}(\theta) \approx \frac{1}{M}\sum_{m=1}^{M}\left[\log p\left(y_T|x_T, w^{(m)}\right) + \log p\left(w^{(m)}\right) - \log q\left(w^{(m)};\theta\right)\right]. \tag{2.13}$$

The optimization can be approached leveraging gradient-based algorithms, where gradient with respect to θ can be obtained via back-propagation. For example, one recent implementation, Flipout [63] VI has been demonstrated to provide substantial computational savings for dense, convolutional, and recurrent neural network architectures. The method injects pseudo-independent weight perturbations in order to decorrelate gradients, thereby achieving drastically decreased variance for the ELBO Monte Carlo estimator. It also offers vectorized implementation that allows one to take advantage of GPU computations.

However even with Flipout, the computational and memory requirements for million-parameters is still extremely high, if not outright prohibitive. [64] found that for large DNN architectures with $O(10^7 - 10^8)$ parameters, MFVI failed to converge altogether. Therefore, one cannot simply take Flipout off-the-shelf to build million-dimensional BNNs, additional algorithmic developments are still needed.

2.2.6 Stein variational gradient descent

A major shortcoming of VI is its ability to balance computational scalability with a flexibility that can capture both correlation effects and non-Gaussian structures. Mean-field VI (MFVI) is highly scalable but independence assumptions are questionable in neural networks where abstract parameters can be highly correlated with one another. Full covariance VI (FCVI) can capture correlation effects but with poor scalability. SVGD [36] addresses this tradeoff with a particle based parameterization approach that roughly follows the VI optimization framework. Particles drawn from an arbitrary starting distribution (though often the prior) are repeatedly transported via a form of functional gradient descent. This gradient descent approach implicitly minimizes the KL divergence between the set of particles and the true posterior by leveraging the connection between the derivative of the KL divergence with Stein's identity and making use of kernelized Stein discrepancy techniques. The result is a flexible framework that captures correlation effects as well as highly non-Gaussian structure. This approach is still somewhat limited by the curse of

dimensionality, and has a tendency to collapse to the MAP approximation as the dimensionality of the problem far exceeds the number of particles [37]. While the MAP approximation is still known to make for good predictive models (without regard to uncertainty), it is still a very low fidelity approximation to full Bayesian inference and may not provide adequate uncertainty quantification.

Key SVGD update steps are summarized below. For a set of m particles $\{w_i^{(n)}\}_{i=1}^m$ at iteration n that approximates $p(w|x, y)$ and positive definite kernel k, the particle update steps entail:

$$w^{(n+1)} \leftarrow w^{(n)} - \alpha\phi^*(w^{(n)})$$

$$\phi^*(w_i) = \sum_{j=1}^{m} k(w_i, w_j)\nabla_{w_j} \log p(w_j|X, y) + \nabla_{w_j} k(w_i, w_j). \tag{2.14}$$

For details of the algorithm derivation, I refer readers to [36].

2.2.7 Projected Stein Variational Gradient Descent

Projected stein variational gradient descent and Newton's method (pSVGD/pSVN) [65] attempt to address scalability issues in SVGD. Below I give a brief introduction of its main idea following [65]. The approach makes use of likelihood Hessian information (either via the full backpropagation second derivative or via the Gauss-Newton first derivative approximation) to identify the low dimensional active subspace parameters are most informed by the data. Specifically, we are interested in the basis functions $\psi_i, i = 1, 2, ...r$ that are the eigenvectors corresponding to the r largest eigenvalues of the following generalized eigendecomposition problem:

$$\mathbb{E}[\nabla_w^2 \eta_y(w)]\psi_i = \lambda_i \Gamma_0^{-1} \psi_i \tag{2.15}$$

where $\mathbb{E}[\nabla_w^2 \eta_y(w)]$ is the average likelihood Hessian and Γ_0 is the prior covariance matrix. The decomposition can be done with randomized numerical linear algebra [66] without explicitly forming thew average Hessian matrix, which otherwise might be computationally intractable. The r leading eigenvectors ψ_r can project the parameters into the low dimensional active subspace where the likelihood is most informative on the prior. Below a certain threshold of eigenvalue, the components corresponding to the inactive subspace can be assumed to be projections of the unmodified prior and can be frozen from their prior samples. The SVGD algorithm need only characterize the posterior distribution in its low dimensional subspace. We start with Bayes' theorem in the parameter space:

$$p(w|x, y) \propto p_\epsilon(y - f(w)|x, w)p(w|x). \tag{2.16}$$

We want to decompose our posterior samples into their active subspace and complimentary components

$$w = w_r + w_\perp. \tag{2.17}$$

First we decompose the prior

$$p_0(w) = p^r(w^r|x)p^\perp(w^\perp|w^r, x). \tag{2.18}$$

Meanwhile, in decomposing the likelihood, the complimentary components are, by design, are considered negligible

$$p_\epsilon(y - f(w)|x) \approx p_\epsilon(y - f(w_r)|x, w). \tag{2.19}$$

Bayes' Theorem with the decomposed prior then becomes

$$p^r(w|x, y) \propto p_\epsilon(y - f(w_r)|x)p_0^r(w^r|x)p_0^\perp(w^\perp|w^r, x). \tag{2.20}$$

By drawing samples from the prior, the complimentary components of the samples can be frozen and SVGD applied to the prior and likelihood only in the projected active subspace. The high dimensional posterior can be reconstructed by projecting the low dimensional posterior back up to its low rank approximation and recombining it with the frozen complimentary components of the prior also projected back into the high dimensional space, to get a full rank approximation of the posterior:

$$\nabla_{z_j} \log p(z_j|X, y) = \psi^r \nabla_{w_j} \log p(w_j|X, y). \tag{2.21}$$

pSVGD is particularly powerful when it can exploit intrinsic low-dimensional geometric structure of the posterior distribution. [65] demonstrated that this low dimensional structure is quite prominent in the context of discretized partial differential equations (PDEs). The same structure exists in the data independently of how the spatial or temporal domain is discretized. We expect similar low dimensional structure to be quite prominent in the context of CNNs for spatio-temporal data. That is to say that while the number of parameters of the neural network architecture grow with sampling the sampling frequency of the IMU data, the intrinsic dimensionality of the data (and thus the model) is much independent of the sampling frequency. Furthermore, the high degree of correlation and redundancy between neurons allows for further parameter dimensionality reduction, in much the same way as CNN compression via SVD. By decoupling the dimensionality of the data from the dimensionality of the neural network parameter space, substantial dimensionality reduction is possible. In the context of SVGD, this allows the ratio of particles to dimensions to be much high and in turn would allow for higher fidelity Bayesian inference for a given computational budget. Furthermore, the truncation of the likelihood Hessian to only the most data informed

components could serve as a form of model regularization, improving predictive performance by eliminating components of the parameters that are more likely influenced by data noise rather than signal.

2.3 Validation of SVGD and pSVGD

This section validates and demonstrates the effectiveness of the SVGD and pSVGD Bayesian inference algorithms against baseline methods on parameterized, data-driven models with a small synthetic dataset. These algorithms are first demonstrated on a linear-Gaussian model and compared against the analytical posterior. Next, the algorithms are demonstrated on a small, dense neural network modeling the same dataset. With no analytical posterior available, SVGD and pSVGD are compared against a state-of-the-art MCMC algorithm: HMC.

2.3.1 Data Generation

Synthetic data is generated from the following equations:

$$y_n \sim N(0, 1)$$

$$X_t = \sin\left(\frac{t y_t}{2\pi} - \epsilon_n\right) \tag{2.22}$$

$$y_t = \frac{2\pi \arcsin(X_t)}{T} + \epsilon_n \tag{2.23}$$

where X_t and y_t are sampled at discrete time steps t on the domain $[0, T = 10]$. This data generating function is selected such that the output space is one dimensional but the input feature space is dependent on sampling frequency. These temporally correlated features cater well to pSVGD in particular, as the data generating function is known to have an innate dimensionality independent of the sampling frequency.

2.3.2 Linear-Gaussian Model

The Gaussian linear model takes the form

$$y = \theta X + \epsilon \tag{2.24}$$

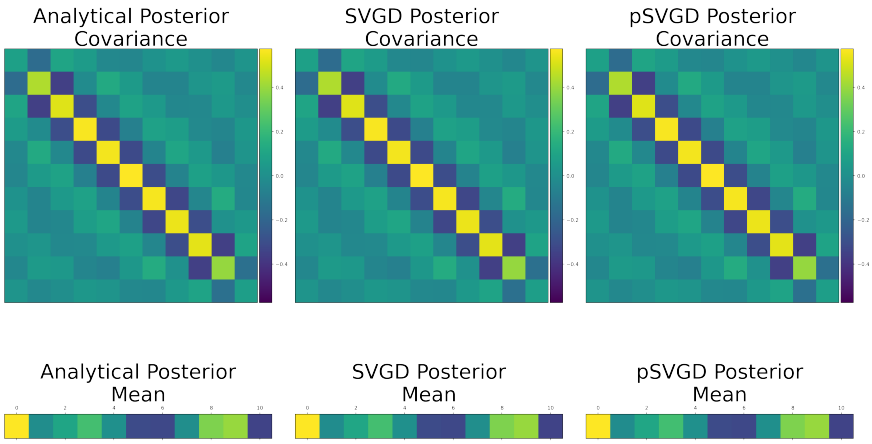


Figure 2.1: Visualization of an 10D linear Gaussian model posterior distribution, showing the mean vector and covariance matrix based on the analytical solution, SVGD numerical inference (500 particles), and pSVGD numerical inference (500 particles). SVGD and pSVGD converge very accurately to the analytical solution.

where θ are the Gaussian random variables that represent the model coefficients and ϵ is additive Gaussian noise. Prior and likelihood functions, shown below, are both independent Gaussian:

$$p_0(\theta) \sim N(0, 1^2) \tag{2.25}$$

$$p_\epsilon(X, y|\theta) \sim N(0, 0.25^2). \tag{2.26}$$

Due to conjugate prior, the Bayesian posterior of this linear-Gaussian model is analytically Gaussian:

$$\Sigma_N = [\Sigma_0^{-1} + X^T \Sigma_\epsilon X]^{-1}$$

$$\mu_N = \Sigma_N [X^T \Sigma_\epsilon^{-T} y + \Sigma_0^{-1} \mu_0].$$

With 500 particles, the analytical, SVGD, and pSVGD posteriors for the linear-Gaussian case are nearly indistinguishable in the plot shown in Figure 2.1. The plot in Figure 2.2 shows the convergence of the SVGD particles toward the true 11-dimensional Gaussian posterior as the number of particles is increased. Convergence roughly follows a linear trend on the log-log scale, consistent with Monte Carlo methods. Figure 2.3 shows the convergence of pSVGD to the true 11-dimensional Gaussian posterior as the projected dimensionality is varied. Figure 2.3 also shows the effect of the pSVGD truncation value on the KL divergence from the analytical posterior. The trend closely follows that of the eigenspectrum, giving some intuition that the eigenvalues of the active subspace can be used to estimate pSVGD approximation error.

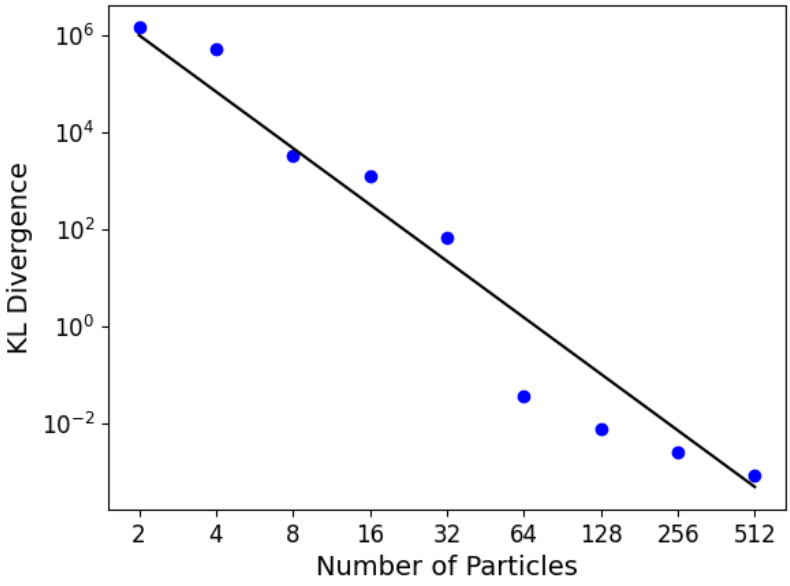


Figure 2.2: KL divergence between the SVGD particles and the true posterior as a function of the number of particles, line of best fit on the log-log scale shown in black.

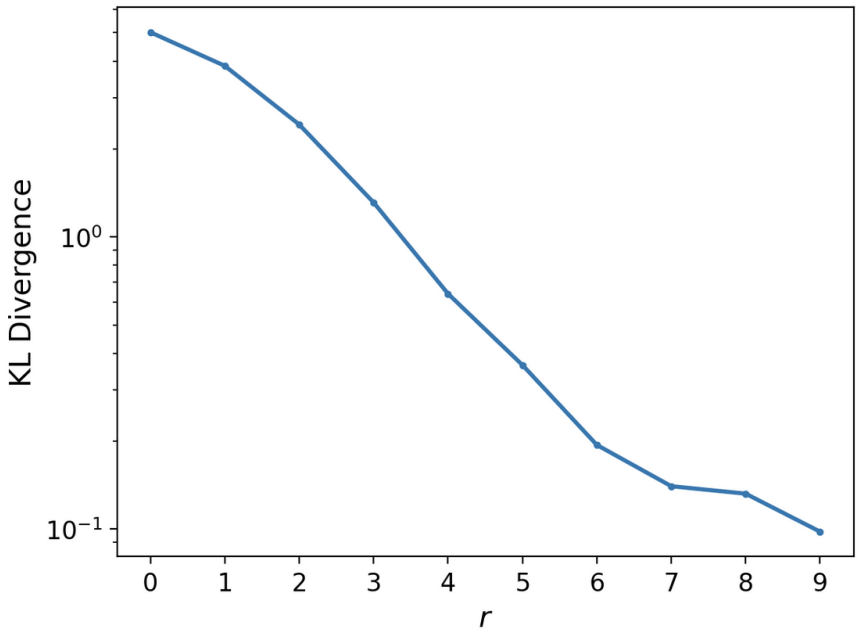


Figure 2.3: KL divergence between pSVGD particles and the true posterior as a function of projected dimensionality, r, with the number of particles fixed at 128. $r = 0$ is equivalent to using the prior to approximate the posterior, while $r = 10$ (no truncation) is equivalent to vanilla SVGD.

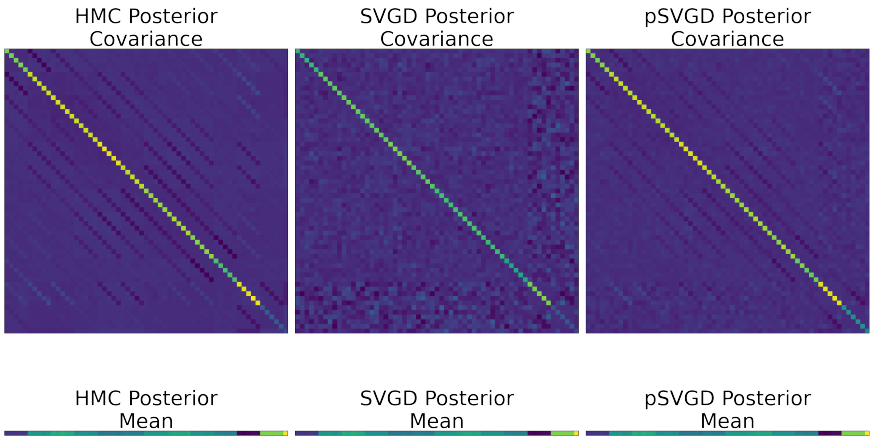


Figure 2.4: Visualization of a nonlinear neural network model posterior distribution with 36 trainable parameters, showing the mean vector and covariance matrix based on Hamiltonian Monte Carlo numerical inference (1M samples), SVGD numerical inference (500 particles), and pSVGD numerical inference (500 particles).

2.3.3 Dense Neural Network Model

In the second test case, a dense neural network (multi-layer perceptron) take the following form:

$$
\begin{aligned}
a_1 &= \sigma(W_1 X + b_1) \\
a_2 &= \sigma(W_2 a_1 + b_2) \\
\hat{y} &= a_2.
\end{aligned}
$$

Prior and likelihood functions are the same as the linear Gaussian model in Equation (2.25) and Equation (2.26).

In general, there is no analytical, closed-form posterior available for a dense neural network. Instead, we show the posterior mean and covariance calculated from $1,000,000$ samples from HMC [61]. HMC becomes intractable in very high dimensions but in a smaller dense neural network allows us to sample very precisely from the posterior distribution. Figure 2.4 illustrates that, compared to the HMC samples, SVGD begins to show noticeable collapse of the posterior. Particles are still clustered around the HMC posterior mean, but with a substantially narrower covariance. pSVGD shows convergence of the mean and covariance diagonal, with some minor deviation on the covariance off-diagonal elements.

2.4 Conclusion

This chapter develops the background algorithms and approaches necessary for understanding the novel contributions later in this work. Neural networks and their Bayesian extensions are outlined in Section 2.2.1 and Section 2.2.2. Specifics on Bayesian prior and likelihood functions are outlined in Section 2.2.3. Lastly, a selection of current numerical Bayesian inference algorithms are presented in Sections 2.2.4 to 2.2.7 along with discussion of their various strengths, weaknesses, and suitability for enabling BNN. While HMC is one of the most advanced MCMC-type algorithms, its scalability is unfortunately very limited. Variational inference demonstrates excellent scalability and to date has been utilized in many BNN implementations, however variational posteriors and mean-field approximations risk substantial underestimation of posterior variance and may lack flexibility in BNN. Nevertheless, its scalability is unmatched and this algorithm is used to enable a large scale BNN implementation in conjunction with my own novel methodological contribution, pBNN, in Chapter 6. The last two algorithms, SVGD and pSVGD, show promise for their scalability and flexibility in BNN, however their adoption in the literature has been limited. Section 2.3 presents my own validation of the SVGD and pSVGD algorithms against analytical solutions (linear Gaussian case) and HMC solutions (small dense NN case). The results shown in this chapter demonstrate good conformity between SVGD and pSVGD and analytical / HMC baselines, which is encouraging for use in the larger applications developed later in this dissertation. Chapter 3 and Chapter 4 go on to make extensive use of the two Stein algorithms, and Chapter 7 makes use of these Stein algorithms in conjunction with my own methodological contributions.

CHAPTER 3

Bayesian Neural Networks for Rotorcraft In-flight Ice Detection via Computational Aeroacoustics

This work develops a novel ice detection framework specifically suitable for rotorcraft using computational aeroacoustics and Bayesian neural networks (BNNs). In an offline phase of the work, the acoustic signature of glaze and rime ice shapes on an oscillating wing are computed. In addition, the aerodynamic performance indicators corresponding to the ice shapes are also monitored. These performance indicators include the lift, drag, and moment coefficients. A BNN is subsequently trained using projected Stein variational gradient descent to create a mapping from the acoustic signature generated by the iced wings to predict their performance indicators along with quantified uncertainty that is highly important for time- and safety-critical decision-making scenarios. While the training is carried out fully offline, usage of the BNN to make predictions can be conducted rapidly online allowing for an ice detection system that can be used in real-time and in-flight. The work presented in this chapter is the result of close collaboration with Myles Morelli, Alberto Guardone, Beckett Zhou, and Xun Huan and is published in *Structural and Multidisciplinary Optimization* [67]. While I preserve the majority of the paper to convey its comprehensive narrative, we acknowledge that in the work presented in this chapter, Morelli and Guardone were chiefly responsible for the icing modeling, and Zhou for computational fluid dynamics and aeroacoustics simulations.

3.1 Introduction

Rotorcraft possess the capability to vertically take-off and land, allowing them to operate in challenging flight conditions where more conventional fixed-wing aircraft cannot. Their operational environment lends itself to low-level altitudes, and many missions are often towards the boundary of the flight envelope. Throughout the course of aviation, the requirement for flight in poor weather conditions and at night has always been a driver for innovation and rotorcraft are no exception.

In the past, rotorcraft were fair weather vehicles with marginal performance, however, now they regularly operate in conditions from hot and dry to cold, wet and windy [68]. Cold climates can potentially leave aircraft susceptible to dangerous icing conditions. In-flight icing encounters can jeopardise the performance and handling qualities of aircraft and hence pose a serious threat to flight safety [69, 70]. This threat to flight safety arises as ice accretion can rapidly alter aerodynamic lifting surfaces such as wings and rotors during flight which are highly sensitive to geometric modifications. Naturally, establishing a comprehensive understanding of the fundamental physics of aircraft icing is paramount to ensure the reduction of ice related accidents and the progression of safety critical technology.

In 2002, Bragg et al. [71] critically assessed aircraft safety during icing conditions and suggested the introduction of smart icing systems to reduce aircraft accident rates. Their work introduced the concept that a smart icing system is potentially a better way to manage the ice protection system. Their work recognised that warning the pilot of ice conditions through the use of ice detection systems could significantly increase the safety of aircraft to either optimize the use of ice protection systems or to avoid icing conditions altogether. Over the past two decades, this has led to sustained technological advancements in ice detection systems.

The methods used for the detection of ice on aircraft can be characterised into either direct or indirect approaches. Methods that use different kinds of sensors to detect ice accretion on the airframe or atmospheric conditions symptomatic of icing environments can be characterised by the direct approach. Methods that rely on a change in response to the aircraft state due to icing can be characterised by the indirect approach.

Direct methods of sensing ice have received the most scientific interest and promising methods involve measuring changes in surface properties due to ice accretion. Sensing techniques may involve the reaction of a signal or electric field to ice. Promising methods will now go on to be discussed. Direct detection methods which involve the reaction to a signal include surface generated wave, infrared thermography, optical reflectometry and ultrasonics wave techniques. Varadan et al. [72] discussed using surface generated Love waves as a mechanism for ice detection. The method involves using piezoelectric devices for producing Love waves. The waves are then propagated through the surface, and as ice accretes, the amplitude and velocity of the waves are altered, which can then be measured. Hongerholt and Rose [42] demonstrated the use of ultrasonic guided waves for ice detection on the surface of a wing. In principle, ultrasonic guided waves can be propagated and reflected on the surface of a wing and the attenuated amplitude of the reflected waves in different mediums can be measured. Zhuge et al. [40] discussed a method of infrared thermography technique for ice detection. Infrared thermography techniques emit electromagnetic signals which are reflected by the surface and once received back the signals are analysed for temperature variations. Bassey and Simpson [41] introduced an advanced method of optical time

domain reflectometry for ice detection. Optical time-domain reflectometry is used to measure the dielectric permittivity and volumetric water content on the aircraft surface.

Direct detection methods which involve the reaction of an electric field include capacitive and impedance techniques. Schlegl et al. [73] discussed using capacitive techniques for sensing ice. Capacitive techniques monitor the presence of dielectric substances on the surface which change the capacitance. The working principle relies on the different electrical properties of air, water, and ice. Roy et al. [74] introduced impedance techniques for ice detection. In principle, the impedance technique generates an excitation signal which is delivered to electrodes embedded in the surface. When ice accretes on the electrodes the impedance of the electrodes is affected and by measuring the changes in voltage ice can be detected.

In the main, ice detection techniques which can be characterised into the direct sub-category allow for precise information regarding the ice location, thickness, and location. However, a major drawback of these methods is that most are either commercially unproven due to preliminary testing not exceeding the laboratory environment, or are unpractical for moving components, making their implementation on rotorcraft challenging.

Indirect ice detection systems have received renewed interest of late. Irrespective of the indirect method, all utilize previously established aircraft states to compare against that of the iced state. The difference between the clean and iced aircraft states then acts as an indicator for icing. Melody et al. [75] discussed using aircraft dynamics as an indicator for in-flight icing. In principle, using the aircraft dynamics as an indicator for icing requires identification algorithms to identify aircraft performance, stability, and control parameters over time, based on measurements of the aircraft state variables and control input. Deiler and Fezans [76] introduced a performance-based ice detection methodology. Performance-based identification uses the change in the steady flight state meaning it can be used prior to reaching dynamic stall conditions and during steady flight conditions contrary to the aircraft dynamics approach. A major limitation of using either the dynamics or performance of aircraft for ice detection is that the type, location, and thickness of ice cannot be determined. Additionally, the detection of ice prior to dynamic stall is fundamental to aircraft safety.

Indirect ice detection methods which involve the use of acoustic noise as an indicator for ice accretion are particularly relevant for rotorcraft. A characteristic and largely undesirable feature of rotorcraft is their rotor noise. Rotor noise sources are complex and are a combination of loading noise, thickness noise, shock noise, blade-vortex interaction noise, and broadband noise. However, Cheng et al. [77] recognised the strong correlation between the ice-induced surface roughness and the broadband noise level suggesting rotor acoustics could be a viable technology for rotorcraft ice detection. Chen et al. [78] further investigated using numerical techniques and an artificial iced notch and concurred that the acoustics could be potentially used to indicate the formation of ice on a rotor. Ice detection via means of aircraft acoustics however remains in the early stages of

research. In the main, a major critique of the literature so far is that the detection of natural ice shapes is yet to be investigated. The experimental work from Cheng et al. [77] used sandpaper to represent ice roughness, while the numerical work from Chen et al. [78] used an artificial ice notch based on the experimental iced surface roughness results of a straight wing from the University of Illinois, presented by Broeren and Bragg [79]. The first characterisation between glaze and rime iced noise signals was established by Morelli et al. [80]. Their work utilized numerical techniques to simulate the accretion of ice on rotors and subsequently the rotor noise signature. The results highlighted that glaze ice accretion produces noise at a significantly higher frequency caused by ice induced flow separation. neural networks (NNs) based icing identification has been shown to be a powerful technique for in-flight ice detection [43]. Meanwhile optimization techniques have also been used to improve the efficiency of ice protection systems [81]. This work seeks to address the issue of very few ice detection systems being designed specifically for rotorcraft by developing a technique which is based on the rotor noise and NNs. While several studies mentioned in the literature have investigated using noise as an method for ice detection, this is the first work to establish an approach for developing a system that predicts the aerodynamic performance directly from observed acoustics. In other words, this work seeks to not simply state whether icing is present or not, but also predict the severity and consequence of the icing condition through the aerodynamic performance metrics directly useful for aviation and piloting.

The high rotational speed of rotor blades makes direct ice detection techniques such as infrared techniques challenging. As a result, indirect ice detection techniques are of particular interest to rotorcraft. To achieve this goal of developing an acoustic ice detection system for rotorcraft, this work utilizes numerical techniques to simulate a large dataset of ice shapes. Subsequently, high-fidelity scale resolving methods are used for the simulation of the flow-field over the ice dataset. Additionally, this includes monitoring the aerodynamic performance characteristics of the different ice shapes. The surface pressure fluctuations are then taken from the aerodynamic data and used by an aeroacoustic solver to predict the far-field noise levels. Finally, a key novelty of this chapter is to train a ML model to directly map the acoustic noise signal produced from the ice structures to the performance indicators of the aircraft. In particular, NNs are selected as the ML model for their expressiveness in capturing complex mappings in a data-driven manner, and more specifically BNNs that can also provide uncertainty quantification (UQ) in the ML model and in their predictions. UQ for NNs is a challenging task due to its high-dimensional nature and often neglected in data-driven modeling. However it is crucial for mission- and safety-critical settings such as in predicting flight performance under dangerous icing situations—the *quality* of a prediction must be understood along with the predicted value. A further contribution of this work is then to enable UQ for ML leveraging recently developed computational algorithm of projected Stein variational gradient descent (pSVGD) [82]. An overview of our framework is shown in

Figure 3.1.

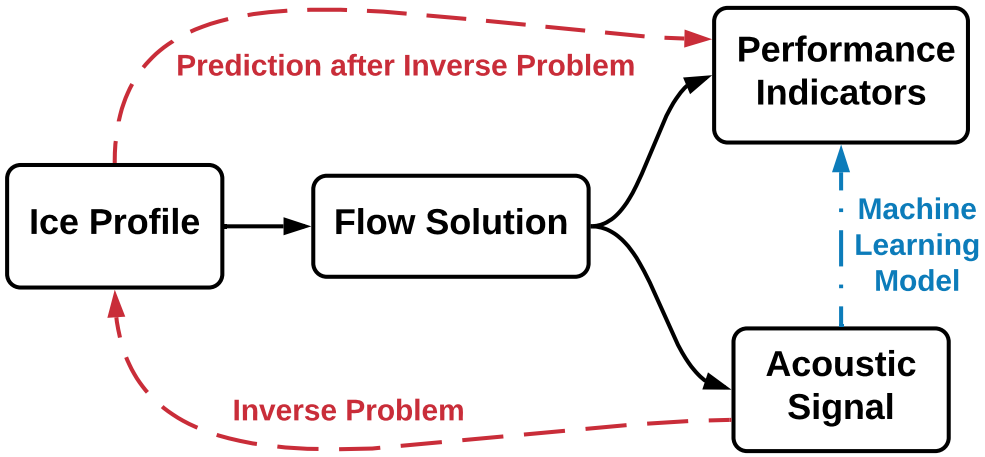


Figure 3.1: Workflow routes illustrating the physics causation (black solid), traditional inverse problem route (dashed red), and proposed machine learning model "shortcut" (dotted-dash blue).

Several aspects of this complex problem do however need to be acknowledged. While this methodology is able to predict the tonal and broadband noise components of iced airfoils. In reality the noise of an operating helicopter is highly complex with many different components including engine noise and tail rotor noise which would likely pollute the small differences in noise observed by an iced rotor. Nevertheless, this is the first step as to understanding if an acoustic ice detection system using NNs is feasible. To that end, if successful the installation on helicopter in operation would be possible due to the system being non-intrusive.

In summary, this work develops a framework for rotorcraft icing detection highlighted by the following key novelty and contributions:

- uses in-flight *acoustic* measurements to enable prediction of aerodynamic performance for rotorcraft, who cannot use visual/image observations due to the high blade RPM during operation;
- enables *real-time* performance predictions with *quantified uncertainty* through advanced ML models of BNNs, using a recently developed pSVGD algorithm;
- uses simulation data that are generated from state-of-the-art *physics-based models* rather than empirical models; and
- requires a truly multidisciplinary approach to combine ice formation and growth, turbulent flow aerodynamics, aeroacoustics, and ML with UQ.

The structure of the remaining chapter is as follows: Section 5.1 introduces the methodology of the acoustic ice detection and the machine learning using BNNs; Section 4.3 discusses the results of

the ice prediction, the turbulent flow simulations and acoustic prediction, as well as the training of the BNN and the design of the ice detection system; Section 3.4 highlights the concluding remarks of the work.

3.2 Methodology

The development of a technique for real-time, in-flight aeroacoustic ice detection system requires a highly inter-disciplinary approach. The methodology used for the prediction of in-flight ice accretion, the simulation of the turbulent iced flow-field and noise prediction, and finally the machine learning are hereinafter introduced.

3.2.1 Icing Simulation

The PoliMIce ice prediction toolkit is utilized to simulate a large sample of ice shapes required for the Neural Network. The PoliMIce ice prediction tool is described in detail in Ref. [83] and will subsequently be summarised. The first stage of the icing framework is to simulate the aerodynamic flowfield around the region of interest. In this work the SU2 solver is used to compute the aerodynamic flow field over the wing. The second stage is to simulate the trajectories of super-cooled water droplets within the flow domain and determine the collection efficiency and impingement locations. An in-house Lagrangian particle tracking code is utilized due to droplet-wall interaction behaviour being of importance to model physics such as droplet splashing on impact. Finally, the PoliMIce icing solver is utilized to predict the rate of ice accretion. The PoliMIce solver provides state-of-the-art ice prediction models. In particular, the model used in this work is the local exact solution of the unsteady Stefan problem for the temperature profiles within the ice layer in glaze conditions [84]. This helps enable the prediction of the ice horns present during glaze icing conditions. An assessment of the ice prediction capabilities has been demonstrated at the 1st AIAA Ice Prediction Workshop. In addition, the PoliMIce toolkit has exhibited its potential for simulating rotorcraft icing [85].

3.2.2 Turbulent Flow Simulation and Noise Prediction

SU2 has been developed with the task of solving partial differential equations (PDE) and constrained optimization problems on general unstructured meshes. the core of the suite is a Reynolds-averaged Navier–Stokes (RANS) solver capable of simulating the compressible, turbulent flows that are representative of many problems in aerospace and mechanical engineering. The finite volume method (FVM) is applied on arbitrary unstructured meshes using a standard edge-based data structure on a dual grid with control volumes constructed using a median-dual, vertex-based

scheme. Several numerical fluxes like Jameson-Schimdt-Turkel (JST), Upwind Roe, and variants of the AUSM schemes are implemented, and slope limiters enable second-order space integration. For time discretization SU2 uses a second-order dual time-stepping method with an outer time loop to march through the physical time, and of an inner loop which is usually a pseudo-time iteration or a (quasi-)Newton scheme. An Arbitrary Lagrangian-Eulerian (ALE) formulation of the RANS equations is implemented in SU2 to account for unsteady grid motion, in which the convective fluxes are adjusted to obtain solutions on arbitrarily moving grids.

In the SU2 suite, of particular relevance to this work are the scale-resolving and acoustic prediction capabilities. For scale-resolving simulation, the Enhanced Delayed Detached Eddy Simulation (EDDES) based on the Spalart-Allmaras (SA) turbulence model with a shear-layer adapted (SLA) sub-grid scale model [86] was used. In addition, to limit the numerical dissipation in the LES part of the EDDES solver, the inviscid flux is computed using the Simple Low Dissipation Advection Upstream (SLAU2) scheme [87]. The EDDES-SA solver has been demonstrated to successfully predict separated flows [88, 89], especially those induced by icing on the wing leading edge [90]

For aeroacoustic prediction, a solid-surface Ffowcs Williams and Hawkings (FWH) analogy is used. To that end, Farassat's Formulation-1A (F1A) [91] has been implemented in the SU2 suite [92, 93] which computes far-field noise from surface pressure fluctuations extracted from a preceding unsteady EDDES-SA simulation.

3.2.3 Machine Learning Using Bayesian Neural Networks

This section first introduces the classical NN, followed by the BNN that captures the uncertainty of NN model parameters, and finally the projected Stein variational gradient descent (pSVGD) algorithm that numerically constructs a BNN.

3.2.3.1 Neural Networks

A NN is a function f mapping input feature x to output target variable y. The notation $\hat{y} = f(x)$ is used, where the hat denotes the *predicted* value from the NN. For illustration, I focus solely on densely connected feedforward NNs (also known as multi-layer perceptrons), though more advanced NN architectures (e.g., convolutional and recurrent NNs) may also be adopted within this overall ML framework. The hyperparameters that define the NN architecture (e.g. number of layers, nodes per layer, activation functions, etc.) are selected prior to NN training, and good hyperparameter choices can be sought systematically through validation or cross-validation. Once the hyperparameters are chosen, the NN training then focuses on determining the weight and bias parameters of all NN layers. I denote the collection of all such trainable parameters by ω. Then,

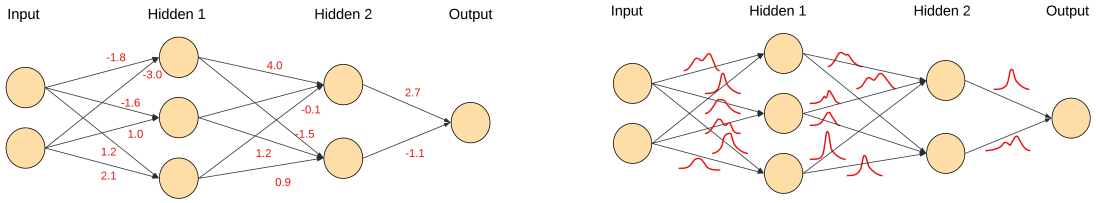
more explicitly, the prediction under a particular NN parameter setting is written as $\hat{y} = f(x;\omega)$

Given N training points in the form of input-output pairs $(x_T, y_T) = \{x_n, y_n\}_{n=1}^{N}$, NN training seeks ω^* that minimizes a loss function reflecting the degree of mismatch between NN predictions and the true target values for these training points. In regression problems, for example, the mean squared error (MSE) loss is often used:

$$\omega^* = \arg\min_{\omega} \left\{ \frac{1}{N} \sum_{n=1}^{N} [f(x_n;\omega) - y_n]^2 \right\}. \tag{3.1}$$

This optimization problem is typically solved via gradient-based algorithms such as stochastic gradient descent [94, 95] or ADAM [96]. In any case, the solution to ω^* is single-valued, and any new prediction made by the trained NN at a new input $\hat{y}_{new} = f(x_{new};\omega^*)$ would also be single-valued. No uncertainty information is produced to convey the confidence or quality of these estimates that would be affected by, for example, how many points were used to train the NN and how noisy those training points were. Having uncertainty quantification for the model and its predictions will be particular important to support safety- and mission-critical decision-making, such as in icing detection.

3.2.3.2 Bayesian Neural Networks



(a) Neural network

(b) Bayesian neural network

Figure 3.2: Illustration of a simple NN and BNN, each with an input layer, two hidden layers, and an output layer. Trainable parameters are depicted in red as deterministic values in the NN, and as probability distributions in the BNN.

I adopt a probabilistic approach that seeks to produce not just a single-valued prediction, but a *distribution* of plausible prediction values. This distinction is demonstrated in Figure 3.2. In particular, a Bayesian approach is taken, which treats ω as a random variable with an associated probability density function (PDF) that reflects the uncertainty in its values. When new training

data become available, the PDF and therefore uncertainty of ω is updated via Bayes' rule:

$$p(\omega \mid x_T, y_T) = \frac{p(x_T, y_T \mid \omega)p(\omega)}{p(x_T, y_T)} = \frac{p(y_T \mid x_T, \omega)p(\omega)}{p(y_T \mid x_T)} \tag{3.2}$$

where $p(\omega \mid x_T, y_T)$ is the posterior PDF; $p(\omega)$ is the prior PDF on the weight parameters; $p(y_T \mid x_T, \omega)$ is the likelihood PDF; $p(y_T \mid x_T)$ is a normalizing term independent of ω often known as the model evidence or marginal likelihood. Solving the Bayesian inference problem then constitutes computing or characterizing the posterior $p(\omega \mid x_T, y_T)$ for a given training dataset (x_T, y_T). Once the posterior is available, uncertainty can be propagated to predictions via Monte Carlo sampling $\omega^{(i)} \sim p(\omega \mid x_T, y_T)$, and correspondingly produce samples of $\hat{y}^{(i)}_{new} = f(x_{new}; \omega^{(i)})$ that represent the posterior-predictive uncertainty—that is, a distribution of plausible prediction values—at any new input x_{new}.

Solving for the posterior is extremely difficult for NNs, since ω can easily reach thousands, even millions of dimension for large NNs intended to model complex scientific and engineering processes. While Markov chain Monte Carlo algorithms [97, 27, 26, 98] are typically used for Bayesian computations for their ability to generate samples from the exact posterior, these methods do not scale well to such high dimensions in practice. In contrast, variational inference (VI) [35, 99] formulates an optimization problem that seeks the best approximation to the posterior from within a class of parameterized distributions in lieu of sampling the posterior. If $q(\omega; \theta)$ is used to denote the approximate posterior with θ representing the parameterization of these approximating distributions (e.g., if the q is from the family of all Gaussian distributions, then θ is the mean μ and covariance Σ of a Gaussian), the VI formulation then seeks θ^* that minimizes the Kullback-Leibler divergence (a measure of farness between two distributions) from the true posterior to the approximate posterior:

$$\theta^* = \arg\min_{\theta} D_{\mathrm{KL}} \left[q(\omega; \theta) \parallel p(\omega \mid x_T, y_T) \right]. \tag{3.3}$$

Since an optimization task is only concerned with find the single optimum point, it is more scalable to higher dimensions than a sampling task that needs to portray the entire landscape of the posterior distribution. Hence, by converting a sampling task to an optimization one, VI effectively trades off accuracy of posterior representation for scalability, making it more suitable for BNNs.

3.2.3.3 Projected Stein Variational Gradient Descent

The work in this chapter focuses on two particle-based VI methods: Stein variational gradient descent (SVGD) [36] and projected Stein variational gradient descent (pSVGD) [82]. With particles representations, these methods are able to capture correlation and non-Gaussian structures of distributions that the commonly used mean-field Gaussian VI cannot.

SVGD is derived by leveraging the relationship between the (functional) gradient of objective in Eqn. (3.3) to the Stein discrepancy, the latter which can be approximated using a set of particles (see [36] for a detailed derivation). A gradient-descent procedure can then formed to iteratively minimize this gradient, as summarized by

$$\omega_i^{\ell+1} \leftarrow \omega_i^{\ell} + \epsilon_\ell \hat{\phi}^*(\omega_i^{\ell}) \tag{3.4}$$

where $\omega_i^{\ell}, i = 1, \ldots, m$ is the location of the ith particle at optimization iteration ℓ, ϵ_ℓ is the learning rate that can also be adapted with ℓ, and

$$\hat{\phi}^*(\omega) = \frac{1}{M} \sum_{m=1}^{M} \left[k(\omega_m^{\ell}, \omega) \nabla_{\omega_m^{\ell}} \log p(\omega_m^{\ell} \mid x_T, y_T) + \nabla_{\omega_m^{\ell}} k(\omega_m^{\ell}, \omega) \right] \tag{3.5}$$

with $k(\cdot, \cdot)$ being a positive definite kernel. Furthermore, the gradient of the true log-posterior in the above equation can be evaluated via the sum of gradients of log-likelihood and log-prior, since the gradient of the log-model-evidence with respect to ω is zero. The overall effect is an iterative transport of a set of particles to best match the target posterior distribution $p(\omega \mid x_T, y_T)$. One observed drawback to SVGD, however, is the tendency for particles to collapse towards the mode of the distribution as the dimensionality becomes high relative to the number of SVGD particles, thereby (potentially drastically) under-representing the uncertainty [38].

pSVGD [82] aims to address this issue by projecting parameters into a low-dimensional subspace, and conducting the particle-based inference in this subspace. This allows inference to take place with a much higher particle-to-dimension ratio, which can mitigate the posterior collapsing phenomenon from SVGD. The specific space selected for projection is a likelihood-informed subspace (LIS). Components with the greatest likelihood-to-prior ratio are identified for inference while components with a negligible likelihood-to-prior ratio are assumed to be unmodified from the prior and need not be inferred directly. In other words, work will focus on only the subspace where the data can make a significant update from prior to posterior. I refer readers to [82] for a detailed derivation, and present a brief description below.

The LIS is determined by first calculating the average (log) likelihood Hessian $-\mathbb{E}_\omega \left[\nabla_\omega^2 \mathcal{L} \right]$ where $\mathcal{L} = \log p(x_T, y_T \mid \omega)$. $-\nabla_\omega^2 \mathcal{L}$ can be calculated either directly via back-propagation of the NN to evaluate the second derivative, or much more inexpensively via the Gauss-Newton approximation $-\nabla_\omega^2 \mathcal{L} \approx -(\nabla_\omega \mathcal{L} \nabla_\omega \mathcal{L}^T)$ needing only first derivative information. Next, the following generalized eigendecomposition problem can be solved:

$$-\mathbb{E}_\omega \left[\nabla_\omega^2 \mathcal{L} \right] \psi_i = \lambda_i \Gamma^{-1} \psi_i, \quad i = 1, ..., r \tag{3.6}$$

where λ_i are the r leading eigenvalues that represent the relative influence of the likelihood over the prior, and ψ_i are the corresponding eigenvectors. Therefore, eigenvalues below a certain threshold correspond to components with negligible influence from the prior, leaving the prior effectively unchanged. These negligible eigen-modes can then be truncated, and the LIS is constructed with the surviving eigenvectors corresponding to non-negligible eigenvalues. The choice of threshold value for truncation is application specific. For lower dimensional, mostly linear models, a threshold value $O(1)$ is sufficient. For larger, more complex models, a lower threshold value may be necessary. The original authors of pSVGD [82] use a value of 0.01.

With the LIS found, the prior parameters ω_0 can be decomposed into the projected component ω_0^r and the its compliment ω_0^T (subscript 0 indicates that the decomposition is done under the prior distribution). The compliment ω_0^T is then frozen at the prior projection. The projected component of the parameters will be inferred under the low-dimensional projected variable χ using the same iterative procedure in SVGD, where $\omega^r = \Psi^r \chi$. The original full parameter variable after pSVGD inference can then be reconstructed at the end:

$$\omega^\ell = \omega_0^T + \Psi^r \chi^\ell. \tag{3.7}$$

Performing Bayesian inference on χ instead of ω permits a much lower-dimensional problem. There is also secondary benefit that by truncating to the r components most informed by data, the compliment components that would otherwise be more amenable to fitting to data noise are now regularized, further improving predictive performance.

3.3 Results

3.3.1 Ice Prediction

The PoliMIce [83] tool kit is utilized for generating a large dataset of ice shapes which can subsequently be harnessed by the Neural Network. The icing conditions selected are based on the experimental work from Shin and Bond [100]. They conducted icing experiments in the Icing Research Tunnel (IRT) at the NASA Lewis Research Center. Their work studied the repeatability of ice shapes on a model wing at a range of temperatures varying from 247K to 271K. This work presents the numerical predictions of four ice shapes at different temperatures during the high-speed conditions for icing validation. The icing conditions selected are summarised in Table 3.1, while the ice predictions are presented in Fig. 3.3. The model used in the icing experiments was a straight wing with NACA0012 profile. The ice accretion simulations are consequently assumed to be two-dimensional due to the wing model being straight. The wing is set at an angle-of-attack

of 4°. The duration which the wing is exposed to icing conditions last for 420s. The wind tunnel airspeed was approximately $M = 0.3$. As previously mentioned, a range of different temperatures are investigated. The Liquid Water Content (LWC) and Mean Volume Diameter (MVD) of the super-cooled water droplets is respectively 0.55 g/m^3 and 20 μm. The numerical prediction shown in Fig. 3.3 are in relatively good agreement with the experimental measurements. The ice predictions at the lower temperature conditions displayed in Figs 3.3a & 3.3b closely resemble the ice measurements. There are slightly greater discrepancies when moving to the higher temperatures as shown in Figs 3.3c & 3.3d. The numerical predictions tend to predict the presence of glaze ice earlier than the experiments.

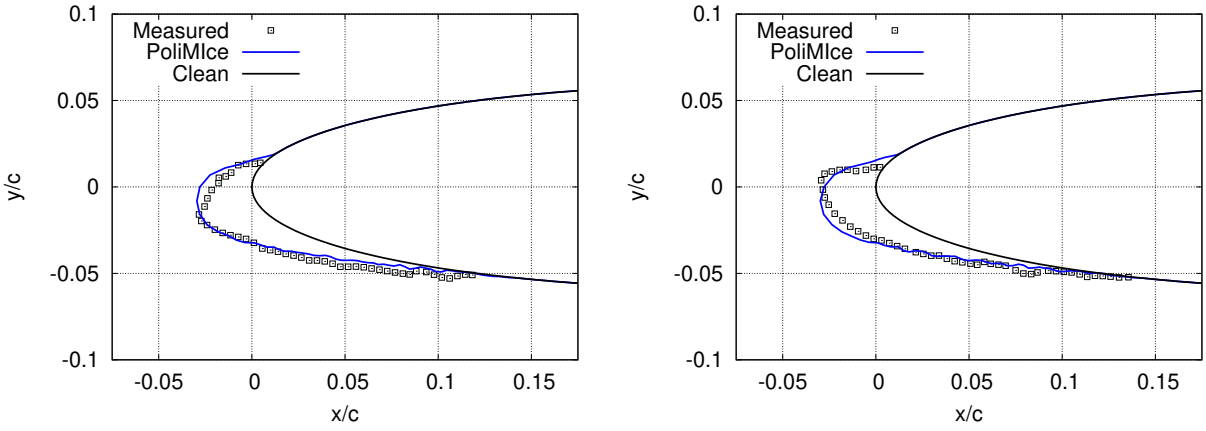
Table 3.1: Icing conditions based on the experimental work at the NASA Lewis Icing Research Center [100].

Run ID	Accretion Time $[s]$	Atmosph. Pressure $[Pa]$	Freestream Velocity $[m/s]$	Outside Air Temperature $[K]$	Liquid Water Content $[g/m^3]$	Mean Volume Diameter $[\mu m]$
Rime 1	420	101325	102.82	247.0	0.55	20
Rime 2	420	101325	102.82	262.0	0.55	20
Glaze 1	420	101325	102.82	267.5	0.55	20
Glaze 2	420	101325	102.82	271.0	0.55	20

Proceeding the icing validation assessment, the subsequent goal of this work was to generate a dataset of ice shapes for the Neural Network. The experimental work from Shin and Bond was selected with this in mind. In their work they studied the effect of 7 different temperatures on the ice shape. Due to the Neural Network requiring a significantly larger dataset, this work increments the temperature by $0.5K$ to provide 50 samples. Additionally, ice shapes at a MVD of 40 μm are simulated to provide 100 samples in total. The icing conditions used for all the samples are summarised in Table 3.2.

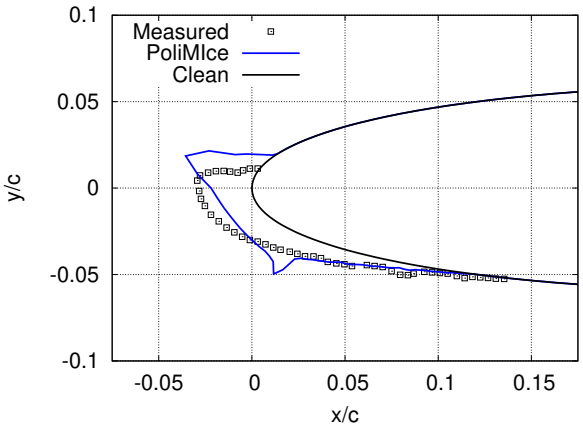
Table 3.2: Test matrix to produce sample of rime and glaze ice shapes.

Accretion Time $[s]$	Atmospheric Pressure $[Pa]$	Freestream Velocity $[m/s]$	Outside Air Temperature $[K]$	Liquid Water Content $[g/m^3]$	Mean Volume Diameter $[\mu m]$
420	101325	102.82	$246 \xrightarrow{(+0.5)} 271$	0.55	20, 40

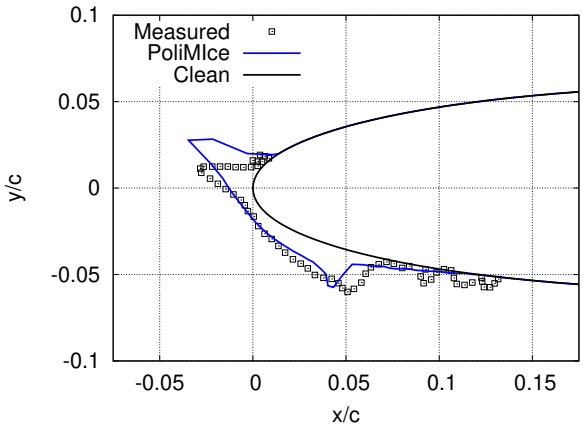


(a) Rime ice regime 1.

(b) Rime ice regime 2.



(c) Glaze ice regime 1.



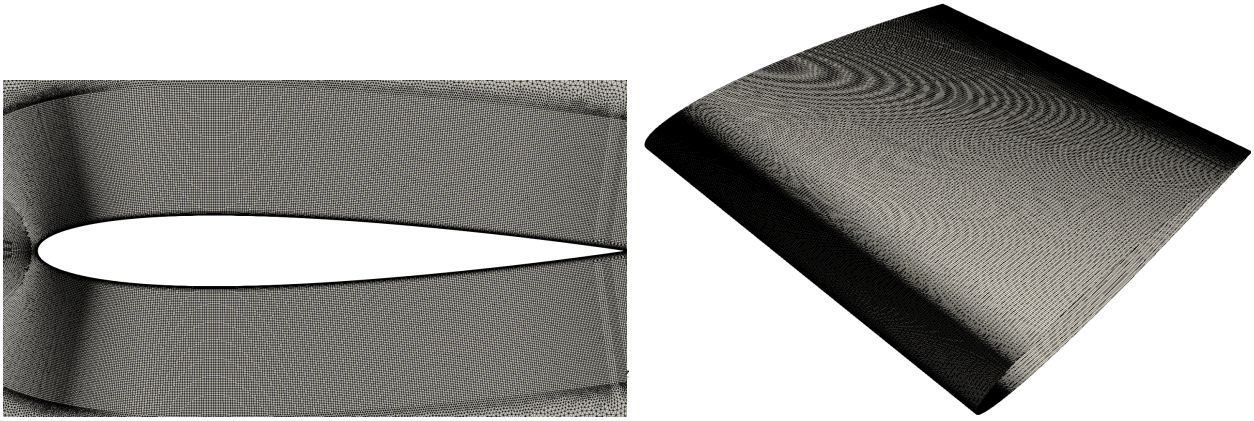
(d) Glaze ice regime 2.

Figure 3.3: Comparison of ice predictions with measurements at different temperatures.

3.3.2 Mesh Generation

The focus on modelling small scale vortical structures using scale resolving simulations capabilities in SU2 requires three-dimensional grids. An in-house grid generation software called uhMesh [101] is utilized to automatically generate a three-dimensional mesh from a two-dimensional ice shape. This is achieved by extruding the ice shape for 1 chord length in the spanwise direction. The assumption is that while the ice geometry contains no variation in the spanwise direction, the flow does due the scale resolving modelling approach. Simulating a fully three-dimensional icing dataset of this size would be beyond the current hardware capabilities. Additionally, the ice prediction approach based on Myers' model [102] would not capture the spanwise variation in the ice shape. To capture such three-dimensional roughness in the ice shape would require an approach such as the one suggested by Szilder and Lozowski [103]. However this is would be highly challenging to mesh in order to determine the aerodynamics and aeroacoustics of the iced wing.

Fig. 3.4 illustrate the mesh generation for the clean wing using this approach. With scale resolving simulations in mind, a boundary layer with a height of approximately $0.5c$ is used to generate a structured region surrounding the wing. The boundary layer at the wall is resolved to ensure $y^+ < 1$ everywhere. The mesh is extruded in the spanwise direction for 1 chord length and discretized by 150 elements. The total number of elements for each mesh is approximately 16M with it varying by as much as 500k depending on the ice shape.



(a) Close-up of the mesh surrounding the wing.

(b) Isometric view of the surface mesh.

Figure 3.4: Visualisation of automated 3D mesh.

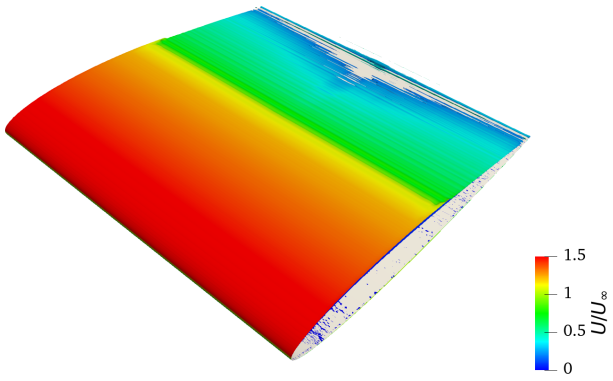
3.3.3 Turbulent Flow Simulation and Noise Prediction for Different Ice Shapes

To compare the turbulent flow field and far-field noise features of the clean and iced wings, the clean NACA0012 airfoil as well as a rime and a glaze iced airfoils associated with Rime 1 and Glaze 2 in Table 3.1 are presented. The scale resolving EDDES-FWH approach outlined in Sec. 3.2.2 is used to model the fine scale turbulent structures induced by the ice accretion. With ice detection on rotorcraft in mind, the $1c$ section of the wing is subsequently set to pitch around its quarter-chord to represent the cyclic pitching motion of a rotor blade in forward flight. While this is a primitive way to model a rotor blade in forward flight, it was a requirement due to rotorcraft icing simulations requiring fully three-dimensional simulations which are computationally very costly. The sinusoidal pitching motion of the wing can be described by $\alpha = 4^\circ \pm 4^\circ$ and pitching with a frequency of 6.0 Hz. The mean angle-of-attack is consistent with the icing experiments presented in Sec. 3.3.1. A time step of $\Delta t = 0.05C/U_\infty = 0.00025s$ is used, which results in 666 time-steps within each pitching cycle. The flow statistics and acoustics are computed after the 5th pitching cycle, over 10 pitching cycles.

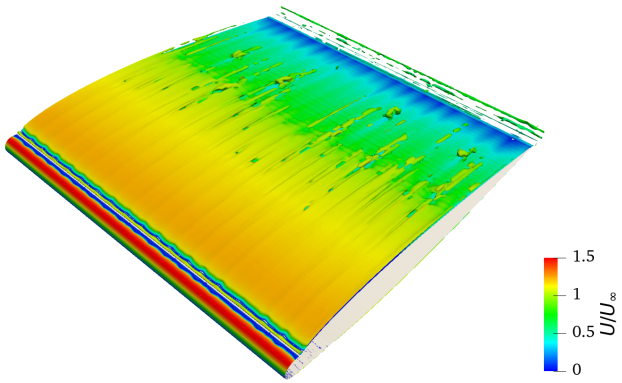
The instantaneous flow fields at AoA=4.0° on the upstroke are visualized by the iso-surfaces of the Q-criterion colored by nondimensionalized velocity magnitude, as shown on Figure 3.5. The effect of leading-edge icing is evident. While the clean and rime ice airfoil show mostly attached flows over the suction side, detached eddies signifying separated flow can clearly be seen in the more severely iced (glaze ice) airfoil, where the Q-criterion shows more fine-scale, three-dimensional turbulent structures over the entire suction side. Note that even though the ice shapes are quasi-2D, the resolved turbulent structures by EDDES are very much three-dimensional, especially in the glaze ice case where the Kelvin-Helmholtz instability is triggered immediately past the ice horn and develops quickly into 3D turbulence.

The time histories of the lift, drag, and moment coefficients of the rime and glaze ice wings, computed using the EDDES-SA simulations are compared on Figure 3.6. In the case of rime ice, the force and moment coefficients are very similar to those of the clean airfoil. Small differences between the clean and rime ice only occur at the peaks of the values which are associated with the maximum and minimum angle-of-attack. On the contrary, the glaze ice airfoil exhibits significantly deteriorated lift and strongly elevated drag, signifying flow separation over a large portion of the airfoil suction side. The moment coefficient is also severely effected during the glaze ice regime.

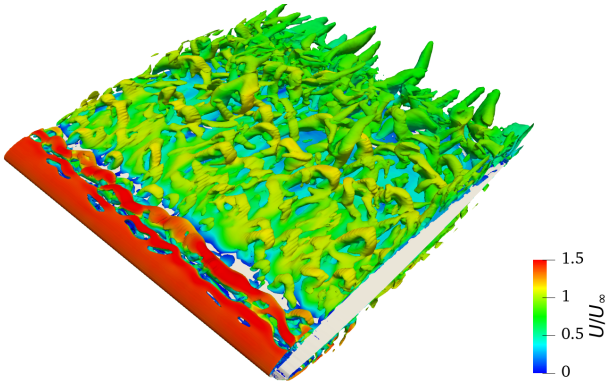
This is confirmed by examining the wall shear stress of the three cases shown on Figure 3.7. Both clean and rime ice airfoils show trailing edge separation due to the dynamic effect of the upstroke. Surface noise footprints visualized by the instantaneous wall pressure fluctuation ($p' = p - \bar{p}$) of the clean, rime and glaze ice airfoil sections are compared in Figure 3.8. The clean and rime



(a) Clean.



(b) Rime ice regime.



(c) Glaze ice regime.

Figure 3.5: Q-criterion iso-surface for the clean and iced wings.

ice airfoils exhibit similar surface noise footprints. On the other hand, the glaze airfoil exhibits a highly irregular noise footprint pattern with significant spatial frequency content. One may already speculate that the far-field noise signature of the glaze ice should be associated with elevated high-frequency contents. This is confirmed by the far-field noise spectra shown on Figure 3.10 – while the rime ice airfoil shows a slight increase of high-frequency noise over the clean airfoil, the glaze airfoil has a significantly elevated noise level over the entire high-frequency range up to the cutoff. Clearly, the more severe the leading edge ice accretion is, the more elevated the high-frequency range of the far-field noise tends to be. This can be exploited as a distinguishing feature in a machine learning model to build a mapping between the far-field noise level and ice shape as well as the associated aerodynamic performance of the ice airfoil.

3.3.4 Machine Learning Predictions of Aerodynamic Performance Under Icing Conditions

Following Sec. 3.2.3, BNNs are developed to predict aerodynamic performance metrics from inputs of acoustic measurements. Specifically, the BNN takes input of a compressed principal component analysis (PCA) representation of the power spectral density (PSD) of the acoustics time series, and predict the minimum, mean, and maximum values of the lift, drag, and moment coefficients over the duration of the time series window. To achieve this, the acoustic time series in the training dataset is pre-processed by computing the log values of PSD for each data signal covering the frequency range 17–1000 Hz, and perform a PCA to find and truncate to only the 6 leading components. All PSDs are then projected into this lower-dimensional representation. All inputs are then standardized using the mean and standard deviation statistics from across all training data simulations. For the output variables, I extract from each simulation's aerodynamic time series the minimum, maximum, and mean over the time window for C_L, C_D, and C_M, a total of 9 quantities of interest (QoIs). Overall, the BNN maps from a 6-dimensional input to a 9-dimensional output. The precise architecture is shown in Table 3.3, with one hidden layer of 15 neurons leading to a total of 249 model parameters. Small BNN architecture are deliberately selected due to the relatively few training samples available.

Table 3.3: BNN architecture.

Layer	Name	No. of Neurons	Activation Fcn.
1	input	6	-
2	hidden	15	swish
3	output	9	linear

To set up the BNN, the prior $p(w)$ from Eqn. (7.4) is chosen to be independent Gaussian $w_k \sim$

$\mathcal{N}(0, 0.2^2)$, and the likelihood $p(y_T \mid x_T, w)$ is chosen to reflect an additive independent Gaussian data noise: $y_k = f(x_k; w) + \epsilon_k, \epsilon_k \sim \mathcal{N}(0, 0.2^2)$. 600 particles are used to perform SVGD and pSVGD for our 249-parameter BNN. In order to obtain a robust measure of predictive performance of the BNN, 5-fold testing is conducted as follows. The overall dataset of 93 simulations is divided into 5 folds of 18-19 simulations each. Then, 5 BNN models are built where each model is trained on 4 of these folds and tested on the remainder fold. This allows us to obtain test performance statistics where each of the 93 data points get a chance to play the role of a testing point.

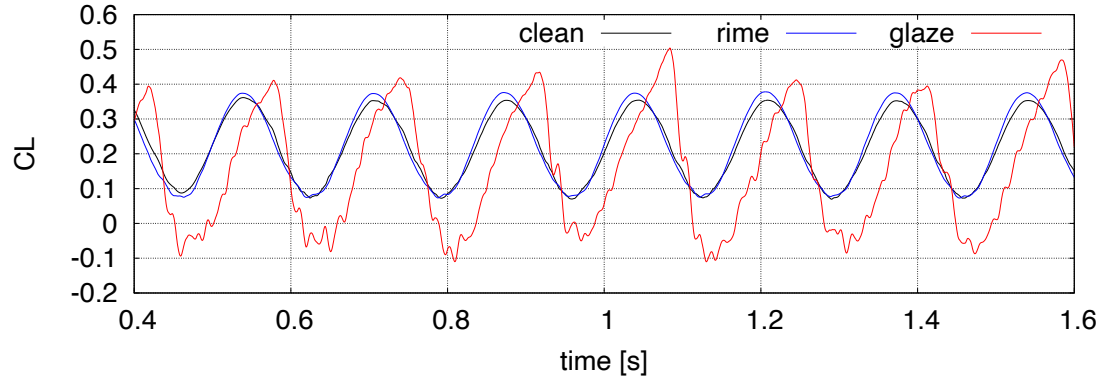
Figure 3.11 presents both the SVGD and pSVGD BNN predictions, with each panel showing an aggregated result from the 5 testing scenarios. By using BNNs, uncertainty information is now available along with predictions, depicted as error bars (± 1 standard deviation) in the plot. The SVGD and pSVGD results match well overall. pSVGD recovers slightly greater uncertainty of around 16% across all predictions compared to SVGD which is amenable to underpredicting uncertainty due to particle collapse. The reduced dimension from pSVGD is about 80% lower than that of SVGD. While pSVGD uses additional approximation in its eigenvalue truncation, I believe this truncation error is outweighed by the benefit from a greater particle-to-dimension ratio resulting from the dimension reduction, leading to an overall more accurate posterior representation for pSVGD.

In general, rime ice predictions are observed to achieve lower error and lower uncertainty than glaze ice predictions, which matches our intuition where generally more complex aerodynamic flows are caused by the more severe glaze icing conditions. These trends are shown in figure 3.12 where histograms of the prediction errors and uncertainty are gathered across all testing scenarios. Figure 3.13 further presents the posterior predictive distributions for C_D, C_L, and C_M QoIs between a specific rime ice case and a specific glaze ice case, illustrating an overall good agreement with the predictive distributions and the true target values, and the greater uncertainty in predicting under glaze than rime ice.

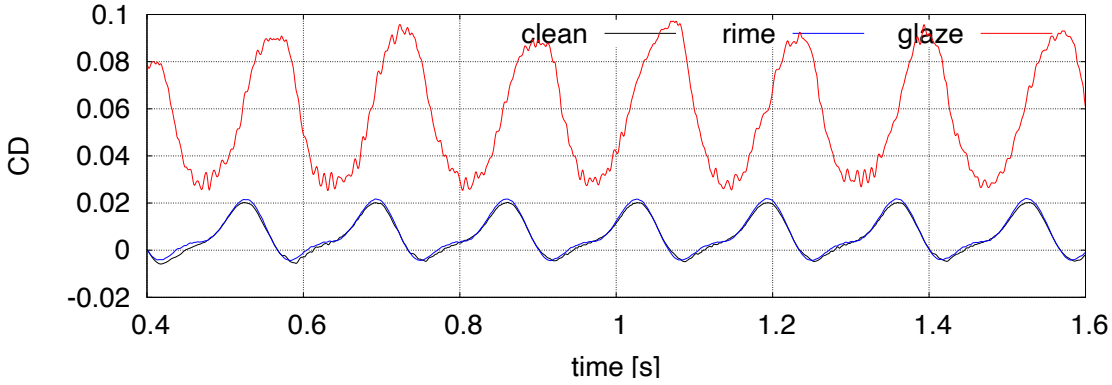
3.4 Conclusion and Future Work

This work introduces an original methodology for developing a Bayesian neural network acoustic ice detection system. The acoustic signatures generated by different iced wings are harnessed to predict their aerodynamic performance. An inter-disciplinary methodology is adopted to develop the Bayesian neural network acoustic ice detection system. The PoliMIce toolkit is utilized to predict a test matrix of iced airfoils. The aerodynamics and aeroacoustics associated with the iced airfoils are respectively solved in SU2 using high-fidelity EDDES scale resolving simulations and the FWH-F1A approach. A Bayesian neural network is then trained using projected Stein variational gradient descent to provide: (1) mapping from observed far-field broadband noise levels

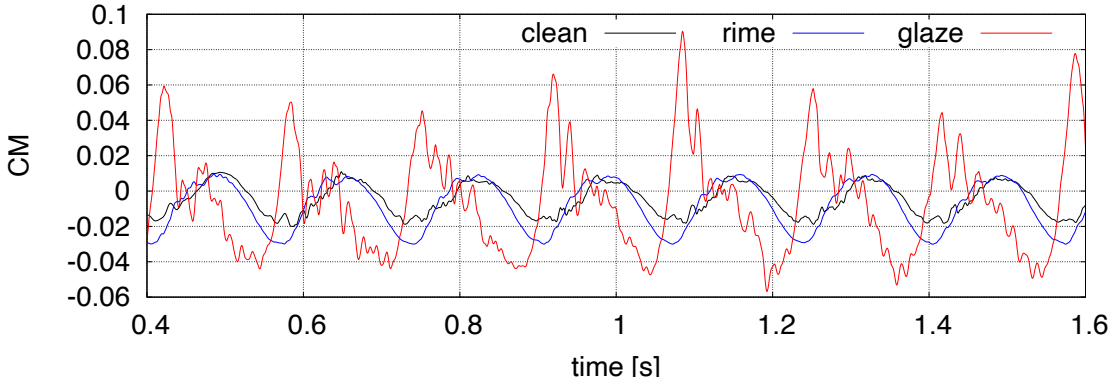
to predictions of aerodynamic performance indicators; and (2) uncertainty information reflecting the neural network's confidence in its predictions. While the icing simulation, aeroacoustic prediction as well as the construction of Bayesian neural network model all require potentially extensive computational resources in the *offline* phase, the prediction of aerodynamic performance indicators based on measured far-field noise level can be obtained rapidly in the time-critical *online* phase in-flight.



(a) Lift coefficient

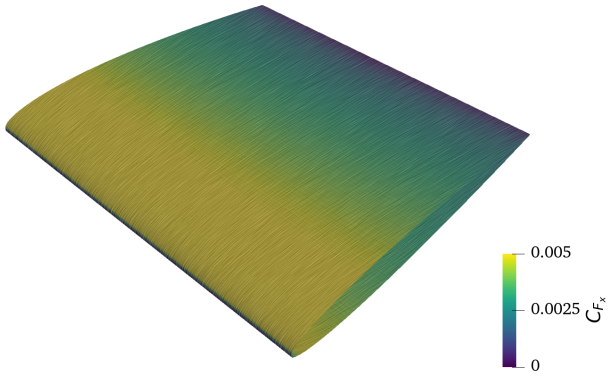


(b) Drag coefficient

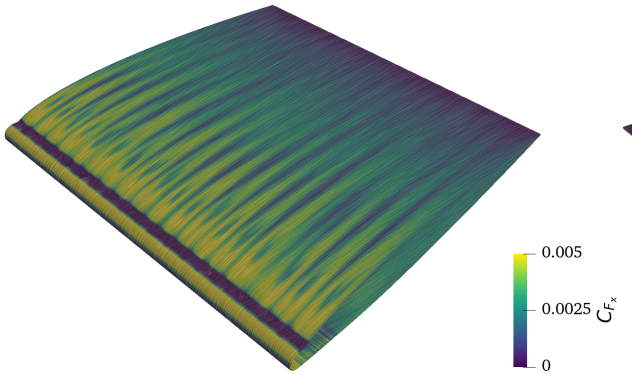


(c) Moment coefficient

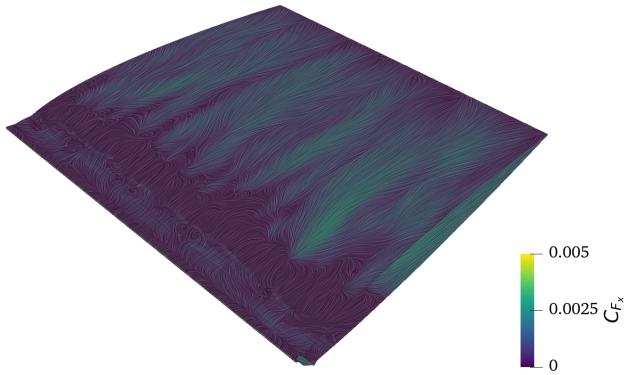
Figure 3.6: Time histories of lift, drag, and moment coefficients of the clean, rime and glaze ice airfoils computing using the EDDES-SA simulations



(a) Clean.

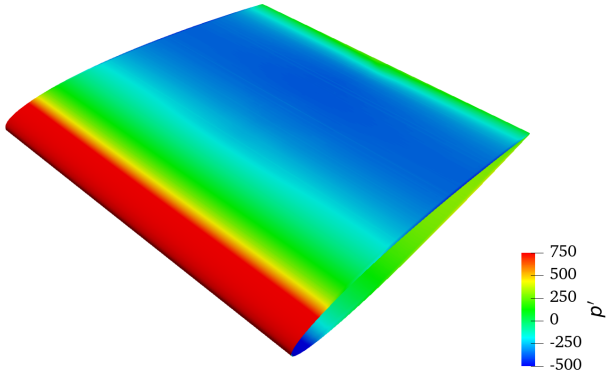


(b) Rime ice regime.

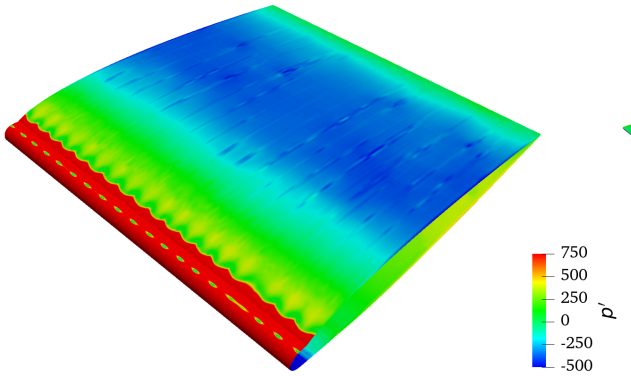


(c) Glaze ice regime.

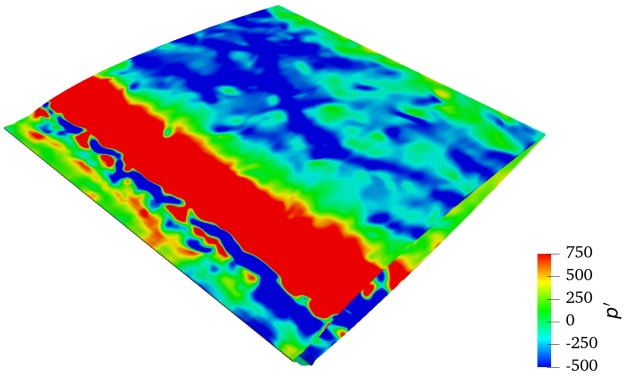
Figure 3.7: Skin friction coefficient, C_{F_x}, for the clean and iced wings.



(a) Clean.



(b) Rime ice regime.



(c) Glaze ice regime.

Figure 3.8: Instantaneous pressure fluctuations, p', for the clean and iced wings.

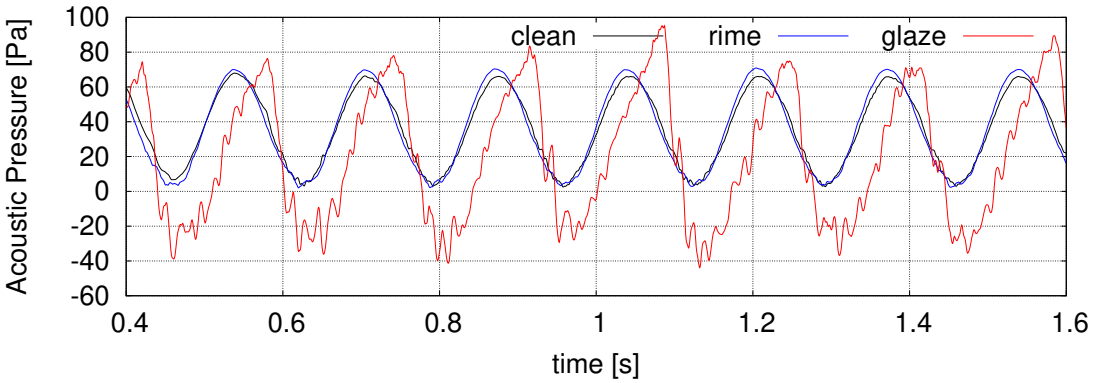


Figure 3.9: Time history of the pressure fluctuations, p', for the clean and iced wings

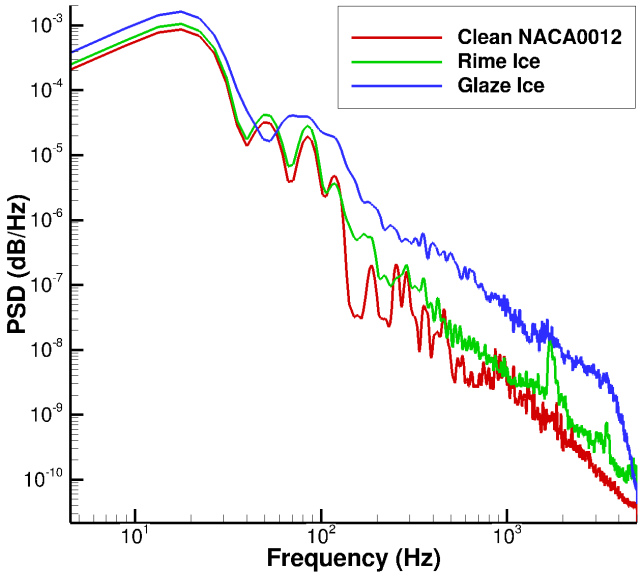


Figure 3.10: Far-field sound spetra of the clean NACA0012, rime and glaze ice airfoil sections, computed 20C below the center of rotation of the pitching airfoil

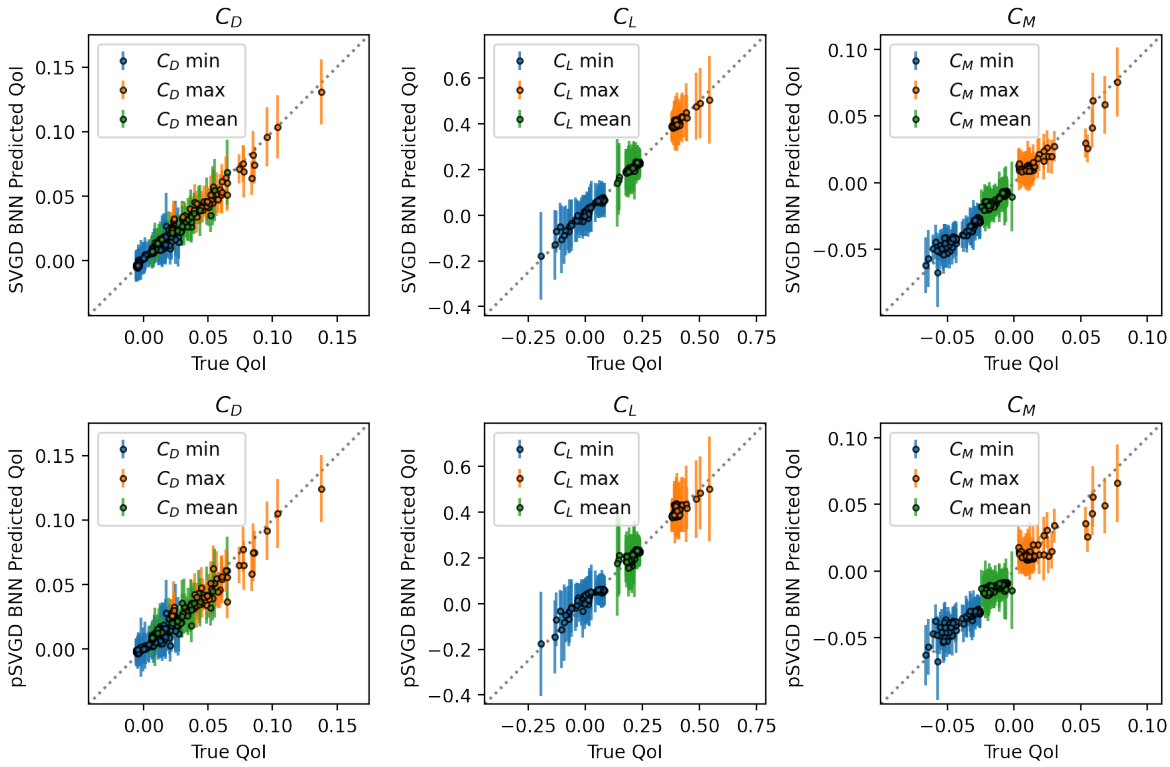


Figure 3.11: Predicted versus true values for all QoIs, using SVGD (top row) and pSVGD (bottom row). The two sets of results overall match well with each other. More specifically, pSVGD recovered approximately 16% more of the Bayesian predictive uncertainty relative to its SVGD counterpart that is amenable to underpredicting uncertainty due to particle collapse. pSVGD also achieved 80% dimension-reduction of the parameter space compared to SVGD.

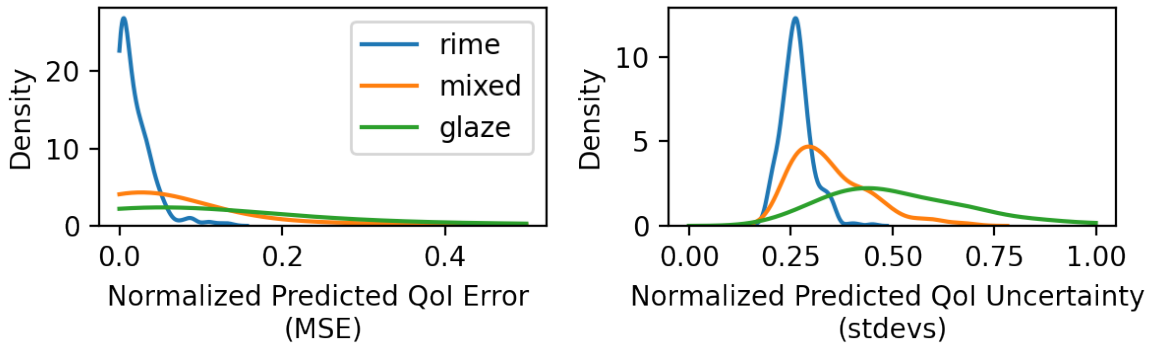


Figure 3.12: MSE of the posterior predictive distributions (left) and standard deviation of the posterior predictive distributions (right) averaged over all QoIs. The predicted QoIs are normalized to allow direct comparison and averaging.

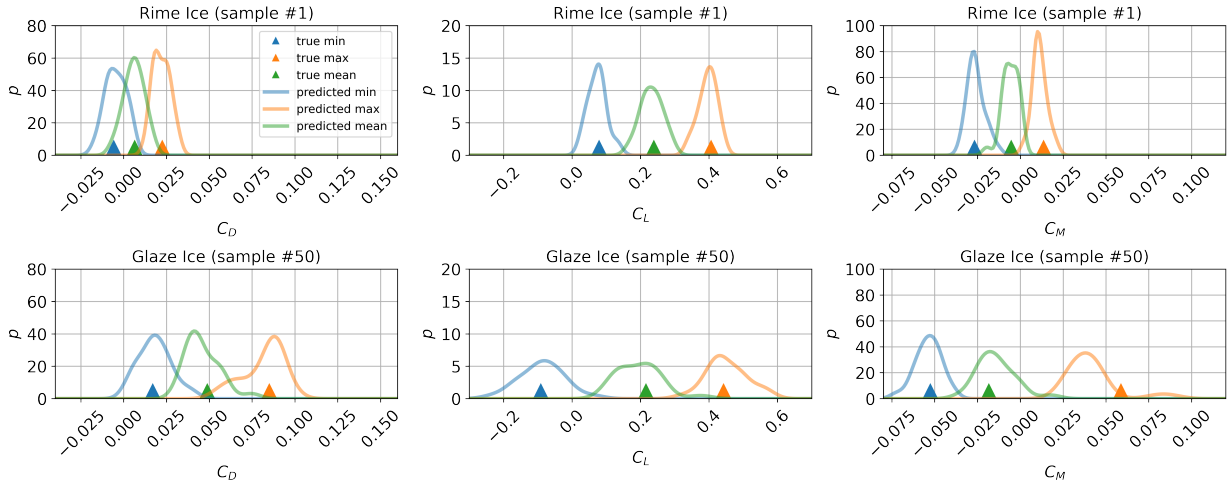


Figure 3.13: Comparison of posterior predictive distributions for a test sample of rime ice (top row) and glaze ice (bottom row) for C_D (left), C_L (center), and C_M (right). Ground truth values are indicated with arrows.

CHAPTER 4

Bayesian Graph Convolution Neural Network Models for Uncertainty Quantification of Time Evolving Poly-crystalline Stress Response

This chapter develops an investigation into the suitability of SVGD and pSVGD as a means of Bayesian uncertainty quantification in time-evolving metal polycrystaline stress deformation response. This type of metal deformation is an important part of metal forming manufacturing processes as well as vehicle crash analysis. Although the governing physics and associated partial differential equations are fairly well understood, repeatedly solving such equations in a design optimization setting would be infeasible. Bayesian graph convolutional neural networks (GCNN) are subsequently trained as a surrogate model to accelerate this process. The results of these GCNNs, calculated with recent SVGD and pSVGD algorithms are compared against the workhorse Bayesian inference algorithm, HMC. Lastly, results are shown on a larger GCNN that would be intractable with MCMC approaches. The work presented in this chapter was the result of close collaboration with Xun Huan and Sandia National Lab, specifically Reese Jones, Cosmin Safta, and Ravi Patel and is in the process of being submitted for peer review. While I preserve the majority of the paper to convey its comprehensive narrative, I would like to acknowledge that in the work presented in this chapter, Reese Jones and Cosmin Safta were chiefly responsible generating training data derived from physical equations and determining the initial GCNN architecture.

4.1 Introduction

The adaptability and effectiveness of neural networks has, in part, helped fuel the growth of scientific machine learning (SciML). Neural networks have been applied to a wide range of scientific tasks, including subgrid surrogate models [104, 105, 106, 107], solving partial differential equations [108, 109, 110], and components of optimal experimental design [111, 112]. SciML neural networks (NN) typically have at least hundreds of parameters, though often far more, and over-parameterized

NNs are common [113, 114, 115]. Unfortunately, as a result of this over-parameterization and of the complex feature relationships that are learned, NNs are typically uninterpretable and are often treated as "black box" models. Hence, the need for uncertainty quantification (UQ) for NN models is evident since it builds trust in these highly parameterized but minimally interpretable "black box" models.

There are a growing number of UQ methods that have been adapted to NN models. With easily adapted open source implementations, mean-field variational inference (VI) has been widely applied [116, 117]. VI is an efficient method but is well known to underestimate uncertainty by its construction due the use of: (a) a multivariate Gaussian surrogate posterior with a diagonal covariance (on the assumption of parameter independence) and (b) the Kullback-Leibler (KL) divergence to compare the surrogate and true posteriors which penalizes surrogate probability mass outside the true posterior. On the other-hand, Hamiltonian Monte Carlo (HMC) is generally seen as a reference standard for UQ and was applied to NNs as early as 1996 [118]; however, it is an inherently serial method and relies on long, computationally-expensive Markov chains. Stein variational gradient descent (SVGD) offers a flexible alternative to traditional VI methods. SVGD relies on an ensemble of particles with interacting dynamics representing likely models, analogous to multiple MC chains without handling the deficits of using entirely independent chains. The projected SVGD variant [119] aims to achieve further efficiency by only evolving the particles in the subspace of parameters significantly affecting the model output. Details of HMC, VI, SVGD, and pSVGD are all presented in Chapter 5. Section 5.1 describes the polycrystalline stress exemplar and the NN architectures used to model it is described. Section 4.3 compares the performance of SVGD on the exemplar using HMC as a reference and demonstrates its use on an scaled-up exemplar infeasible for HMC. Lastly this chapter concludes with a summary of findings and a discussion of potential future work in Section 4.4.

4.2 Methodology

This section develops a methodology for predicting the stress response of polycrystals undergoing plastic deformation [106], using a GCNN-RNN architecture. The underlying data-generating models are solutions to conservation equations on a regular domain and have a single quantity-of-interest (QoI) that evolves in time. This data and the governing physics equations are described in Section 4.2.1. In the first part of the investigation, architectures used are relatively inexpensive, both in terms of execution time and number of parameters, to allow for exhaustive investigations. Later, a larger, expensive GCNN-RNN model is used to demonstrate the projected Stein method.

4.2.1 Physical System

The deformation response of metal polycrystals underlies the phenomenological plasticity models that enable metal forming and crash analysis. Polycrystalline aggregates where crystal plasticity governs the individual grain response are a common target for homogenization, such as [120, 121, 122, 123, 106, 124]. Each polycrystalline sample is a collection of convex sub-regions Ω_K of a square $\Omega_\square$, and the number of Ω_K comprising the $\Omega_\square$ vary sample to sample. Each sub-region represents an individual crystal in the aggregate. The crystal lattice orientation distinguishes one crystal from another, which is characterized by the angles $\phi(\mathbf{X})$. For this 2D exemplar the angles were drawn from a uniform distribution, $\mathcal{U}[0, 2\pi]$.

For this exemplar, the average stress $\bar{\boldsymbol{\sigma}}(t)$ of a sample is the QoI. Crystal plasticity models [125, 126] are comprised of an algebraic relation mapping recoverable elastic strain to stress and a set of ordinary differential equations governing the evolution of irrecoverable plastic strain. The stress $\boldsymbol{\sigma}$ is given by

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{\epsilon}_e) = \mathbb{C} : \boldsymbol{\epsilon}_e \ , \tag{4.1}$$

where $\mathbb{C}$ is the 4th order elastic modulus tensor and $\boldsymbol{\epsilon}_e$ is elastic strain. In each sub-region Ω_K, the orientation vector ϕ rotates $\mathbb{C} = \mathbf{R}(\phi) \boxtimes \underline{\mathbb{C}}$ from the canonical $\underline{\mathbb{C}}$. Here $\boxtimes$ is the Kronecker product:

$$\mathbf{R} \boxtimes [\mathbf{A}]_{ij\ldots}\mathbf{e}_i \otimes \mathbf{e}_j \otimes \ldots = [\mathbf{A}]_{ij\ldots}\mathbf{R}\mathbf{e}_i \otimes \mathbf{R}\mathbf{e}_j \otimes \ldots \tag{4.2}$$

The common face centered cubic (FCC) symmetry was assumed for each crystal, which has three independent components of the elastic modulus tensor $\mathbb{C}$. Refer to Table 4.1 for all parameter descriptions and values. In each crystal, plastic flow can occur on any of the 12 FCC slip planes defined by slip plane normals $\mathbf{n}_\alpha$ and slip directions $\mathbf{s}_\alpha$, where $\mathbf{s}_\alpha \perp \mathbf{n}_\alpha$. The plastic velocity gradient, which determines the plastic strain,

$$\mathbf{L}_p = \mathbf{L}_p(\mathbf{S}) = \sum_\alpha \dot{\gamma}_\alpha(\mathbf{S})\, \mathbf{s}_\alpha \otimes \mathbf{n}_\alpha \tag{4.3}$$

is governed by slip rates $\dot{\gamma}_\alpha$ and the rotated dyads $\mathbf{s}_\alpha \otimes \mathbf{n}_\alpha = \mathbf{R}(\phi) \boxtimes \underline{\mathbf{s}}_\alpha \otimes \underline{\mathbf{n}}_\alpha$. The slip rate $\dot{\gamma}_\alpha$ follows a power-law

$$\dot{\gamma}_\alpha = \dot{\gamma}_0 \left| \frac{\tau_\alpha}{g_\alpha} \right|^{m-1} \tau_\alpha \ , \tag{4.4}$$

driven by the shear stress $\tau_\alpha = \mathbf{s}_\alpha \cdot \mathbf{S}\mathbf{n}_\alpha$ resolved on slip system α, while the slip resistance $\dot{g}_\alpha$ evolves according to [127, 128] :

$$\dot{g}_\alpha = (H - R_d g_\alpha) \sum_\alpha |\dot{\gamma}_\alpha| \ . \tag{4.5}$$

Elastic moduli	C_{11}	1.000
	C_{12}	0.673
	C_{12}	0.617
Hardening modulus	H	0.00174
Initial slip resistance	g_α	0.000597
Applied strain rate	$\dot{\bar{\epsilon}}$	1.000
Reference slip rate	$\dot{\gamma}_0$	1.000
Rate sensitivity exponent	m	20
Recovery constant	R_d	2.9

Table 4.1: Crystal plasticity parameters representative of steel. All moduli are made dimensionless by $C_{11} = 205$ GPa and all rates were non-dimensionalized by the the applied strain rate $1s^{-1}$. The maximum strain attained by the simulations was 0.003. Note the moduli $[\mathbb{C}]_{iiii} = C_{11}$, $[\mathbb{C}]_{iijj} = C_{12}$, and $[\mathbb{C}]_{ijij} = C_{44}$, where $i \neq j$.

Figure 4.1a shows 3 of the realizations on a 32×32 grid. For each sample the balance of linear momentum

$$\operatorname{div} \boldsymbol{\sigma} = \mathbf{0} \tag{4.6}$$

is solved for the displacement field, subject to tension at a strain rate of $1/s$ effected by minimal boundary conditions on the domain $\Omega_\square$, i.e. two parallel boundaries and one corner are constrained. Figure 4.1b illustrates the inhomogeneities in the stress response (at 0.3% strain in tension), which are particularly marked at grain boundaries. The overall response of samples reflects the anisotropic nature of each crystal which share boundaries and, hence, have kinematic constraints on their deformation. The resulting mean stress $\bar{\boldsymbol{\sigma}}(t)$ response curves

$$\bar{\boldsymbol{\sigma}}(t) = \frac{1}{V} \int_\Omega \boldsymbol{\sigma} \, \mathrm{d}^3 X \tag{4.7}$$

are the QoIs. They vary due to the particular texture $\phi(\mathbf{X})$ of the realization, where V is the volume of $\Omega_\square$. In fact, the different crystal textures evoke different effective elastic slopes, yield points and (post-peak) flow stresses as can be seen in Figure 4.2.

4.2.2 Neural Network Architecture

The general neural network (NN) architecture for this data has been published before [104, 106] and is depicted in Figure 4.3. The physical inputs are the orientation field $\phi(\mathbf{X})$ (Figure 4.3) and the applied strain $\mathsf{f} = \bar{\boldsymbol{\epsilon}}(t)$; the output is the mean stress over time $\mathsf{y} = \bar{\boldsymbol{\sigma}}(t)$. The orientation field $\phi(\mathbf{X})$ is discretized on a mesh. The discrete values of the texture ϕ and the associated cell volume fractions v comprise the cell data $\mathsf{X} = [\phi, v]$. The nodal input data X and the adjacency matrix A for the mesh are fed into a convolutional stack. The graph convolutional neural network (GCNN)

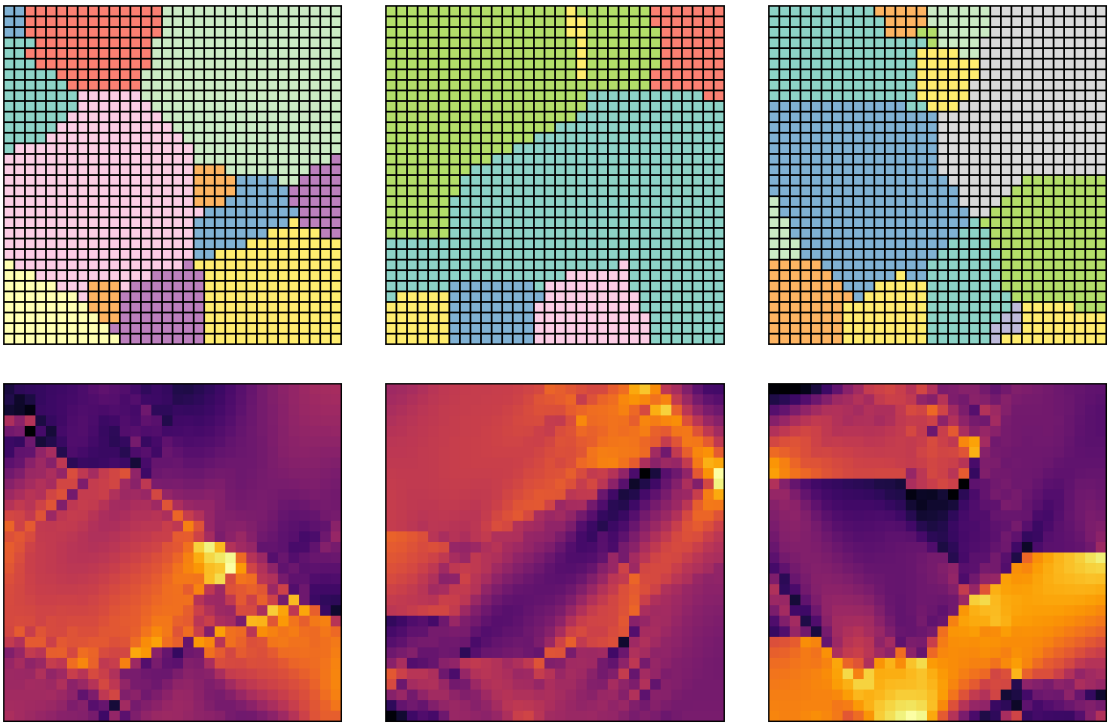


Figure 4.1: Realizations of the crystal orientation angle ϕ (upper row) and the corresponding stress σ (lower row).

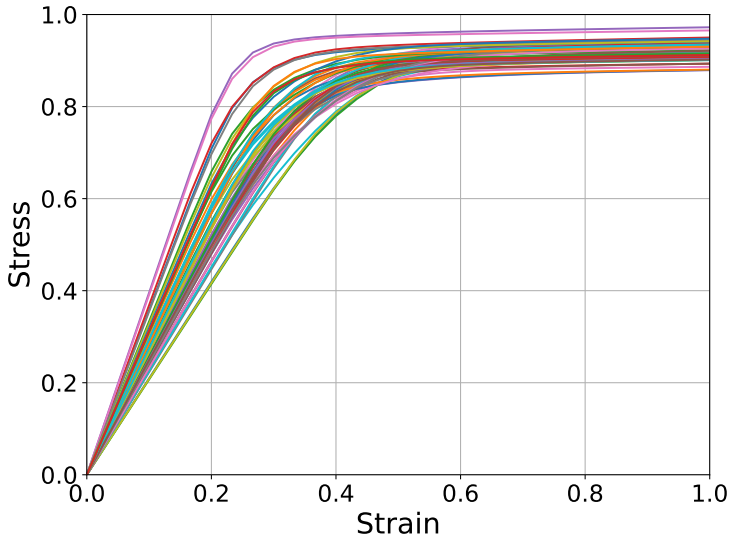


Figure 4.2: Normalized stress response curves are shown for different polycrystalline configurational realizations.

layers have the form of the Kipf and Welling GCN [129] with an independent self weight. The result of this convolutional stack is pooled with global averaging to produce hidden features that

describe the factors that are salient to the prediction of the mean stress. There is one feature per filter in the convolutional layers. These features depend only on the initial microstructure as described by $\phi(\mathbf{X})$ and, hence, are time-independent. These features and the time dependent input $\bar{\boldsymbol{\epsilon}}(t)$ are concatenated and fed into a Gated Recurrent Unit (GRU) [130] recurrent neural network. The output of this layer is fed to a linear dense layer to produce $\bar{\boldsymbol{\sigma}}(t)$. Figure 4.3 provides a schematic of the NN model with further details.

A larger version of the model is also investigated in the results section to demonstrate pSVGD on a model that is intractable to HMC. This GCNN-RNN model operates on 3D data with 2 convolutional layers with 16 filters, 1 post pooling dense layer with *swish* activation, and 1 dense output layer. It has 2609 parameters, with 112+528 in the convolutions, 272+17 in the dense layers and 1680 parameters in the GRU.

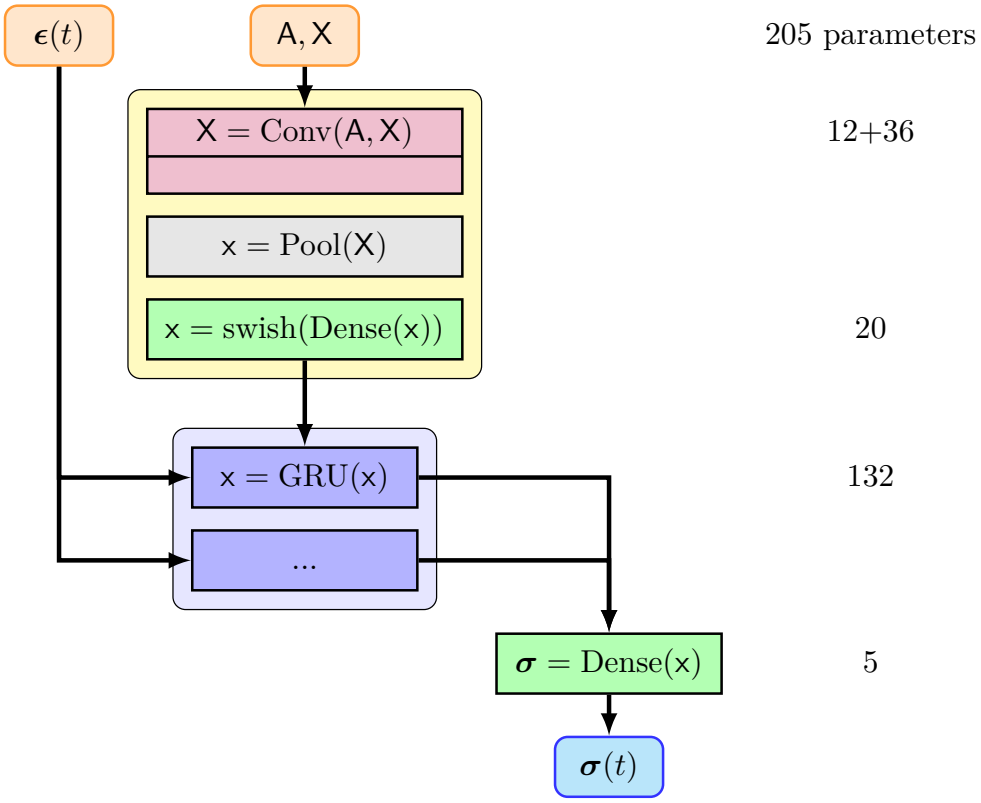


Figure 4.3: GCNN-GRU model of the crystal plasticity data, $N_w = 205$. The 2 convolutional layers with 4 filters each and *swish* activation. The RU has tanh activation and is depicted as unrolled with a layer per time-step.

4.2.3 Sampling and Training

This section provides method implementation details and NN training hyper-parameters. Full details on the HMC, SVGD, and pSVGD approaches are explained in Chapter 5.

Adam optimizer [131] with early stopping is used to find MAP estimates. The optimizer minimizing objective is the negative log likelihood $-\log \eta$ with the negative log prior as a regularizer, ie the negative log posterior. The prior was taken to be normal distribution $\mathcal{N}(\mathbf{0}, \sigma\mathbf{I})$ with zero mean and diagonal covariance $\sigma\mathbf{I}$ with $\sigma = 1$. Likewise, the likelihood taken to be normal with the data *noise* estimated as $\sigma = 0.05$ for the crystal plasticity data and $\sigma = 0.02$ for the gas flux data. All data $\mathcal{D} = \{\mathsf{X}, \mathsf{f}, \mathsf{y}\}$ was normalized to lie in $[0, 1]$.

To boost efficiency of HMC in terms of computational time and memory for the larger datasets, the gradient evaluation of the likelihood, Equation (2.6), used a random subset of the data that was resampled every evaluation to estimate the full likelihood. This can be seen as a form of pseudo-marginal subsampling [132].

The SVGD sampling initialized the N_p particles with the MAP values with some jitter from a prior to make them distinct. This implementation of SVGD sampling uses the following Gaussian kernel:

$$\varphi(\omega_i, \omega_j) = \exp\left(-\frac{1}{h}\|\omega_i - \omega_j\|^2_{\tilde{\mathsf{H}}}\right), \tag{4.8}$$

where $\tilde{\mathsf{H}}$ is an approximate Hessian matrix and h is the bandwidth. An adaptive bandwidth $h = \bar{d}^2/\log N_p$ was selected, where $\bar{d}$ is the median pair-wise distance between particles. The projected SVGD used the same initialization and used a tolerance on the eigenvalues of the Hessian to construct a subspace from its eigenvectors. The Hessian-based reductions were done layer-wise to prevent disconnected networks.

4.3 Results

In this section, the 2D NN models described in Section 4.2.2 are used to compare SVGD to the reference standard HMC. Next a larger 3D NN is used to explore the a problem infeasible with HMC.

In the following, $\mathsf{y}_i = \mathbb{N}(\mathsf{X}, \mathsf{f}; \omega_i)$ is referred to as model *realizations* and $(\mathsf{X}_J, \mathsf{f}_J, \mathsf{y}_J)$ as data *samples*. Data at particular time steps will be indicated via $[\mathsf{y}]_a$.

4.3.1 Comparison of HMC and SVGD

This section compares the UQ provided by SVGD and HMC for the plasticity exemplar. Since many NN are over-parameterized, two cases are examined: (a) $N_D = 1/2N_w$ with the number of

samples being half the number of weights, and (b) $N_D = 2N_w$ with the number of samples being twice the number of weights. In this case the number of SVGD particles $N_p = 96$ was on par with the number of weights.

As Figure 4.4 depicts, HMC chains have difficulty mixing (high correlation between successive samples), likely due to the fungibility of the NN weights. HMC does jump between different modes but it does so very infrequently. Future analysis might leverage a multistart strategy to get a multitude of MAPs, each of which can initialize a number of MCMC chains. Figure 4.5 shows low dimensional representation of the HMC chains, using both T-SNE and spectral embedding methods. These give insight to that fact that projective algorithms, such as pSVGD, might leverage such low dimensional latent spaces.

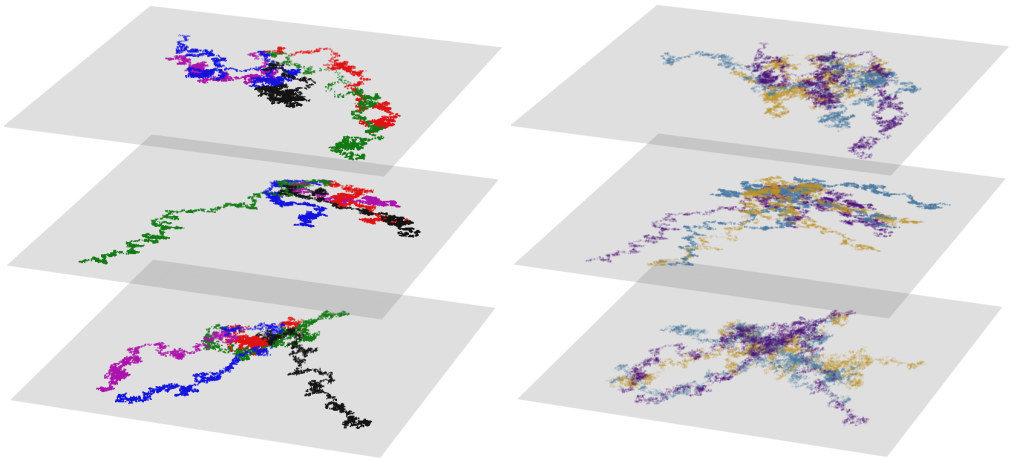


Figure 4.4: HMC chains for the GRU model showing the dynamics for several parameter pairs. Left frame shows chain samples that start at the same MAP value, with different random number generator seeds. The right frame shows samples obtained from chains initialized from different MAP values. Results were thinned out (shown every 10^3 samples for clarity.

Table 4.2 gives the distance correlation between the weight ensembles by layer. The distance correlation measures nonlinear correlations and is applicable to sets that are not necessarily equal size [133, 134]. In terms of standard covariance

$$d_{\text{cov}}(X,Y) = \text{cov}(\|X - X'\|, \|Y - Y'\|) - 2\,\text{cov}(\|X - X'\|, \|Y - Y''\|) \tag{4.9}$$

the covariance follows. For the GRU all layer parameters are highly correlated, with the correlation

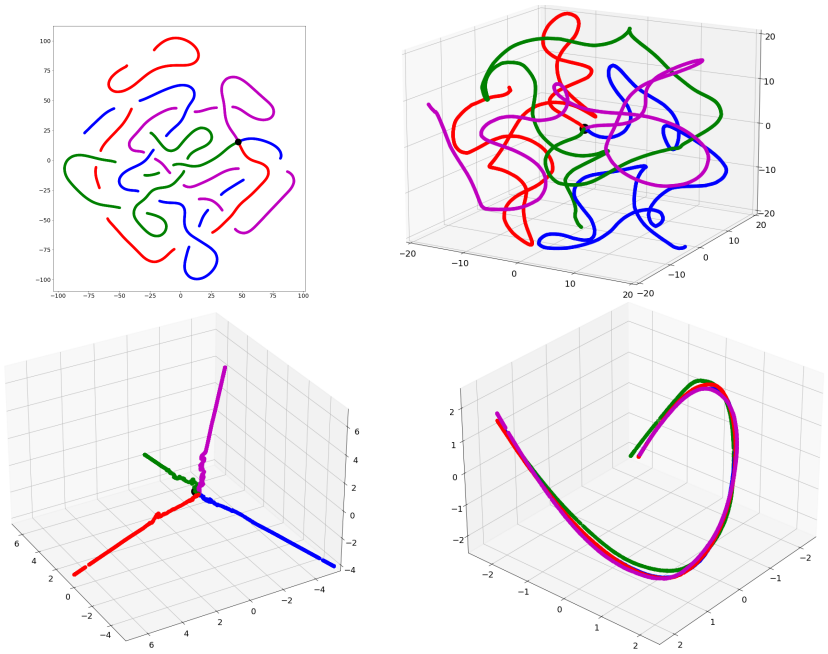


Figure 4.5: GRU: Low-dimensional embeddings for the HMC chains corresponding to the case using $N_d = 1/2N_w$. Results correspond to 4 chains with different RNG seeds, all starting from the same MAP value (location shown with black circle in some of the plots). Top row shows 2D and 3D data processed with the T-SNE algorithm; bottom row show results generated via spectral embedding.

with the output having a slight increasing trend from input convolution (*GCNN1*) to the pre-output dense layer. The correlations for NODE on the other hand, show that the graph convolutions are highly correlated and work together to represent the complex generation behavior while these layers are only loose correlated with the single parameter (*Scale*) used in the diffusion component. The NODE also shows increasing correlation of the layers from input to output with the output layer.

	GCNN1	GCNN2	Dense	GRU	
$N_D = 105$	GCNN2	0.90–0.94			
	Dense	0.91–0.92	0.93–0.96		
	GRU	0.92–0.94	0.97–0.98	0.95–0.98	
	Output	0.72–0.73	0.78–0.83	0.81–0.85	0.81–0.85
$N_D = 420$	GCNN2	0.92–0.93			
	Dense	0.91–0.94	0.93–0.96		
	GRU	0.91–0.93	0.96–0.97	0.96–0.97	
	Output	0.58–0.83	0.75–0.85	0.75–0.89	0.78–0.89

Table 4.2: Distance correlation values for the HMC-based inference using the GRU model and $N_D = 105$ (top) and $N_D = 420$ (bottom) data samples.

Figure 4.6 show the GCNN-RNN pushforwards for three cases the minimum, median, maximum MAP discrepancy. The MAP discrepancy is defined in this work as error in MAP PF with respect to the corresponding data sample:

$$\|\mathsf{y}_I - \mathbb{N}(\mathsf{X}_I, \mathsf{f}_I; \omega_{\text{MAP}})\|. \tag{4.10}$$

The confidence bands of the PFs show some signature of uncertainty growing in time and being largest where the there is the most variance in the data. The confidence bands also show that the SVGD appears to be better at cover the data; however, the HMC and SVGD predictions are more comparable when the whole data ensemble is taken into account.

Figure 4.7 measure the dissimilarity between HMC and SVGD and each with the data. Figure 4.7 The Kolmogorov-Smirnov statistic (KSS) is used to measure the similarity of two CDFs:

$$\text{KSS}(\text{CDF}_1, \text{CDF}_2) = \int_{-\infty}^{\infty} |\,\text{CDF}_2(y) - \text{CDF}_1(y)|\,\mathrm{d}y \tag{4.11}$$

where the cumulative distribution function (CDF) is approximated with the empirical CDF constructed with the push-forward distributions given samples from the HMC chains and Stein VI particles, respectively. To compare an ensemble of PFs with data, a Heaviside function centered at the data value y is used as the data CDF. These statistics also show the signatures of the variance of the output features through time. Similarities between the predictions and the data are good throughout time but are lower (higher scores/stats) in the highly variable, hard to predict regimes. The highly variable plastic transition in the region of $[0.1, 0.4]$, where the elastic-plastic transition occurs, is seen in the GRU similarity scores.

Finally Figure 4.8 shows the correlation over time steps .

$$c_{ab} = \langle [\Delta\mathsf{y}(\mathsf{X}_J, \omega_i)]_a [\Delta\mathsf{y}(\mathsf{X}_J, \omega_i)]_b \rangle_{Kl} \quad \text{with} \quad \Delta\mathsf{y}(\mathsf{X}, \omega_i) \equiv \mathsf{y}(\mathsf{X}, \omega_i) - \mathsf{y}(\mathsf{X}, \omega_{\text{MAP}}) \tag{4.12}$$

where the average is over weights i and inputs J. The plots show that HMC and SVGD predict similar covariance structures albeit with some qualitative differences. For the GRU, the HMC covariance structure is more diagonally dominant, while SVGD appears to be capturing significant cross correlation in the elastic-plastic transition.

4.3.2 3D Stress Evolution

This section demonstrates SVGD and pSVGD on a model infeasible for HMC due to the size of the parameter space. As mentioned in section 4.2.2, the architecture used is a larger version of the GCNN-GRU model with 2609 parameters: 640 in convolution layers, 1680 in GRU, and rest in

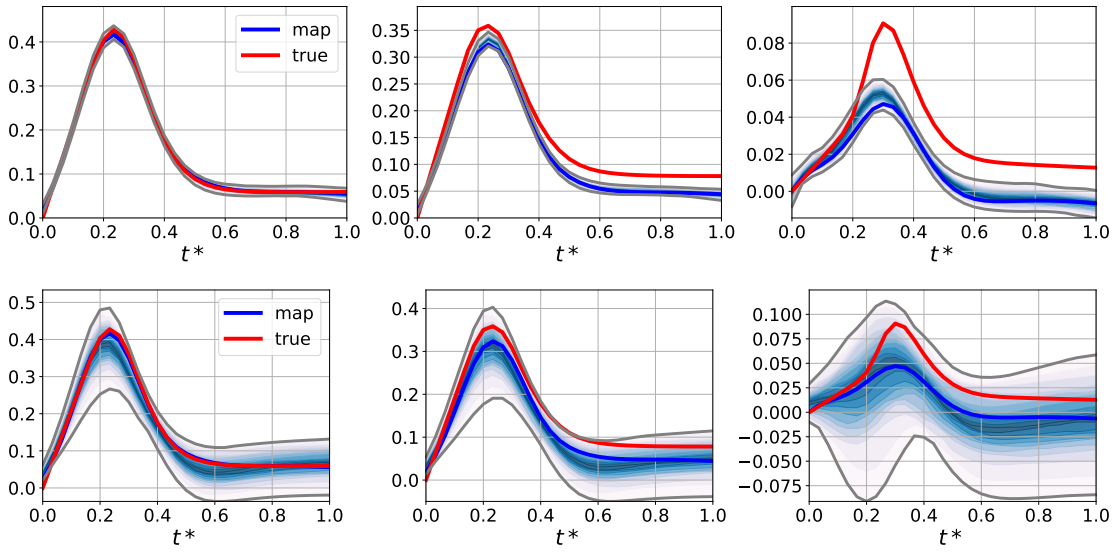


Figure 4.6: RNN: Push-forward distributions for minimum (left), median (middle), maximum (right) MAP discrepancy for $N_D = 1/2N_w$; top row shows HMC results and bottom row shows SVGD results. Note the change in vertical scale from left panels to right.

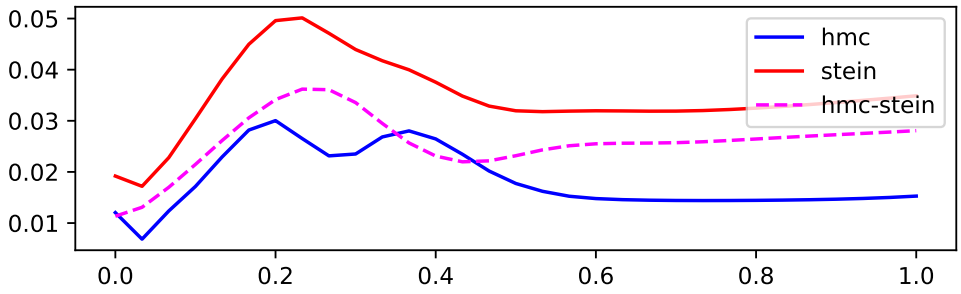


Figure 4.7: CPR score for HMC and SVGD and Kolmogorov-Smirnov distance between HMC and SVGD.

dense layers. SVGD and pSVGD were limited to just 32 particles due to memory constraints on a single GPU.

Figure 4.9 shows the eigenspectrum of the Hessians of all the layers. Note the large range and the steep drop off in magnitude. Figure 4.9 also shows profile of reduction for two pSVGD cases: 3% and 10% reduction in the active weight space. The parameters are primarily eliminated in the GRU which has the the largest proportion of the weight and also controls the stability of the time evolution. Figure 4.10 shows the pushforward for the minimum, median and maximum MAP discrepancy. In this figure shows the bundle of PFs trajectories to illustrate the quality of the PFs for full SVGD, projected SVGD with 3% reduction, and pSVGD with 10% reduction. The 3% reduction is visibly similar to the full SVGD results; however, the realizations with 10%

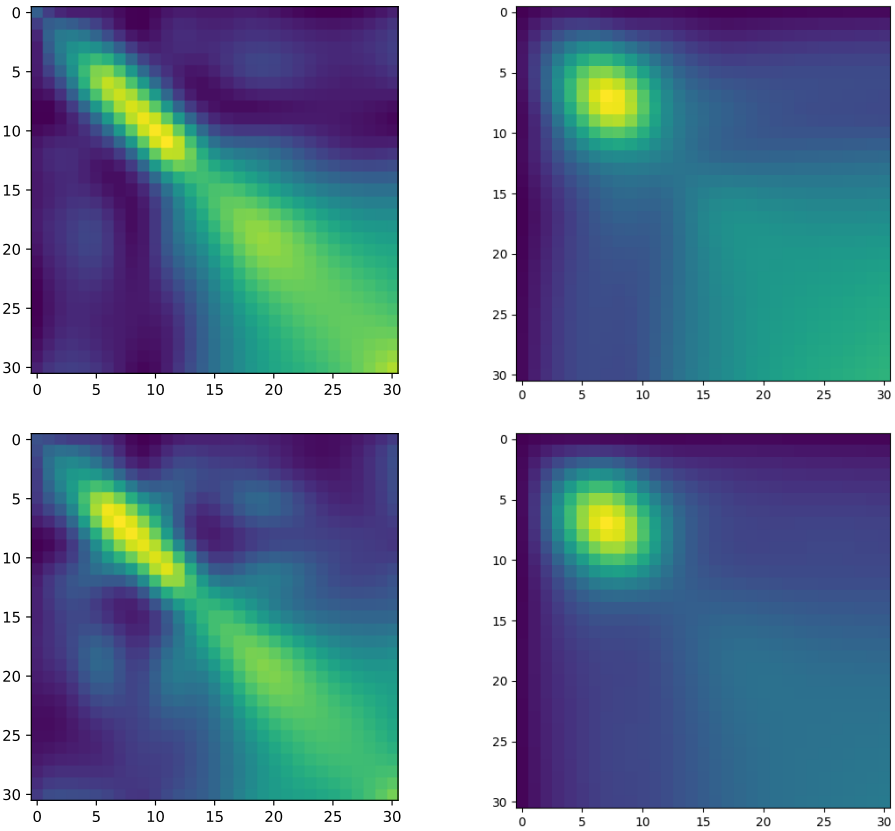


Figure 4.8: GRU: comparison of posterior distributions over time: compare HMC(left) vs SVGD(right) for $N_D = 1/2N_w$ (top), $2N_w$ (bottom).

weight reduction indicate that the greater reduction is sampling some trajectories that appear to be marginally stable, which may be incurred by a relatively wide prior.

Figure 4.11 compares the covariance structure of the SVGD and pSVGD with 3% reduction. The covariance matrices are largely similar. The pSVGD covariance structure show more variance at the initial state and less in the elastic-plastic transition regime. Figure 4.12 shows the similarity of the PF distribution with the data as measured by CPRS, and similarity of the PF distributions with KS. For $N_D = 1000$, the full and 3% reduction SVGD are nearly indistinquishable except near initial state, while both are far from the 10% reduction. Figure 4.12 shows the effect of dataset size. Clearly the PF predictions become farther from the data with less training data. At the same time the difference between full and projected SVGD becomes less, as the prior becomes relatively more influential. Figure 4.14 shows distribution similiarity across multiple (9) MAPs.

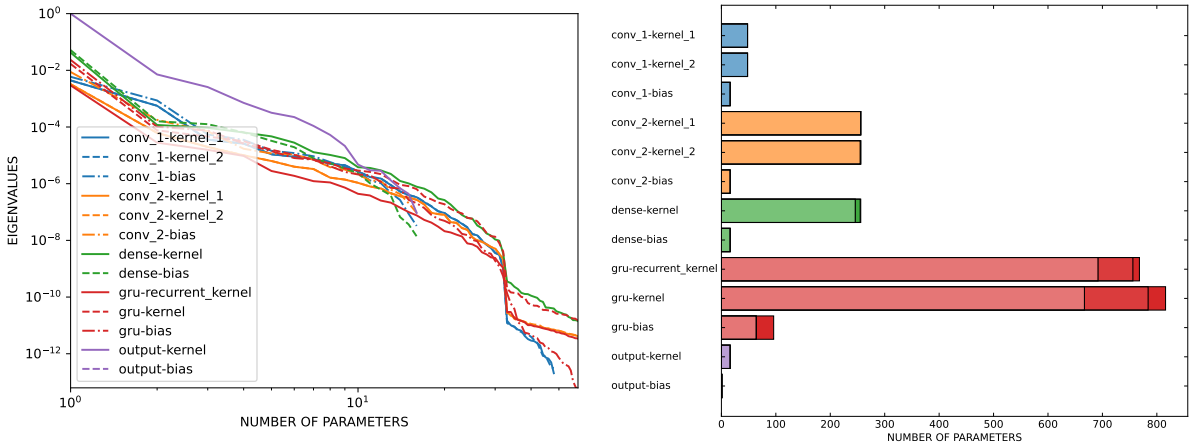


Figure 4.9: 3D GRU: energy per layer and reduction profiles (right). For the reduction profiles, the three models, full, 17% reduction, and 32% reduction, are shown by stacked bars.

4.4 Conclusion and Future Work

Building upon the successful implementation of SVGD and pSVGD in BNN in Chapter 3, this chapter develops an implementation of Stein methods to model uncertainty of metal polycrystaline stress deformation response. Whereas the BNN in Chapter 3 were limited to small, dense neural networks on frequency-binned data, the BNN in this chapter make full use of advanced GCNN-RNN to automatically learn features and propagate predictions along with predictive uncertainty through time. In a number of specific case studies, uncertainty typically grew modestly over time, but high gradients in the function being approximated typically had a greater influence on the predictive uncertainty.

In contrast to Chapter 3, pSVGD had difficulty projected down to a lower dimensional subspace while still producing stable time propagation. Although this algorithm was able to work on the 3D GCNN-GRU model that was infeasible for HMC, results showed little improvement in posterior predictions relative to SVGD.

Future work will largely focus on understanding these stability issues. One possible limitation is the very low number of particles used in the SVGD and pSVGD models. In SVGD, this limitation would likely lead to some degree posterior collapse, artificially deflated the posterior variance. However in pSVGD, it also has the effect of limiting the Hessian approximation to the number of particles. Future research will leverage greater parallel processing of many GPUs.

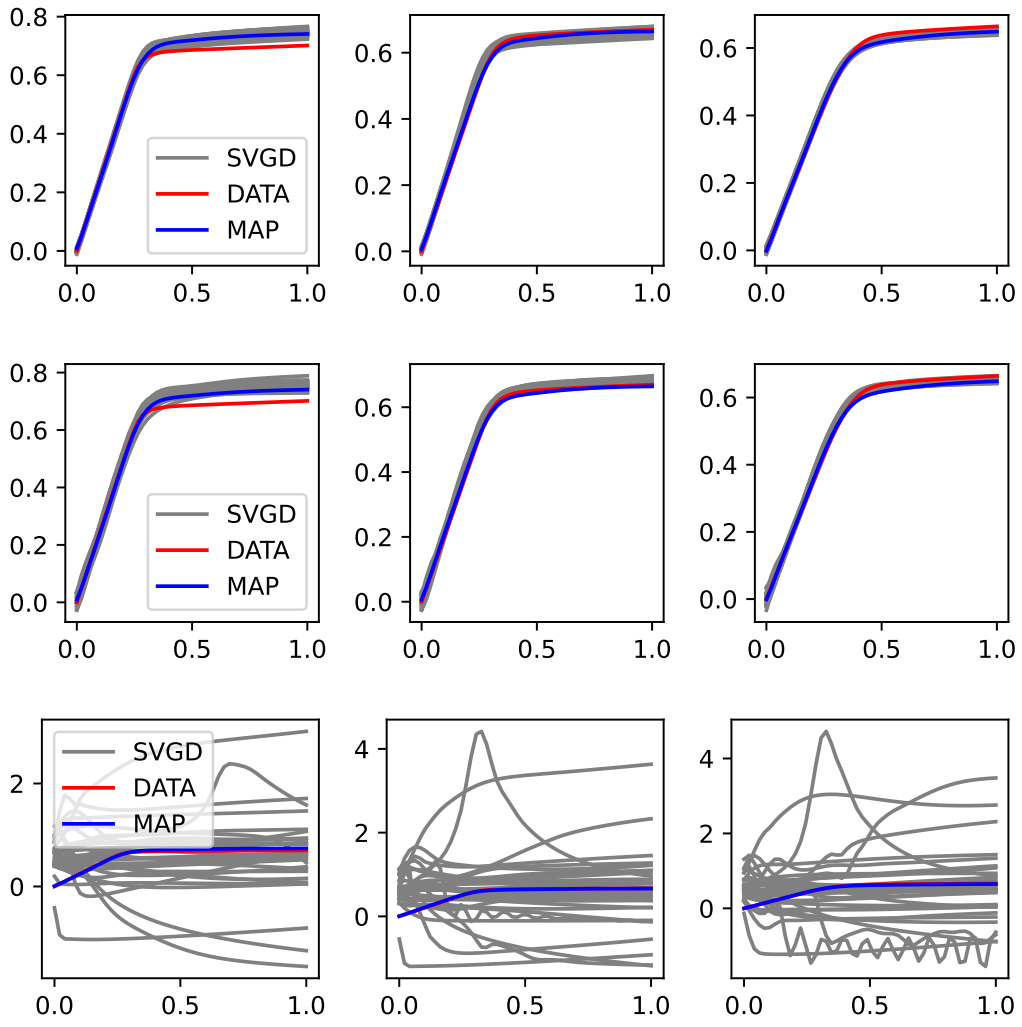


Figure 4.10: 3D GRU: PFs for minimum (left), median (center), and maximum (right) MAP discrepancy for SVGD (top panels) and projected SVGD with 3% (middle panels) and 10% reduction (bottom panels)

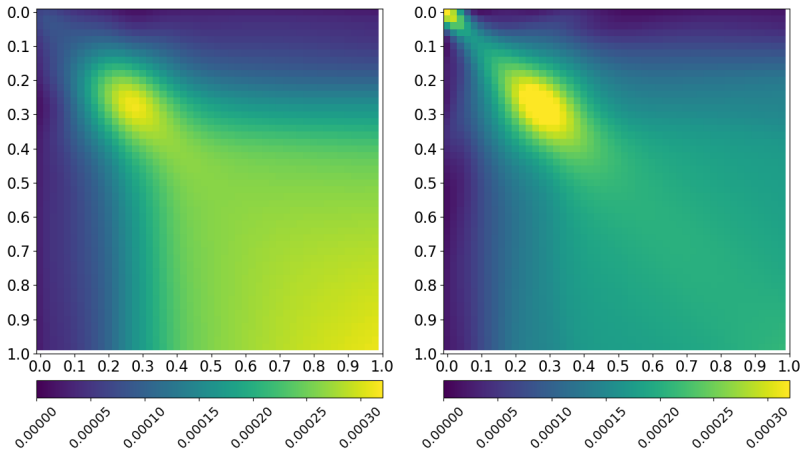


Figure 4.11: 3D GRU: covariance of PFs for SVGD and pSVGD with 3% weight reduction

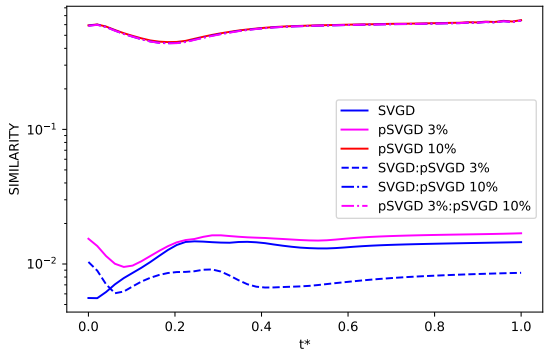


Figure 4.12: 3D GRU: comparison of full SVGD and projected and data (solid lines) and between these two ensembles of PFs (dashed lines).

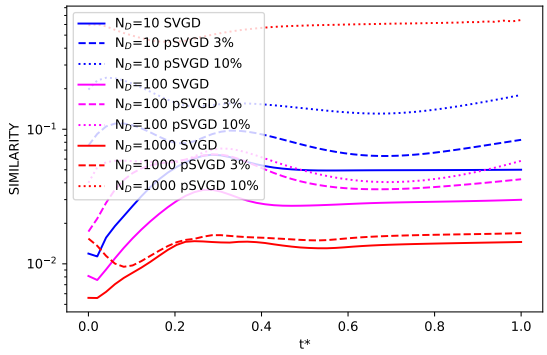


Figure 4.13: 3D GRU: KSS of full (solid), 3% reduction (dashed) and 10% reduction (dotted) SVGD with data for N_D = 10,100,1000 (blue,magenta,red).

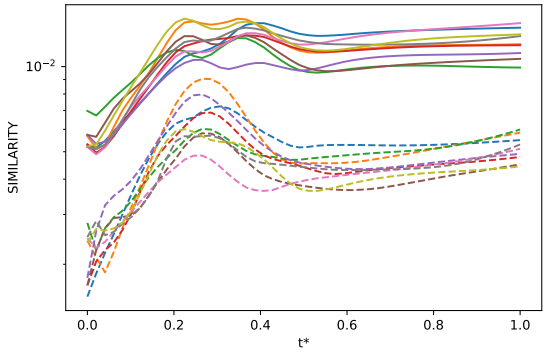


Figure 4.14: 3D GRU: SVGD for an ensemble of MAPs, KSS with data (solid) for each and KSS of the particular SVGD with ensemble without the particular SVGD samples (dashed).

CHAPTER 5

Scalability and Informed Priors

While many approaches for Bayesian inference have been proposed, scalability for combating the curse of dimensionality remains a major challenge. The first section of this chapter introduces a novel, general-purpose approach, called partially Bayesian neural networks (pBNN), in which a layer from a neural network is subselected to be Bayesian, in order to reduce the dimensionality the Bayesian inference problem while still capturing the predictive uncertainty similar to a fully Bayesian neural network.

Additionally, in formulating BNNs, there remains an open question of how to select priors in the abstract neural network weight space. The second section of this chapter proposes a novel method for selecting priors based on favorable regularization effects and applying a final transform layer in the neural network such that the prior push-forward distribution captures intuitive information about the physical quantity being predicted.

5.1 Partially Bayesian Neural Networks

This section introduces a novel method to address the scalability challenges of building million-parameter BNNs, by performing *targeted* Bayesian inference on a strategically selected portion of the overall DNN, referred to as a partially Bayesian neural network (pBNN) throughout this dissertation. This is joint work with Snehal Prabhudesai, and is published in *Frontiers in Computer Science* [135].

5.1.1 A Heuristic Procedure for Layer Selection in pBNN

The overall procedure is presented in three main steps (see Figure 5.1): (1) access a deterministic DNN model, (2) conduct sensitivity analysis (SA) to select a DNN layer for Bayesian inference, and (3) perform targeted Bayesian inference on the selected layer using Flipout VI to arrive at the final pBNN. Each step is described below in detail.

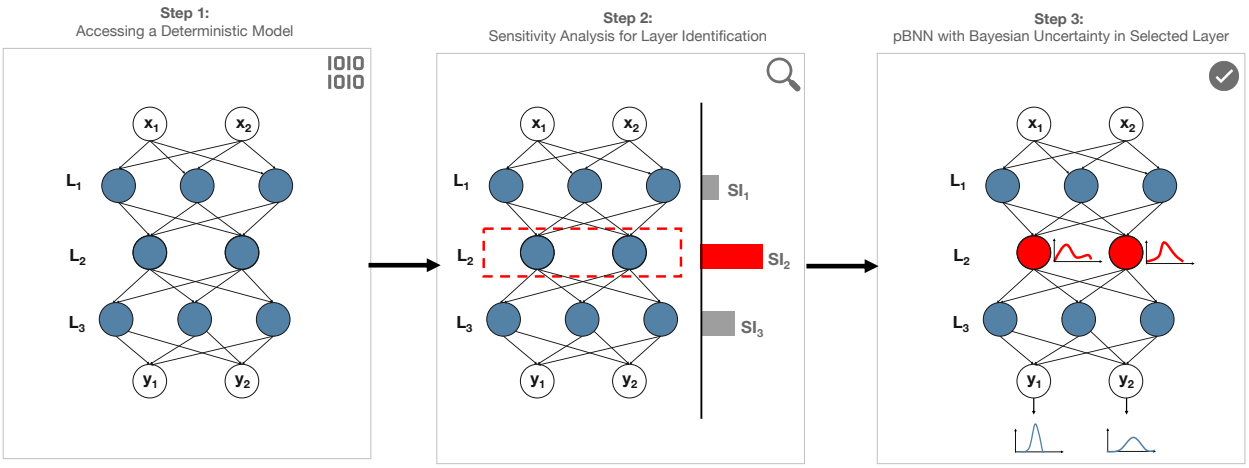


Figure 5.1: Flowchart to address scalability challenges of BNN by building pBNN: Step (1) represents a million-parameter deterministic NN. Step (2) Conduct Sensitivity Analysis to identify the layer that has greatest impact on model output. Step (3) Conduct Bayesian inference on the most sensitive layer to build pBNN. The resulting pBNN combines the strengths of both deterministic and Bayesian NNs, allowing scalability.

5.1.1.1 Step 1: Access a Deterministic DNN Model

The first step is to access a deterministic DNN model (i.e. one that is trained by minimizing the loss in Eqn. 6.1). The purpose of this DNN is to provide an inexpensive and meaningful starting point for the upcoming SA and layer selection. One scenario is that such a deterministic DNN is already available from a pre-existing study or another research group, and can simply be inherited. If it does not exist yet, a new model may be trained in the typical non-Bayesian (frequentist) manner, which is quite inexpensive relative to a full BNN training procedure.

Since this deterministic DNN will be used to guide the selection of model layer for Bayesian inference, it must have the same architecture as the DNN that will eventually undergo Bayesian inference. However, the precise training setup (e.g., choice of loss function, learning rate) for creating this deterministic DNN is flexible since the goal in the next step is to seek a general sense of the sensitivity behavior. This flexibility is important in practice, for instance in situations where a pre-existing deterministic model is available but without information on how it was trained.

5.1.1.2 Step 2: Conduct Sensitivity Analysis to Select a Layer for Bayesian Inference

Next, SA is performed on the deterministic DNN from Step 1 in order to assess the model prediction behavior as a result of model parameter variation. Gradient-based sensitivity is adopted for its simplicity, although other types of sensitivity (e.g., sample-variance-based sensitivity) may also

be used. Specifically, the partial gradient of model prediction output is taken with respect to the parameters of the candidate layer being considered (e.g., ℓth layer), which can be calculated easily from a DNN using only a single pass of back-propagation. Since this partial gradient is a vector, the L2-norm of the partial gradients are taken to transform them into a scalar metric, though other norms such as the L1-norm may be used as well. Lastly, to avoid automatic favoring of larger layers due to simply having more parameters (and therefore more entries contributing to the gradient vector), the gradient scalar is normalized (divided) by the number of parameters in layer ℓ (N_ℓ) to arrive at the *average sensitivity per parameter*. my overall Sensitivity Index for layer ℓ is:

$$\mathrm{SI}(\ell) = \frac{1}{N_\ell} \left\| \nabla_{w_\ell} \hat{y} \right\|_2. \tag{5.1}$$

The layer with the highest SI, $\ell^* = \arg\max_\ell \mathrm{SI}(\ell)$, would then induce the largest change in prediction $\hat{y}$ (relative to the number of parameters) when its parameters w_ℓ are perturbed. This layer thus presents as the most critical layer, justifying to focus my resources to capture the Bayesian uncertainty for layer ℓ^*. Layer ℓ^* will then undergo Bayesian inference in the next step.

Note that gradient is a local operator and captures sensitivity locally at the optimized parameter values of the deterministic DNN (i.e. from Eqn. 6.1). To capture sensitivity more globally across a wider range of possible w_ℓ values, one may also consider averaging the gradient from multiple locations of w_ℓ, for instance in estimating the prior-expectation of the gradient: $\mathbb{E}_{w_\ell}\left[\frac{1}{N_\ell}\left\| \nabla_{w_\ell} \hat{y} \right\|_2\right]$. However, these global measures are more expensive to compute.

5.1.1.3 Step 3: Perform Targeted Bayesian Inference to Build pBNN

In the last step, targeted Bayesian inference is performed on the identified layer ℓ^* to build the pBNN. Formally, w is partitioned into two subsets: $w = \{w_B, w_D\}$ where w_B are the DNN parameters corresponding to layer ℓ^* that will undergo Bayesian inference, and w_D consists of all remainder parameters that are treated deterministically (i.e. not as random variables). This sets up for a pBNN where only a strategically chosen portion (layer) is Bayesianized.

At this point, w_D may simply be frozen at the deterministic DNN values w_D^* (from Step 1), and VI is then used to compute the *conditional* posterior:

$$\theta_B^* = \arg\min_{\theta_B} \; D_{\mathrm{KL}}\left[q(w_B;\theta_B) \parallel p(w_B|w_D^*, x_T, y_T)\right]. \tag{5.2}$$

However, allowing w_D to change may allow additional improvement in approximating the posterior. Therefore, it is proposed to jointly optimize w_D (the deterministic parameters) and θ_B (parameters

of the variational posterior for the Bayesian DNN parameters)

$$\{\theta_B^*, w_D^*\} = \underset{\theta_B, w_D}{\arg\min}\ D_{\mathrm{KL}}\left[q(w_B;\theta_B) \parallel p(w_B|w_D, x_T, y_T)\right]. \tag{5.3}$$

Lastly, Equation (5.3) is solved numerically using an appropriate VI algorithm.

5.1.2 Demonstration

Before applying the pBNN to the tumor segmentation DNN, this work first demonstrate my framework on a test problem using a small densely-connected DNN with synthetic training data in order to bring intuitions and insights.

To set up the problem, this work generate 256 synthetic training data points following the example from [136] with some modifications:

$$y_i = x_i + 0.3\sin(2\pi x + \epsilon_i) + 0.3\sin(4\pi x + \epsilon_i) + \epsilon_i \tag{5.4}$$

where $\epsilon_i \sim \mathcal{N}(0, 0.02^2)$.

I fit the data to a DNN comprised of an input layer, 9 hidden dense layers with swish activation, and a dense output layer with linear activation (see Table 7.1). The dense layers varied in width to mimic the varying sizes of convolutional layers encountered in tumor segmentation DNNs. The DNN has a total of 13,385 trainable parameters.

Layer	Tunable Parameters
Input	0
Dense 1	4
Dense 2	12
Dense 3	40
Dense 4	144
Dense 5	544
Dense 6	2112
Dense 7	8320
Dense 8	2064
Dense 9	136
Output	9

Table 5.1: Architecture of the densely-connected DNN used for the test problem.

In **Step 1** of the pBNN procedure, this work produced a deterministic DNN by minimize the loss in Eqn. 6.1, with the result plotted as the solid orange line in Fig. 5.3. In **Step 2**, this work computed the gradient-based SI in Eqn. 5.1 for each of the 10 candidate layers, shown in Fig. 5.2.

The bar graph indicates generally lower SI for the middle layers compared to those near the input or output; the lowest SI is at Layer 7, and the highest SI is at Layer 9. Layer 9 is therefore selected for Bayesianization. In **Step 3**, this work construct a pBNN by performing Bayesian inference on Layer 9 using Flipout VI and training all other layers deterministically, following Eqn. 5.3. this work provide more details of the pBNN implementation and validation below.

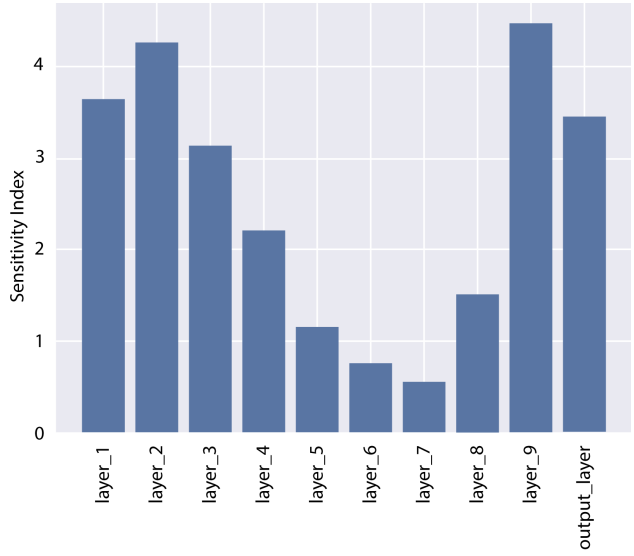


Figure 5.2: SI for all layers in the DNN for the test problem. Layer 9 has the highest SI and is selected for Bayesianization.

My pBNN is implemented using TensorFlow Probability (TFP) [137], where Layer 9 is replaced with a DenseFlipout layer and all other layers remaining to be Dense. Training is then performed simultaneously on θ_B and w_D as in Eqn. 5.3. For the Bayesianized DNN parameters (w_B), this work adopt independent Gaussian priors with a rather wide standard deviation to reflect an initial uninformative distribution: $p(w_B) \sim \mathcal{N}(0, 10^2)$. A Gaussian likelihood is also used to depict independent Gaussian observation noise ($\sigma_\epsilon = 0.02$ to be consistent with the data-generation in Eqn. 5.4) on the y targets:

$$p(y_T|x_T, w_D, w_B) = \prod_{n=1}^{N_T} \frac{1}{\sigma_\epsilon\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y_n - f(x_n; w_D, w_B)}{\sigma_\epsilon}\right)^2\right]. \tag{5.5}$$

The variational posteriors are from the family of independent Gaussians $q_i(w_B; \theta_B) \sim \mathcal{N}(\mu_i, \sigma_i^2)$, and so the variational parameters are $\theta = \{\mu_i, \sigma_i\}$. I train this pBNN using the Nadam [138] optimizer for 10^4 epochs with a learning rate of 10^{-3} and batch size of 64.

To assess whether the pBNN constructed with SA-selected Layer 9 provides a good approximation to the uncertainty of the full BNN, this work also train a full BNN where all layers are

replaced by DenseFlipout layers; the posterior push-forward uncertainty[1] of the full BNN is shown in Fig. 5.3. Furthermore, to investigate the performance of pBNNs if a layer different from Layer 9 were selected, this work construct separate pBNNs where a different layer is Bayesianized.

Since a good pBNN is one that faithfully approximates the uncertainty in the full BNN, this work measure its performance based on how close the pBNN's posterior push-forward is to the full BNN's posterior push-forward via the KL divergence. Note the caveat that a close posterior push-forward does not necessarily imply a close posterior in general. However, the posterior pushforward, which is much lower dimensional than the posterior can act as a proxy to assess the closeness of the overall Bayesian uncertainty. Following the same reasoning as the SI from Eqn. 5.1, this work normalize this quantity by multiplying N_ℓ (since lower KL is better), to arrive at my validation metric:

$$D_{\text{norm}} = \frac{N_\ell}{N_{tot}} D_{\text{KL}}[f(x; w_D^*, w_{\text{partial},B}) \,||\, f(x; w_{\text{full}}))], \tag{5.6}$$

where $w_{\text{partial},B} \sim q_{\text{partial}}(w_{\text{partial},B}; \theta_B^*)$ are the Bayesianized parameters in the pBNN, and $w_{\text{full}} \sim q_{\text{full}}(w_{\text{full}}, \theta^*)$ are the full set of parameters (all Bayesianized) in the full BNN. The D_{norm} values for each pBNN are plotted against SI in Figure 5.4, where this work observe a Spearman's correlation of $r_s = -0.83$ which is considered to be strongly correlated in literature [139]. This supports my SA-based layer selection strategy, where the layers with high SI will also lead to low D_{norm}. Lastly, note that while this work performs this brute-force comparison to validate the effectiveness of my approach, in practice one would only train the pBNN on the SA-selected layer as described in Section 5.1.

[1]Posterior push-forward is $p(f(x; w_B, w_D)|x, x_T, y_T)$, which differs from the posterior predictive $p(y|x, x_T, y_T)$.

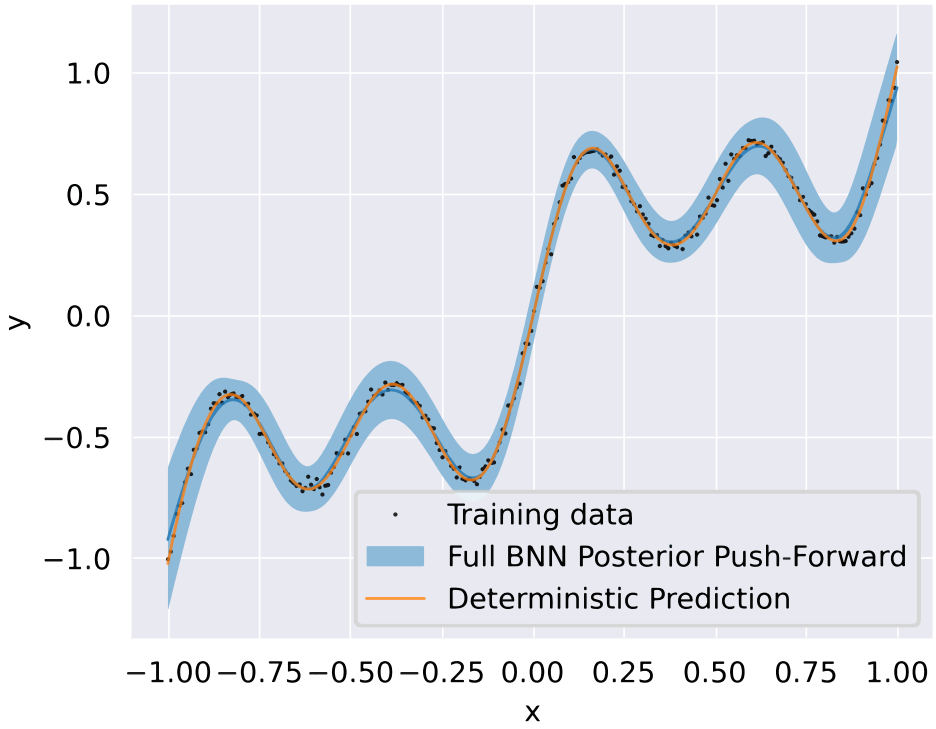


Figure 5.3: Training data (black dots) overlaid with the determinisitic DNN prediction (solid orange) and full BNN posterior push-forward uncertainty (solid blue line is the mean, blue shadow is the 95% credible interval).

Figure 5.4: Normalized KL divergence from the full BNN posterior push-forward to the different pBNNs is highly correlated with the SI that is used to select the layer for Bayesianization (Spearman's correlation $r_s = -0.83$).

5.2 Predictive-Informed Prior Selection

Briefly, there are different Bayesian philosophies that emphasize different ways of viewing a prior distribution. Subjective Bayes adheres to the framework that ones understanding of a system is inherently subjective and specific each observer. Different observers might have different understandings of a system and might select different, but equally valid priors. Objective Bayes takes issue with this ambiguous interpretation, preferring pragmatic priors with regularizing attributes or deliberately uninformative priors when no natural prior exists. One challenge of Bayesian inference on the highly abstract parameter space of a deep neural network is that there is very little intuition as to what a prior distribution should be, particularly in the subjective sense.

One solution might be to first train the weights of a BNN such that predictive distributions directly match the desired prior predictive distributions that encode expert opinions or physical intuition of the trends in a given dataset, however this solution highly nonunique and would likely induce very poor model performance due to highly atypical regularization. However, from a model regularization and performance perspective, certain priors can still be useful even if the distributions are not physically intuitive. Zero-mean Laplace priors, analogous to L_1 regularization in frequentist settings, is useful encouraging model parsimony. Zero-mean Gaussian priors, analogous to L_2 regularization in frequentist settings, is useful for avoiding model overfitting. When used together as a joint Laplace-Gaussian distribution, they induce an elastic net regularization. In the literature, many works on Bayesian neural networks simply assign uninformative Gaussian priors and avoid the problem of appropriate prior selection altogether, regardless of Objective or Subjective philosophy.

This work proposes a justification for BNN prior selection that incorporates aspects of both Subjective and Objective philosophies. This prior selection justification is done via a validation or cross validation scheme, analogous to frequentist regularization parameter tuning. Deterministic models can be affordably trained via maximum *a posteriori* (MAP) estimates based on a selection of different priors along with the likelihood function above. The MAP estimates on model parameters are then used for producing model predictions on a held-out validation set. Final priors are selected based on relevant model performance metrics, such as mean squared error, on the validation set. This is different from the so-called "inverse crime" of choosing a prior based on knowledge of a posterior distribution. Instead, this can be framed as a selection of hyperpriors in a hierarchical Bayesian formulation, similar to [140]. However, these hyperpriors are inferred via a maximum likelihood method, to avoid the additional cost of a fully Bayesian inference are every hierarchical stage. Following the flow of [140] (with some modified notation), w are the parameters of the neural network model and σ are the parameters of the prior distributions, for example the scale

parameters of a zero-mean Gaussian or Laplacian distribution:

$$w \sim p(w|\sigma) \tag{5.7}$$

$$\sigma \sim p(\sigma). \tag{5.8}$$

The joint distribution of both the weight prior and scale hyperpriors is therefor

$$p(w, \sigma) = p(w|\sigma)p(\sigma), \tag{5.9}$$

and the posterior distribution is therefor

$$p(w, \sigma|\mathcal{D}) = p(w|\sigma, \mathcal{D})p(\sigma|\mathcal{D}). \tag{5.10}$$

With this formulation, σ could be learned via Bayesian inference, though without simplifying assumptions would be very costly. Instead, I propose using a cross-validation scheme to select the best hyperprior $\hat{\sigma}$

$$p(w|\sigma, \mathcal{D})p(\sigma|\mathcal{D}) \approx p(w|\hat{\sigma}, \mathcal{D}). \tag{5.11}$$

This selection of priors via cross-validation is very similar to the standard approach of selecting hyperparameters in a frequentist setting.

One problem with this approach is that the priors in the abstract parameter space of the model might have unforeseen influence on model predictions. This phenomenon is illustrated in Figure 5.5, where three neural networks, each with identical architectures but different prior distributions yield very different (likely undesirable) prior predictive distributions.

This can be addressed by using transforms on the model output in order to map outputs from the untrained "prior model" to a physically intuitive "prior predictive" distribution. These transforms are calculated by repeatedly sampling a prior distribution of the model and then observing the output from the final layer with a linear activation function. An empirical PDF can be calculated for this untransformed output function. From this PDF, the CDF can be calculated. It is then trivial to use the CDF to transform the prior predictive to a uniform distribution, i.e. $U(0, 1)$ and from here, transforming to an arbitrary prior predictive. This custom prior predictive distribution might be uninformative– for example a uniform distribution across some finite support prediction domain. Or the custom prior predictive might be informative– for example a Gaussian distribution designed to approximate the sample mean and variance of the quantity that we are predicting. This process yields an intuitive prior prediction while leveraging (and without modifying) the regularizing properties of the prior distribution on the model parameters. This process is detailed below in Algorithm 1 and is implemented and further explored in Chapter 7.

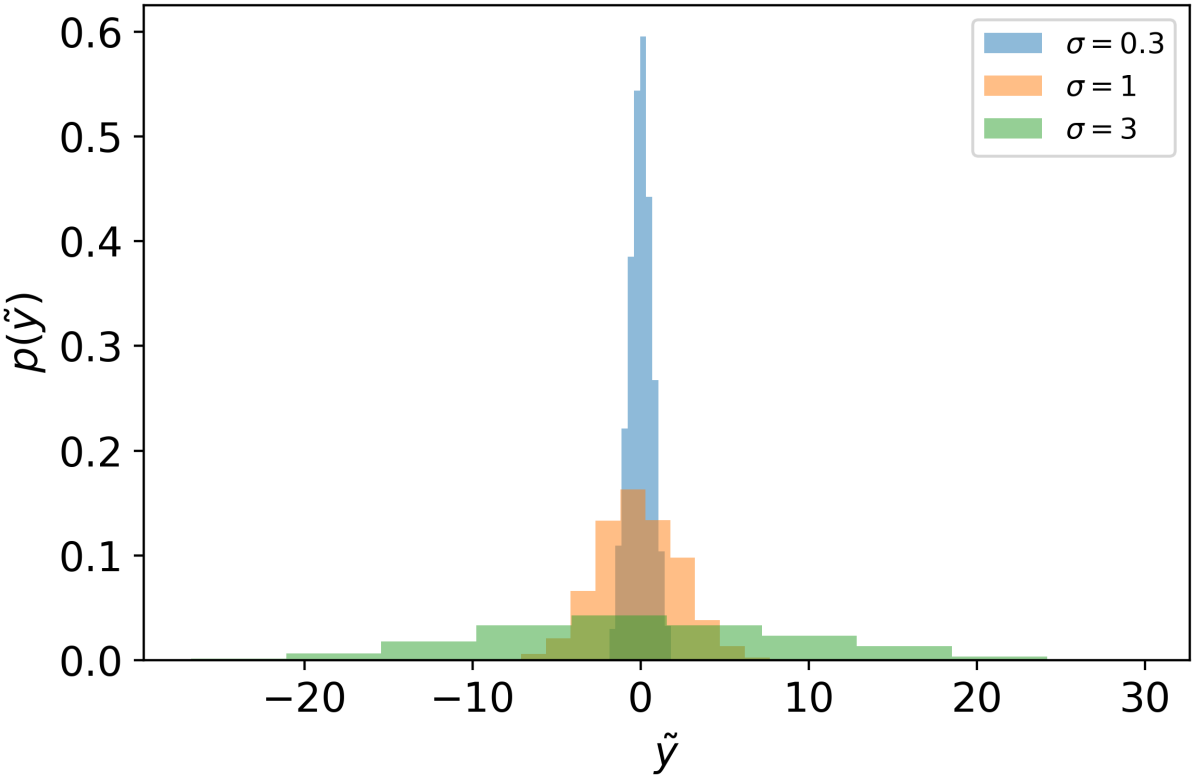


Figure 5.5: Predictive distributions $p(\tilde{y})$ are shown for 3 BNN with identical fully-connected architectures and data but featuring zero-centered Gaussian prior distributions with different scale parameters.

Algorithm 1 Pseudo code for the prior selection process

Input : model architecture h with linear output, paired training feature and label pairs $\{X_T, y_T\}$, parameterized priors for prior selection (e.g. σ_0 for a zero-mean Gaussian prior, b_0 for a zero-mean Laplace prior, $[\sigma_0, \sigma_1, ..., \sigma_k]_0$ and $[\alpha_0, \alpha_1, ..., \alpha_k]$ for a mixture of k zero-mean Gaussian distributions, etc).

Output : A set of validation or cross-validation performance metrics for each combination of prior parameters.

for each combination of prior parameters, e.g. instances of σ_0, b_0 **do** :

  for iteration n of N total MC samples **do**:

    Sample $w_{0,n} \sim w_0$

    Propagate $w_{0,n}$ through $h(w_{0,n}|X_T, y_T)$ to sample $\tilde{y}_{0,n}$

  end

  $U = \text{CDF}([\tilde{y}_0]_{1:N})$ calculated empirically or by Gaussian approximation of $[\tilde{y}_0]_{1:N}$

  $f(w|X_T) = T(U(h(w|X_T)))$, where T is a transform of $\mathcal{U}(0,1)$ to some user-defined prior-predictive

  $w_{MAP} = \arg\min_w -\log p(w|X_T, y_T)$ train the MAP model with appropriate gradient optimization

  $\hat{y}_{MAP,val} = f(w_{MAP,X_{val}})$ calculate forward-pass validation predictions

  $\mathcal{M}(y_{val}, \hat{y}_{MAP,val})$ assess relevant validation metrics $\mathcal{M}$ on label and prediction

end

5.3 Conclusion and Future Work

This chapter introduces the Bayesian inference frameworks and evaluates several algorithmic approaches to numerical Bayesian inference. MCMC algorithms have historically been the workhorse of numerical Bayesian inference, but even advanced algorithms such as HMC struggle to scale to the dimensionality demanded by modern SciML tasks. VI is a more recent algorithm that is well suited for implementation in BNNs, though it struggles to adequately approximate multi-modal and non-Gaussian posterior distributions and typically underestimates posterior variance. SVGD avoids familial approximation and offers greater flexibility; pSVGD expands upon SVGD to provide greater scalability, particularly with datasets that make consist of spatially or temporally discretized data. Despite these algorithmic advances, Bayesian inference is far more computationally demanding that traditional maximum likelihood (or sometimes maximum *a posteriori*) estimates of NN parameters.

To partially address this issue of computational cost, this chapter introduces a novel heuristic for subselecting BNN parameters for Bayesian inference and therefor reducing the computational expense. The relative influence of a given BNN layer, measured via a sensitivity index, is highly correlated to the similarity that a pBNN based on that layer has to a hypothetical full BNN.

Furthermore, in the literature there is generally a lack of consensus on how to select highly abstract prior weight distributions in a BNN. This work does not provide a definitive answer to this dilemma, but rather provides a novel justification to a prior selection methodology that leverages Bayesian analogues to frequentist hyperparameter selection that is ubiquitous in the field of deep learning.

With the pBNN approach, open questions still remain regarding how exactly to choose which subset of neural network parameters. Choosing a single layer of parameters based on the sensitivity index is a simple and code-friendly approach, but selecting partial layers (even individual parameters) or multiple layers would likely give greater flexibility to focus computational power on the most impact parameters. Future work will evaluate whether simple strategies based on the sensitivity index are sufficient to reasonably predict the faithfulness to the far more computationally challenging full posterior.

With the predictive-informed weight prior approach, there remain open questions on how this approach impacts model performance. Furthermore, future research will try to expand the CDF transform technique beyond 1D transforms, toward a more generalized multidimensional approach.

CHAPTER 6

Partially Bayesian Neural Networks for Transparent Medical Imaging AI

This chapter discusses a computationally-efficient UQ approach– partially Bayesian neural networks (pBNN)– to segment tumors on MRI scans using state-of-the-art neural architectures. In pBNN, only a single layer, strategically selected based on gradient-based sensitivity analysis, is targeted for Bayesian inference. This work illustrates the effectiveness of pBNN in capturing the full uncertainty for a 7.8-million parameter U-Net. This work also demonstrates how practitioners and model developers can use the pBNN's predictions to better understand the model's capabilities and behavior. The work presented in this chapter was the result of close collaboration with Snehal Prabhudesai, Dingkun Guo, Arvind Rao, Nikola Banovic, and Xun Huan and is published in *Frontiers in Computer Science* [135]. While I preserve the majority of the paper to convey its comprehensive narrative, I would like to acknowledge that in the work presented in this chapter, Snehal Prabhudesai was chiefly responsible for providing the background and making the connection to the medical application domain, interpreting uncertainty maps, and contributed in training pBNN models and aggregating performance metrics.

6.1 Introduction

Clinical Decision Support Systems (CDSS) based on Artificial Intelligence (AI) are promising to assist clinicians in providing better patient care in high-stakes medical decision-making [141], including for brain tumors. AI-based CDSS have shown potential to accurately segment tumors [142, 143], which could aid in treatment planning for patients with glioma [144], a malignant manifestation of brain tumors [145, 146]. AI models such as Deep Neural Networks (DNNs) can learn patterns from radiologist-annotated multi-modal Magnetic Resonance Imaging (MRI) scans for tumor delineation with high accuracy [4]. These optimistic results have contributed to accelerating approvals from governing agencies such as the U.S. Food and Drug Administration (FDA) [147], with the goal to integrate them into clinical workflows [148].

In the domain of radiology, AI-based CDSS have shown potential to assist in treatment planning, which requires precise segmentation to ensure that the treatment is effective in resecting the bulk of the tumor, without unnecessarily affecting functional brain areas. Radiologists rely on their years of training and practice to estimate the extent of tumor location and spread as precisely as possible. Radiologists delineate the tumor based on its appearance on the MRI, and mark additional boundaries around it to indicate certainty about their segmentation, or lack thereof [149]. Recently, researchers have used DNNs [150, 151, 152, 4] to automate tumor segmentation [153] for a large cohort of patients at a fraction of the time it would take to do it manually.

However, these algorithms do not communicate uncertainty, which remains a key barrier to their responsible adoption in clinical workflows [154]. AI-based CDSS that produce single-valued (i.e. deterministic) predictions do not reflect the inherent uncertainty in medicine [46], and encmyage impressions of "superhuman" ability of AI [47] without mechanisms to contest these claims [48]. This can exacerbate problems with inappropriate trust and reliance on AI-based CDSS [155], with potential harm to patient health [156]. Despite calls in Transparent and Responsible AI [157, 158] to quantify [159, 160], report [161, 162, 163] and design for uncertainty [164], little attention has been dedicated to address uncertainty effects in AI models for healthcare [48, 154]. This lack of model transparency remains a key barrier to the responsible adoption of AI in clinical practice [165, 166, 167].

Uncertainty Quantification (UQ) for AI-based CDSS [168] remains challenging due to their high computational costs [169]. For example, Bayesian analysis [170, 171, 172] provides a principled and systematic approach to quantify and update the uncertainty of model parameters, but typically is solved via expensive iterative algorithms such as Markov chain Monte Carlo (MCMC) [25, 26, 27, 28] or variational inference (VI) [33, 34, 35]. While Bayesian methods have been historically used for UQ in medicine [173], they are difficult to scale for handling e.g., DNNs in AI-based CDSS that often involve millions of parameters [174, 175, 176].

This work seeks to enable Bayesian UQ for million-parameter DNNs, i.e. to create million-parameter Bayesian neural networks (BNNs) [136, 177]. This is accomplished by introducing a low-computation strategy for performing Bayesian inference on a small, strategically chosen portion (layer) of the entire DNN, thereby creating a *partially* BNN (pBNN) that approximates the uncertainty of a full BNN. The Flipout [63] VI algorithm is used to solve the resulting Bayesian problem on the target layer. Unlike related strategies [178, 179, 180] that solely attempt Bayesian inference only on the last layer of a DNN, which does not necessarily offer the best uncertainty representation [181], this approach allows a justified selection on which part of the DNN to "Bayesianize" guided by sensitivity analysis (SA).

To impart the ability to express uncertainty in tumor segmentation models, pBNN are built using a state-of-the-art 7.8 million-parameter U-Net architecture [4] proposed for medical image

segmentation. To empirically validate the approach, a full BNN is trained as a baseline for the uncertainty approximation discrepancy of the pBNNs to this full BNN. This experiment demonstrates that the pBNN based on the SA-selected layer is the most efficient in approximating the full Bayesian uncertainty (per Bayesianized parameter) compared to other layer choices. Results suggest that the SA-based layer-selection offers an effective and inexpensive way to identify the best DNN layer to perform Bayesian inference.

The main contribution of this work is to enable Bayesian UQ for million-dimensional medical DNN models. It features a novel computational method that uses a SA-based DNN layer selection for targeted Bayesian inference, which is especially useful in scenarios where practitioners and model developers want to understand the model's uncertainty but do not have the resources for a full Bayesian inference on the entire DNN. This work advances future interactions for transparent and responsible AI that can support the investigation of the uncertainty in DNN models. This work illustrates one such interaction, through the use of uncertainty maps, to show how clinicians could be enabled to interpret uncertainty of the tumor segmentation AI.

6.2 Methodology

6.2.1 Bayesian Neural Networks

A DNN takes input x and predicts output $\hat{y}$ and can be written as $\hat{y} = f(x; w)$ where w represent all tunable model parameters of the DNN (e.g., DNN weights and bias). Given N_T training data points $(x_T, y_T) = \{x_n, y_n\}_{n=1}^{N_T}$, the DNN is typically trained by finding w to minimize a loss function:

$$w^* = \arg\min_{w} \ \mathcal{L}(w, x_T, y_T). \tag{6.1}$$

For example, a popular choice is the least squares loss $\mathcal{L}(w, x_T, y_T) = \frac{1}{N_T} \sum_{n=1}^{N_T} [f(x_n; w) - y_n]^2$. The optimization is often done with stochastic gradient descent [94, 95]. Once Eqn. (6.1) is solved, it produces a deterministic DNN that makes single-valued prediction for any new input x: $\hat{y} = f(x; w^*)$.

BNN [174, 175, 176, 136, 177], in contrast, treats w as random variables with an associated probability density function (PDF) that represents the uncertainty on w. When training data become available, these PDFs are updated through Bayes' rule:

$$p(w|x_T, y_T) = \frac{p(y_T|x_T, w)p(w)}{p(y_T|x_T)}, \tag{6.2}$$

where $p(w)$ is the prior PDF[1], $p(y_T|x_T, w)$ is the likelihood PDF, $p(w|x_T, y_T)$ is the posterior PDF, and $p(y_T|x_T)$ is the marginal likelihood (a PDF-normalization term). The prior thus represents the uncertainty on w before seeing any training data, and the posterior describes the updated uncertainty after incorporating the training data. Solving the Bayesian inference problem then entails computing the posterior $p(w|x_T, y_T)$. Both DNNs and BNNs are further elaborated upon in Section 2.2.5.

6.2.2 Partially Bayesian Neural Networks

This section presents the main demonstration of the pBNN, applying it to a state-of-the-art DNN model used for medical image segmentation: the U-Net [4]. Since the goal is to demonstrate the pBNN framework presented in Section 5.1 and not to develop or justify the choice of architecture, the U-Net architecture is taken *as given*.

6.2.2.1 Step 1: Access a Deterministic DNN Model

First, a U-Net model is trained deterministically using the architecture and code published by [182]. As shown in Fig. 6.1, the U-Net comprises of an encoder, decoder, and skip connections. The encoder module composes of 2D Convolutional layers followed by MaxPooling layers with window size 2×2, and 2D Spatial Dropout layers for regularization. The decoder module is composed of corresponding Conv2DTranspose layers followed by Concatenate layers. Nested in between the encoder and decoder is the 2D Upsampling layer with bilinear interpolation. The U-Net has feature map size set to 32, kernel size 3×3, and dropout rate 0.2. For the 2D Convolutional layers, ReLU activation is used along with He uniform variance scaling initializer, and kernel size is set to 2×2. Overall, this U-Net has a total of 7.8 million tunable parameters.

Training data is taken from the Medical Segmentation Decathlon Challenge Task 01 [183] consisting of brain MRI scans of patients with glioma. The training set contains of 58,464 2D images (each sized 144×144) and a separate validation set contains 6,624 images. Data augmentation is employed during training to increase data diversity including random choices of horizontal and vertical flips, $\pm 45°$ rotation, and $\pm 2.5°$ shearing. The ADAM optimizer [184] is run for 30 epochs used a learning rate 0.001 and a mini-batch size of 128. The training loss function is a weighted sum of the Dice coefficient [185, 186] and standard Binary Cross Entropy. The training results are summarized in Table 6.1.

[1]Note that $p(w|x_T) = p(w)$, i.e. the prior uncertainty should not change from knowing only the input values of the training data without their output values.

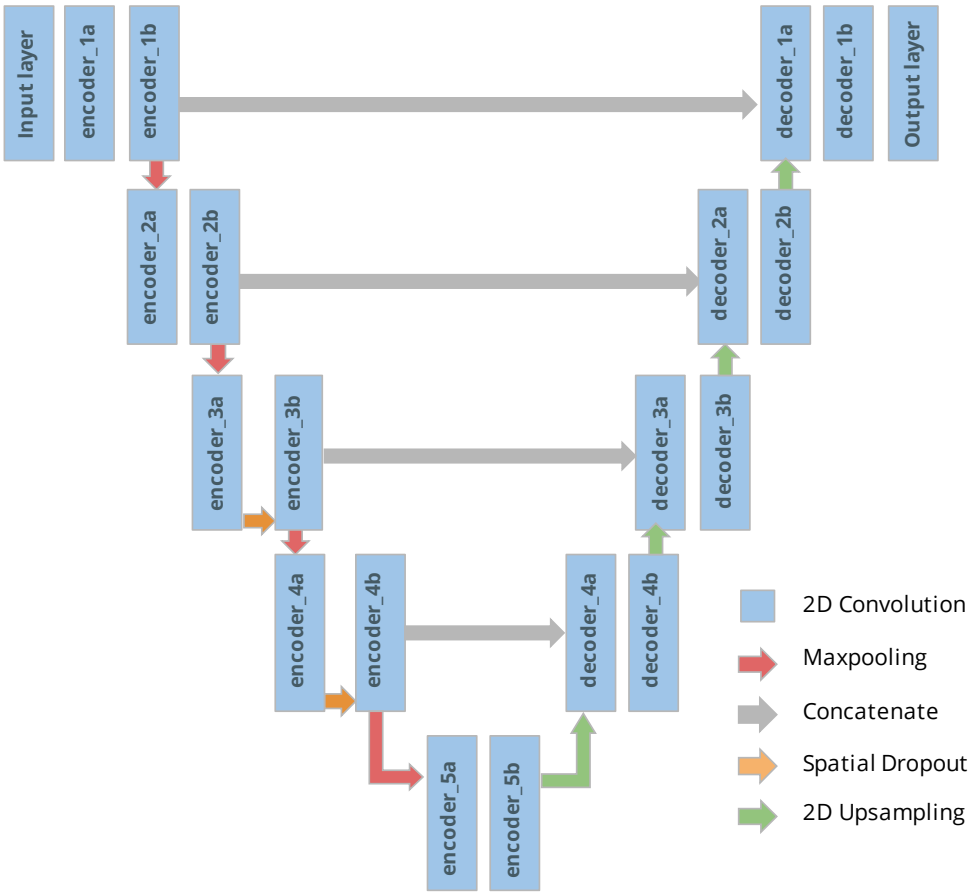


Figure 6.1: U-Net architecture with utilizing the naming system of the different tunable layers considered for the pBNN.

6.2.2.2 Step 2: Conduct Sensitivity Analysis to Select a Layer for Bayesian Inference

Gradient-based SI is computed in Eqn. 5.1 for each candidate layers, shown in Fig. 6.2. In particular, the layers near the top portion of the encoder carry larger SI. This response gradually decreases proceeding further into the model, reaching a minimum near the middle, followed by a light rebound at the decoder layers. The lowest SI occurs at Layer decoder_3a, and the highest SI is at Layer encoder_1a. Layer encoder_1a is therefore selected for Bayesianization.

6.2.2.3 Step 3: Perform Targeted Bayesian Inference to Build pBNN

A pBNN is first constructed by performing Bayesian inference on encoder_1a using Flipout VI and training all other layers deterministically, following Eqn. 5.3. This pBNN is implemented using TensorFlow Probability (TFP) [137] by replacing the 2DConvolution layer of encoder_1a with `tfp.layers.Convolution2DFlipout`. Training is then performed simultaneously on θ_B and w_D as in Eqn. (5.3). For the Bayesianized DNN parameters (w_B), independent Gaussian priors

Performance	Value
Training time	5 hrs
Model size	94.3 Mb
Trainable parameters	7.8 million
Validation accuracy	0.9915
Validation Dice coefficient	0.8508

Table 6.1: Summary of the deterministic U-Net training.

are selected with a rather wide standard deviation to reflect an initial uninformative distribution: $p(w_B) \sim \mathcal{N}(0, 10^2)$. Since the label for each pixel is binary while the model prediction ranges $[0, 1]$, binary crossentropy was selected as the natural likelihood function:

$$p(y_T|x_T, w_D, w_B) = \prod_{n=1}^{N_T} [y_n f(x_n; w_D, w_B) + (1 - y_n)(1 - f(x_n; w_D, w_B))] \,. \tag{6.3}$$

The variational posteriors are from the family of independent Gaussians $q_i(w_B; \theta_B) \sim \mathcal{N}(\mu_i, \sigma_i^2)$, and so the variational parameters are $\theta = \{\mu_i, \sigma_i\}$. The pBNN is trained using the Nadam [138] optimizer for 400 epochs with a learning rate of 10^{-4} and batch size of 196, using 8 GeForce RTX 2080 Ti GPUs in parallel. The ELBO training history of the pBNN is shown in Fig. 6.3 (plot with encoder_1a emphasized in red). While the plot indicates a small rise around 150 epochs due to numerical instabilities with the optimization algorithm, overall the curve steadies out by 350 epochs (note that the y-axis is logarithmic which amplifies the lower-value fluctuations) and appears noticeably flatter compared to the beginning of the optimization.

6.3 Results

This section provides validation for the pBNN procedure by illustrating whether the U-Net pBNN constructed with SA-selected encoder_1a layer provides a good approximation to the uncertainty of the full BNN. To achieve this, a full BNN where Bayesian inference is performed on all layers must be trained as a reference point. Such attempts to obtain the full uncertainty are highly expensive, and is in fact the original motivation for the new pBNN. To clarify- the full BNN is not needed as a part of the regular pBNN procedure described in Section 5.1 but solely to validate this methodology.

A full BNN is trained where all 19 layers of the U-Net are Bayesianized, using the full extent of available computational resources: 2000 epochs using the Nadam [138] optimizer with learning rate 10^{-4} and batch size of 196, 94.72 GB of memory, 8 GPUs in parallel, which required 3 days of continuous computation to arrive at a steadied ELBO. The best-performing model is saved during

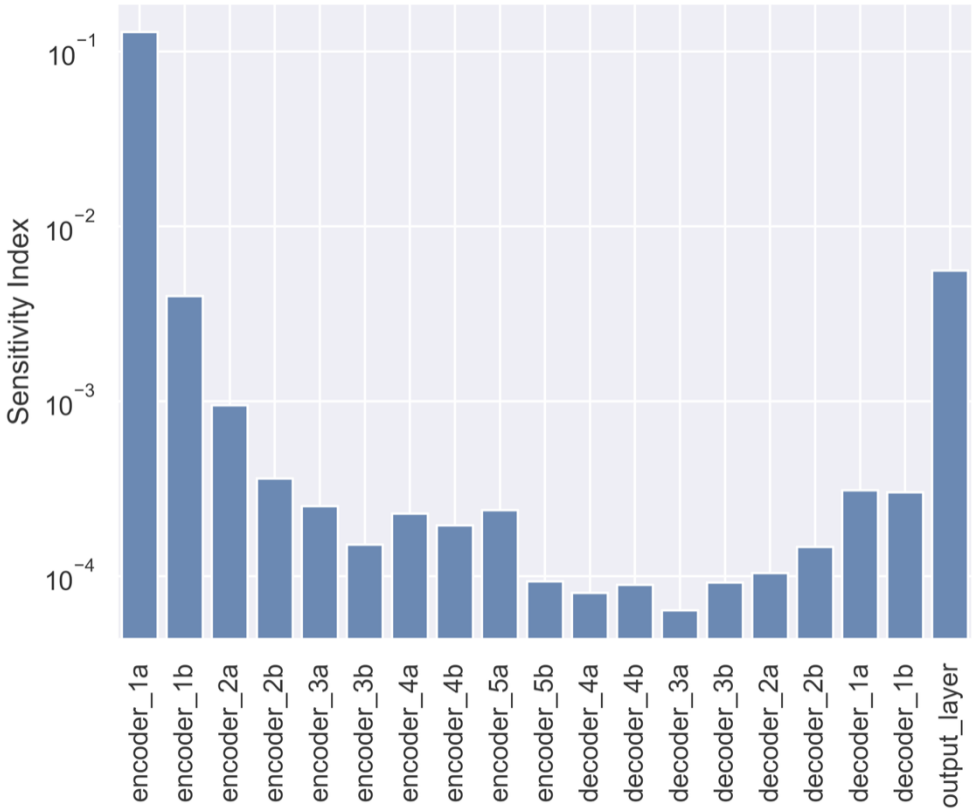


Figure 6.2: SI for all layers in the U-Net used for tumor segmentation. layer_1a has the highest SI and is selected for Bayesianization.

the training process. Due to the tremendous size of this BNN, expert experience is also relied upon for randomized initialization parameters and algorithm tuning to arrive at a reasonably converged full BNN. Additionally, 19 pBNNs are trained, each with one of the 19 layers targeted for Bayesian inference, but with shorter 400 epochs. The ELBO training history for all models are summarized in Fig. 6.3, with the pBNN using the SA-selected layer (encoder_1a) emphasized in red. While the full BNN is trained for significantly longer than the pBNNs, the ELBO is plotted in Figure 6.3 to around 350 epochs for better readability and comparison. All training curves gradually decrease and start flattening around 150 epochs, after which noise dominates over any further convergence trends. The plot suggests that all pBNN models converge reasonably well.

Since a good pBNN is one that faithfully approximates the uncertainty in the full BNN, the pBNN performance is measured using the normalized KL divergence between the pBNN's posterior push-forward to the full BNN's posterior push-forward, D_{norm}, defined in Eqn. 5.6.

With each U-Net output being a 144×144 pixel values, directly computing the KL divergence for the joint 144×144-dimensional posterior push-forward PDF would be very difficult. The computation of D_{norm} is then further simplified through an independence approximation, breaking the

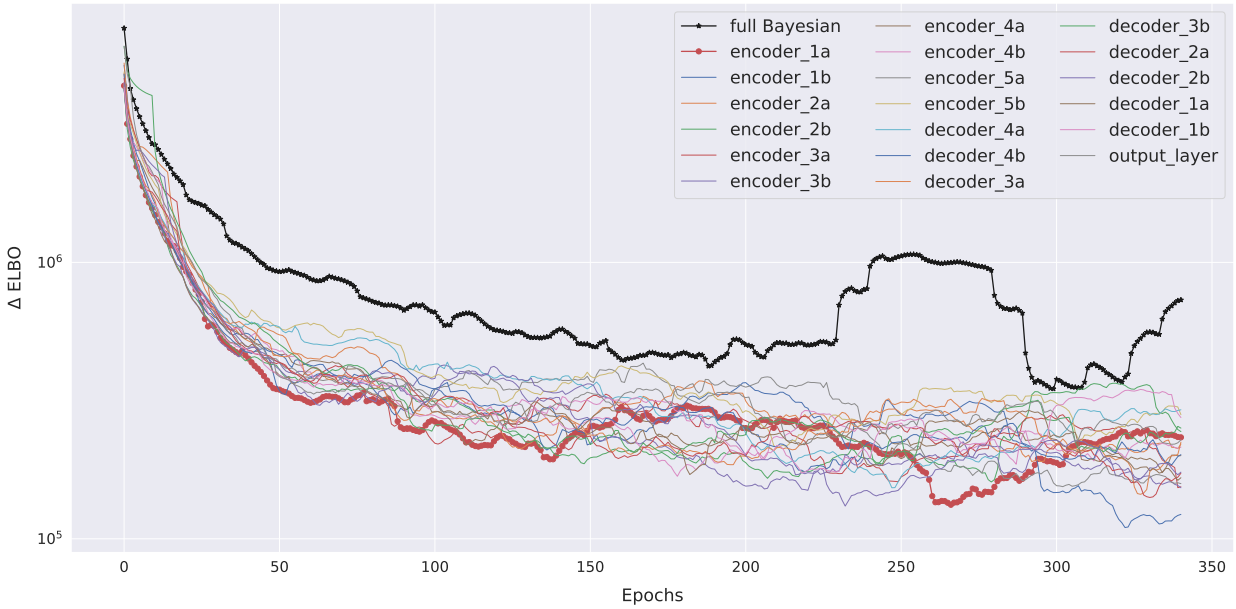


Figure 6.3: ΔELBO training history for all models.

144×144-dimensional KL into 20,736 pixel-wise one-dimensional KL computations. For a given prediction image, each pixel's KL is estimated by first generating 100 samples of the DNN parameters $w_{\text{partial},B}$ from the optimized variational posterior $q_{\text{partial}}(w_{\text{partial},B}; \theta^*_B)$, combine it with the other deterministic parameters and evaluate the corresponding pixel predictions $f(x; w^*_D, w_{\text{partial},B})$, and for each pixel fit its 100 samples to a Beta distributions to approximate its posterior push-forward PDF. Beta distributions are often used to model the uncertainty of Bernoulli distributions and are suitable for this problem to ensure pixel-wise prediction values reside between $[0, 1]$. Once the Beta PDFs are extracted, KL divergence can be calculated analytically. The pixel-wise KL divergence values are then summed to obtain the estimate to the joint PDF's KL value (since the KL divergence between two joint PDFs that are independent factors into the sum of KL divergences of individual marginals). The final D_{norm} are shown in Fig. 6.4, and they show a correlated, inverse trend compared to the SI values in Fig. 6.2, supporting that SI is a good inexpensive indicator of the KL performance metric.

Figure 6.5 shows a scatter plot for all 19 pBNNs the KL divergence (not yet normalized by N_ℓ) versus N_ℓ the number of parameters of the layer targeted for Bayesian inference; the pBNN with the SA-selected layer (encoder_1a) is marked in red. Ideally,the layer selected for Bayesianization would have low KL divergence from the full BNN *and* small number of layer parameters (i.e. towards the bottom-left corner). However, these two desirable properties generally conflict and a tradeoff needs to be made. This can be seen by the Pareto optimality front by the plot's lower-left convext hull. In this case, the SA criterion indeed identifies one of the Pareto points in layer

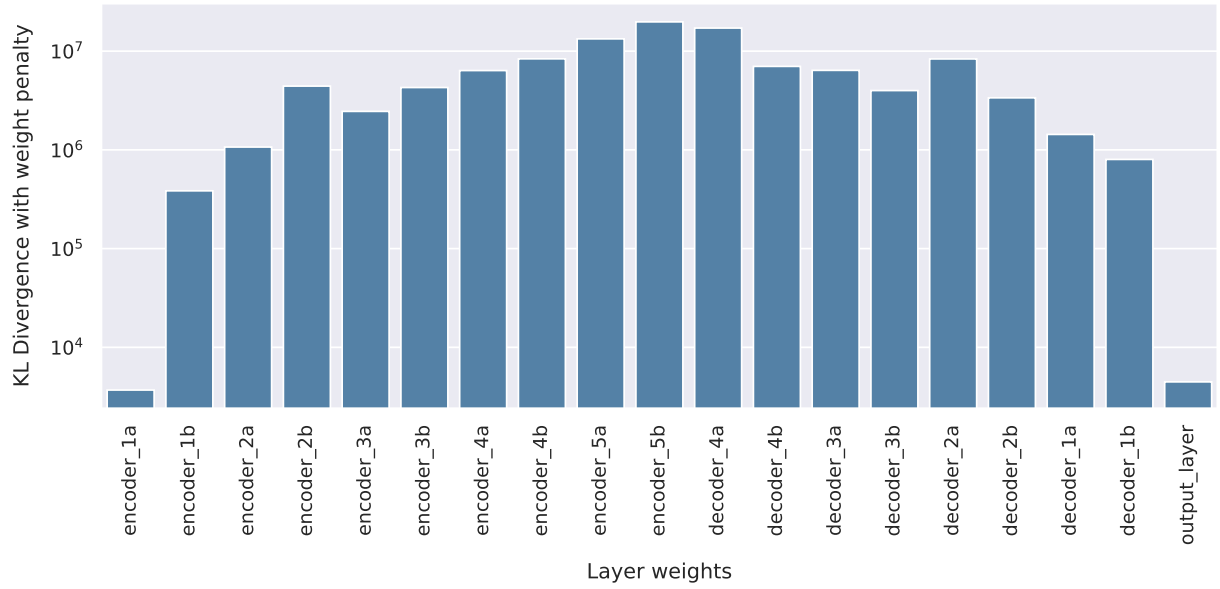


Figure 6.4: Normalized KL D_{norm}, for all 19 pBNNs each built with Bayesian inference on one of the layers in the U-Net.

encoder_1a. The output_layer also resides on the Pareto, and would also serve as a reasonable selection for Bayesianization; indeed, the SI values presented in Fig. 6.2 does show the output_layer to be the runner-up. However it should be noted that the nature of how the two layers are good choices are different: encoder_1a achieves a lower KL than output_layer, but output_layer has a smaller number of parameter.

6.4 Interpreting Uncertainty Maps

Although the main contribution of this work is to provide a scalable method for Bayesian UQ in large DNNs, this section also illustrates one way that uncertainty information could be used in practice. Uncertainty maps are a tool that communicates how confident (uncertain) a model is in its prediction. Fig. 6.6 illustrates these maps for one such example: top row displays the 4 modalities of an MRI scan; bottom row displays the ground truth, prediction, uncertainty and truth-prediction discrepancy respectively for the SA-guided pBNN (Section 6.2.2.3).

6.4.0.1 Construction

Descriptions of how to compute and visualize the uncertainty of the pBNN's prediction for a specific use case are shown in Figure 6.6. This particular example was selected from the test dataset because the ground truth indicated a large percent of the pixels belonging to the tumor

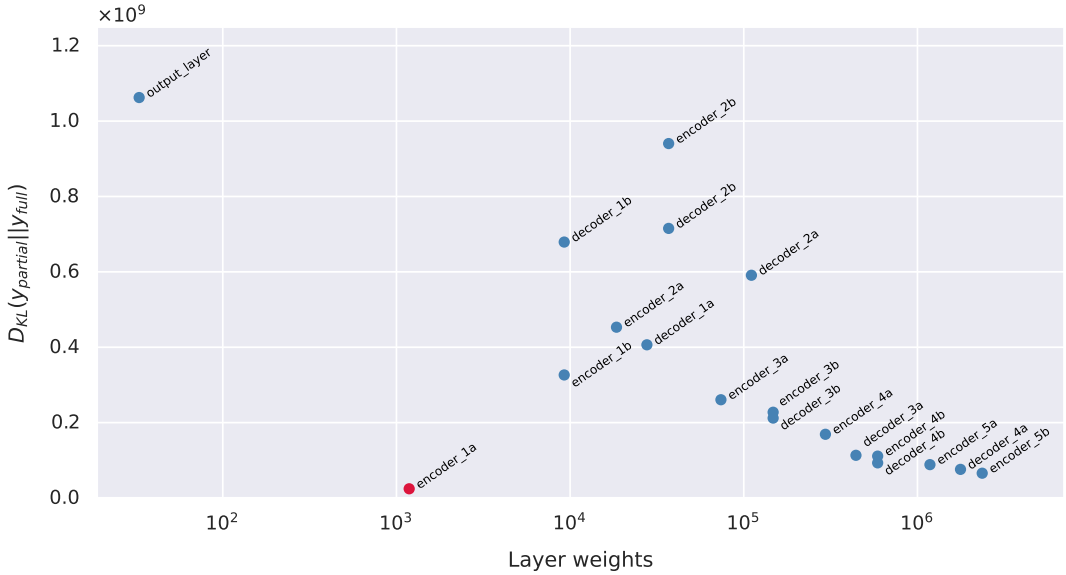


Figure 6.5: Scatter plot of KL divergence sum versus number of parameters in the Bayesian layer.

class (25%). First, 100 samples are drawn from the posterior push-forward distribution of the trained pBNN for this MRI. The binary prediction map is then calculated from the mean of these 100 samples. The continuous values [0,1] are then mapped to their dominant class (tumor or non-tumor) with thresholding (value of 0.5).

The uncertainty map is then constructed from the same 100 samples, by computing per-pixel standard deviation. The range of uncertainty displayed is unique to the image and is not a global uncertainty measure. The prediction map is then subtracted from the ground truth in order to visualize the truth-prediction discrepancy map. The negative extreme value (-1) indicates regions where the model predicts tumor when the tumor is absent in ground truth (false positive, or over-prediction). The positive extreme value ($+1$) indicates regions where the model does not predict tumor when the tumor is present in the ground truth (false negative, or under-prediction).

6.4.0.2 Interpretation

To investigate use-case scenarios on how to interpret and analyse the uncertainty maps shown in Figure 6.6, preliminary qualitative data is collected from an expert radiological researcher. Note that this interpretation is not performed by board-certified radiologists, so it only serves as a guiding example.

Expert interpretation indicated that the pBNN is fairly confident in distinguishing the tumor core from the background, but is highly uncertain at the boundary regions of the tumor, which is consistent with manual segmentation [187]. For example, even when a single radiologist performs

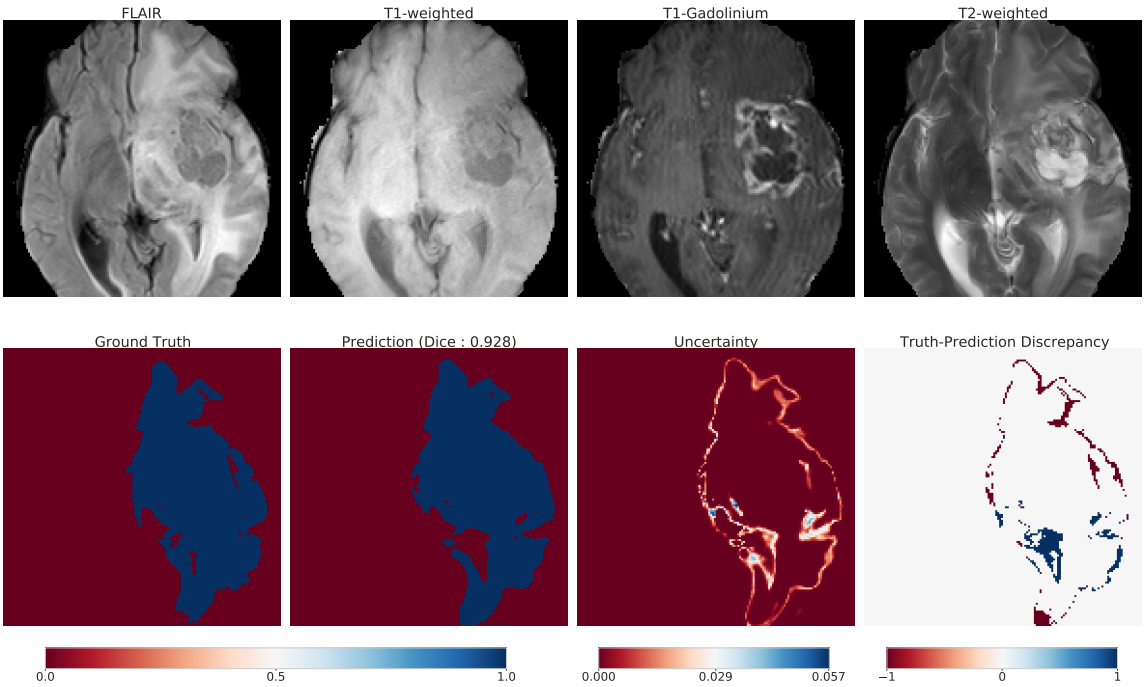


Figure 6.6: Uncertainty map accompanying a prediction made by the pBNN.

multiple segmentations on the same image, the most differences (intra-rater variability) are often present in the boundary region [188].

In this particular example, the necrosis, enhancing, and non-enhancing parts of the tumor are surrounded by edema. Since the model predicts the whole tumor and does not perform subclass recognition, it predicts the edematous region of the tumor. However, since edema has a distinct contrast in FLAIR, there is a possibility that this FLAIR modality is contributing largely to the observed uncertainty. This is in line with segmentation by radiologists, which also has the most uncertainty in FLAIR modality.

Further, comparing the uncertainty map with the truth-prediction discrepancy revealed that the pBNN is highly uncertain where it under-predicts and had low uncertainty where it over-predicts. A possible reason for this is in the context of the model's ability to correctly call out all of the tumor pixels in the ground truth. For over-prediction, the model covered the tumor region and went beyond it (which does not count as a "miss"), so the uncertainty is low. Whereas for under-prediction, the model failed to correctly call out tumor (which counts as a "miss"), thus the uncertainty is high. One such area of under-prediction is in the lateral ventricle of the right hemisphere. This region is darker than the rest of the tumor (in FLAIR modality), but is still a part of the tumor due to the presence of edema.

One possible explanation of this model behavior is that the model has learned to distinguish tumors based on pixel intensity and contrast. As the algorithm "sees" the lateral ventricle of

the right hemisphere to be significantly darker than the rest of the tumor, it thinks this is not a tumor. While this seems like a reasonable mistake for the model to make, a radiologist would have no problem marking this as a tumor as they have a firm understanding of anatomy. Example regions of over-prediction are near the middle frontal gyrus and near the necrotic region in the right hemisphere. This region is craggy, and it seems like the model smoothed over these intricacies to get a "good enough" prediction.

6.5 Discussion

The results of the empirical validation in Section 6.3 illustrate the effectiveness of the pBNN: the pBNN where Bayesian inference is performed on the SA-selected layer achieves the lowest normalized KL divergence—that is, compared to if a different layer were chosen, it provides the best capturing of the full BNN uncertainty. The three-step pBNN methodology in Section 5.1 is set up with implementation considerations in mind. Step 1 allows the leveraging of pre-existing trained DNNs, but also describes the procedure if a new DNN needs to be trained from scratch. Step 2 assesses and identifies the portion of DNN for Bayesian inference on a layer-by-layer basis (i.e. keeping "layer" as the unit), since accessing and replacing an entire layer (e.g., from a Dense layer to DenseFlipout layer) is very convenient in programming infrastructure such as TensorFlow Probability. Such modifications do not need the user to program new layers, or to Bayesianize portions of a layer. The gradient-based sensitivity measure adopted is also inexpensive to compute and can be readily extracted from a deterministic DNN. Step 3 takes advantage of existing implementations of Bayesian inference algorithms, such as the Flipout VI that can be specified and used together with non-Bayesian layers.

This method is particularly valuable for high-dimensional problems (e.g., DNNs with millions of parameters), where performing a full Bayesian inference would be extremely expensive. This work enables principled approximate Bayesian inference to be scaled up to millions of dimensions. While other related *partial* Bayesian strategies compare test error performance [181], this method explicitly computes closeness to the fully Bayesian model, with the aim of faithfully capturing the true uncertainty that is justified based on the quality and quantity of training data available. The overarching goal is provide a transparent quantification of uncertainty in an AI expert system, and not to only optimize for predictive accuracy.

Section 6.4 illustrates how this computational framework could be used to visually represent predictive uncertainty. While uncertainty communication is not the primary goal of this work, these contributions can help advance an understanding of AI limitations and uncertainty in a clinical healthcare context. A preliminary analysis on these maps demonstrates how a radiologist might interpret these images. The maps allow one to correlate the model's mistakes and how

confidently the model fails. With clinical insights and interpretation, one can also explore the model's behavior on specific use cases. For example, model predictions on out-of-distribution data can be potentially flagged based on uncertainty [189] to caution the radiologist about model's reliability in those regions. This would be especially useful for healthcare contexts where models are highly domain-dependent and localized [190]. Apart from potential to enable flagging of unreliable cases, this Bayesian framework can allow for construction of an expert-informed pBNN. It permits a flexible choice for prior and likelihood, potentially allowing practitioners to inject their specialized knowledge. For more information on this process, one should refer to the branch of Bayesian statistics called expert knowledge elicitation [191].

While this work produced promising results, limitations still exist. While the analysis of this approach included brute-force computation of the full BNN as an uncertainty reference point, this would not be done in practice. Assessing the quality of posterior approximation is a challenge in general for all BNNs, where inexpensive and effective diagnostics are needed to monitor the algorithm progress especially for large models. Analyzing ELBO curves is one such approach, but one cannot know when the ELBO is stuck in a local minimum. Furthermore, the magnitude of ELBO value does not directly indicate how far the variational approximation is to the true posterior, since ELBO differs from the KL divergence by an unknown constant (i.e. one should not expect ELBO does not converge towards zero with more epochs, as one would with mean-squared-error). This work also does not explore performing Bayesian inference on multiple layers, portions of layers, or individual weights, which may offer further improvements. Such expansions would require the development of efficient strategies to optimize SI across various combinations of multiple layers, and additional benchmarking and validation to assess the tradeoffs between their computational cost and ability to approximate the full BNN. Despite these limitations, this work demonstrates the effectiveness of SA-based pBNN to reduce computational cost and memory footprints of Bayesian UQ of large, million-dimensional DNNs.

6.6 Conclusion and Future Work

The work presented in this chapter proposes a novel approach to compute the Bayesian uncertainty of million-dimensional DNNs for medical image segmentation. Starting from a deterministic DNN, gradient-based sensitivity analysis is used to identify a layer to perform Bayesian inference, thereby creating a partially Bayesian neural network (pBNN) that is computationally much less expensive to construct than a full BNN. This chapter demonstrated the pBNN method on state-of-the-art 7.8-million parameter U-Net for brain tumor segmentation. This analysis indicated that the pBNN based on SA-selected layer provided the best approximation to the uncertainty from a full BNN, compared to other layer choices.

This methodology enables model developers and practitioners to compute the Bayesian uncertainty for large deep learning models. In a life-critical domain such as healthcare, deep learning models can have far-reaching impact, but also can make mistakes leading to disastrous consequences. Communicating uncertainty in model predictions help encourage clinicians to engage with the AI-based DSS with a healthy dose of skepticism and failure-centric mindset. Additionally, UQ mitigate "super-human" perceptions of AI that can lead to unjustified over-reliance. Thus, quantifying and communicating uncertainty would allow for safer and responsible deployments of AI-based CDSS in clinical workflows.

Future work will focus on conducting a large scale evaluation of the impact of uncertainty communication to radiologists for the task of tumor segmentation. The work presented in this chapter serves as a preliminary interpretation of these maps and demonstrate the possibility of such a study. Other future work will seek to understand if these maps can be used as a tool for model explanation, and whether these maps can be useful alongside conventional metrics. It can also be a useful tool in reducing alarm and click fatigue, as the model can only predict when absolutely certain, and refrain from predicting otherwise.

CHAPTER 7

Bayesian Convolutional Neural Networks with Intuitive Priors for Precision Health Balance Assessment

This chapter implements a Bayesian convolutional neural networks for the purpose of automatically assessing physical therapy balance tasks based on kinematic data collected from a single inertial measurement unit (IMU). A novel intuitive prior selection approach, as developed in Chapter 5, is also implemented in this chapter. Uncertainty is found to have a high correlation to error, which has implications for future data collection and also balance progression algorithms. he work presented in this chapter was the result of close collaboration with Jamie Ferris, Jenna Wiens, Kathleen Sienko, and Xun Huan and a manuscript is in the process of being prepared for submission in the *Journal of Machine Learning for Modeling and Computation*.

7.1 Introduction

Losses of balance and resultant falls are a chronic concern for large fraction of the general population, but particularly for older adults and people with vestibular (inner ear) disorders [49, 50]. Such falls can induce anxiety, fear of injury, and even loss of physical mobility. Balance training via physical therapy can be an effective treatment for poor balance [192, 193]. Thus far however, the most effective balance training programs require in-person supervision from a physical therapist (PT)[51, 52]. Remote supervision from a PT is an option, though it is typically less effective than in-person treatment; a major benefit to in-person supervision is real-time feedback, and delayed feedback may hinder rehabilitation [53, 194, 54]. While patients may be asked to self-assess their balance training performance, these self-assessments may be inaccurate relative to their PT's assessment and are particularly prone to overconfidence [195, 196]. This discrepancy can also lead to hindered rehabilitation or even injury. Furthermore, effective treatment and progression plans are dependent on the skill and knowledge of one's particular PT. Different PTs might have dramatically

different assessments of a patient's balance and might therefor have divergent treatment plans with different health and mobility outcomes.

The delayed feedback of remote supervision motivates a need for accurate, real-time balance exercise assessment. Recent progress has been made on this front by utilizing deep learning (DL) tools to map motion data collected from patients via inertial measurement units (IMU) to corresponding balance assessments made by PTs [55, 56]. Such DL tools can provide accurate and near-instantaneous automatic balance assessments, which can hasten in-home balance recovery. Furthermore, such tools can leverage the assessments of multiple PTs, potentially mitigating the biases of individual PTs.

Similar to [55] and [56], I explore using data from a single IMU on the lower back to capture body sway data to automatically assess balance. By leveraging the unprocessed IMU data, deep learning methods can automatically learn useful features for the task at hand, thus circumventing the need for labor-intensive or iterative feature engineering.

However, despite the incredible promise of DL, AI and ML systems have notable shortcomings, especially in the domain of uncertainty quantification. Traditional ML models often provide point estimates, lacking the ability to convey the degree of uncertainty associated with their predictions. Classical approaches such as bootstrapping data and aggregating ensemble models are more typically used for improving model robustness and getting confidence intervals on model performance metrics across whole datasets, rather than conveying uncertainty on individual predictions [197]. This limitation can be problematic in critical applications, as it makes it challenging to understand the model's confidence and reliability, hindering trust and transparency.

To address the issue of uncertainty quantification and model transparency, Bayesian neural networks (BNNs) have emerged as a promising solution. BNNs extend traditional neural networks by treating their weights as random variables and placing probability distributions over them.

Via Bayes' theorem, a BNN can carefully weigh the quality and quantity of data and update its understanding, or *posterior* distribution, on thousands or even millions of neural network parameters and hyperparameters. This approach allows BNNs to provide not just point predictions but also a measure of uncertainty in their predictions, which is invaluable for health and safety domains where data is limited and / or noisy and the consequences of incorrect decisions can be severe. A major challenge though in implementing BNNs is in characterizing their extremely high dimensional posteriors distributions.

The gold standard of numerical Bayesian inference is the Markov chain Monte Carlo (MCMC) class of sampling algorithms [97]. Such algorithms are well studied and often have assymptotic convergence guarantees for almost any posterior distribution (barring limitation from numerical precision). However, such algorithms become intractable in the extreme dimensionality cases of BNNs. Although nearly all Bayesian inference algorithms suffer to some degree from the curse of

dimensionality, most MCMC algorithms fail to converge in any reasonable amount of time when dimensionality exceeds thousands of dimensions, let alone tens of millions of parameters in state of the art predictive models.

Variational inference is a class of algorithms that optimizes the parameters of parameterized families of distributions to determine the instance of that family, that is the variational posterior, that best approximates the true posterior [35]. The Kullback-Leibler (KL) divergence is generally the metric that is selected to measure the discrepancy that is minimized between the true posterior PDF and the variational posterior PDF. The KL divergence has roots in information theory and is also very computationally tractable compared to other divergence metrics. Asymptotic convergence is unfortunately lost with variational inference computational tractability is a major benefit. Works such as [136] adopt mean-field assumptions in the variation inference formulation (MFVI), which uses parameterized families of independent Gaussian distributions to describe each parameter and enable faster convergence speed, at the cost of lower fidelity to the true posterior distribution.

More recently, the Stein Variational Gradient Descent algorithm [198] makes use of particles, rather than parameterized distribution families, to approximate the true distribution without the need for explicitly prescribed variational families. This work takes advantage of recent intersections between active subspace projections and particle-based Stein variational inference algorithms to conduct Bayesian inference on the parameters of the predictive models, thus yielding BNNs with both high scalability and high fidelity to the true Bayesian posterior distribution.

The second challenge to successfully implementing BNNs is in selecting Bayesian priors for the model weights. Selecting appropriate prior distributions is a non-trivial task, as it can significantly impact the model's performance and the quality of uncertainty estimates.

In this work, I introduce a novel approach for intuitive prior selection of BNN weights and demonstrate the implementation of SVGD and pSVGD algorithms for scalable inference of BNN weights posteriors on the real-world application of assessing balance in older adults. The study protocol was reviewed and approved under the oversight of the institutional review board IRBMED with protocol number HUM00086479. Informed consent was received from all individuals.

Using machine learning in physical therapy, like many other fields of medicine, requires great care to ensure that the model can be trusted and that risk is mitigated to any potential patient. Within the risk analysis community, risk is often understood as potential consequence weighed by potential incidence [199]. The consequences of poorly assessed and prescribed exercises can be quite severe. As such, mitigating risk in an ML expert system requires quantifying the uncertainty of its predictions.

ID	Age	Gender	Vestibular Diagnosis?
1	73	Female	Yes
2	73	Male	No
3	73	Female	Yes
4	69	Male	No
5	66	Male	No
6	23	Male	Yes
7	66	Female	No
8	76	Male	Yes
9	73	Female	No
10	65	Male	No

Table 7.1: Demographic information for the ten participants. All individuals recruited experienced balance concerns.

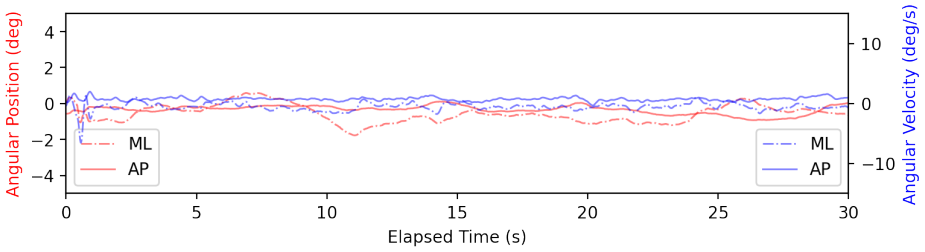
7.2 Methodology

7.2.1 Data

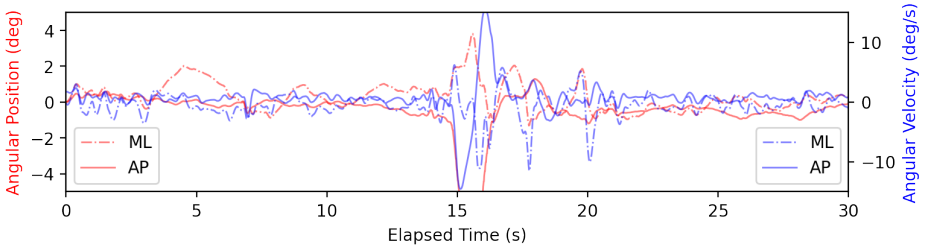
Data was collected from ten participants with balance concerns. Each participant performed 15 exercises, three repetitions each; exercises were randomly selected from 54 unique exercises. Each repetition had a duration of 30 seconds, though kinematic data was captured for several seconds before and after this period and was cropped in preprocessing. The 54 unique exercises were fully described by different combinations of 4 parameters: eye status, head position, stance, and surface type. These combinations were then one-hot encoded into an 11 dimensional vector for each exercise.

Each repetition was video recorded from both the front and back perspectives. The side-by-side videos were then reviewed by 2-5 PTs with specialization in treating patients with balance disorders. PTs assessed each repetition on a scale of 1-5, with a 1 representing excellent, independent performance with no sway and a 5 representing substantial sway, where a participant was unable to maintain position, even with assistance. Along with the PT assessments, self-assessments were collected from the participants themselves, based on a similar scale, though the language describing the scale differed slightly from the language given to the PTs. Both self ratings and PT ratings were stored as 1 dimensional vectors. Self ratings were used as a feature while PT ratings were the label being mapped to.

During exercises, kinematic data was captured for each patient via a single inertial measurement unit (IMU), worn on the lower back. The IMU measured body sway relative to gravity in the pitch (anterior-posterior) and roll (mediolateral) directions, sampled at 100 Hz. The kinematic data (angular position and angular velocity in both the pitch and roll directions), captured at 100 Hz



(a)



(b)

Figure 7.1: Top: a static standing exercise rated between 1 and 2 by multiple PTs. Bottom: the same static standing exercise performed by another subject, rated between 4 and 5 by multiple PTs.

for 30 seconds was thus encoded as a 3001x4 matrix for each repetition of an exercise. Figure 7.1 shows a visualization of this kinematic data and a distinct difference can be seen between the signal of an exercise rated 1-2 and an exercise rated 4-5.

Throughout the duration of each repetition, any loss of balance that drew safety concerns was recorded as a 'step-out'. Step-out were further encoded as one of three subcategories: 1) no assistance required, 2) little assistance required, 3) substantial assistance required. If a step-out occured, participants were encouraged to finish out a repetition up to the 30 second mark. The stepout information was one-hot encoded a binary vector with two elements.

The final set of input features were the kinematic data $X_{\text{kinematic}} \in \mathbb{R}^{3001\times4}$, the step-out data $X_{\text{step-out}} \in \{0,1\}^2$, exercise encoding $X_{\text{exercise}} \in \{0,1\}^{11}$ and the self-rating data $X_{\text{self-rating}} \in [1,5]$. For simplicity, the set of all features will be denoted X for the remainder of the paper. The PT rating label is denoted as $y \in [1,5]$.

The study protocol was reviewed and approved under the oversight of the institutional review board IRBMED with protocol number HUM00086479. Informed consent was received from all individuals.

7.2.2 Neural Networks

A neural network (NN) is a function that maps input x to output $\hat{y}$, that is–

$$\hat{y} = f(x) \tag{7.1}$$

where the hat in $\hat{y}$ denotes the NN prediction. The architecture of a given NN involves passing and processing information through successive layers of the network. For example, a densely connected feed-forward NN is organized into layers of different nodes, where each node performs calculations on output values from the previous layer before passing the result onto the next layer. Mathematically, the computation for node j in layer ℓ is

$$a_{\ell,j} = \sigma_\ell \left(\sum_i W_{\ell,ij} a_{\ell-1,i} + b_{\ell,j} \right) \tag{7.2}$$

where $a_{\ell,j}$ is the node output, σ_ℓ is an activation function (e.g., rectified linear units (ReLU), sigmoid function, hyperbolic tangent (tanh), etc.), $a_{\ell-1,i}$ is the ith nodal input (which is the output from the ith node in the previous layer), $W_{\ell,ij}$ is the weight placed on the ith nodal input, and $b_{\ell,j}$ is a biasing term. Definitions vary as to what qualifies a neural network to be "deep", though generally a DNN consists of two or more hidden layers (i.e., layers that are not the input or output layer).

Convolutional neural networks (CNNs) build upon this concept of dense neural networks, though rather than using matrix multiplication, cross-correlation[1] is used to produce outputs from input vectors and weight matrices. Mathematically, the computation for node j in layer ℓ is

$$a_{\ell,j} = \sigma_\ell \left(\sum_i W_{\ell,ij} * a_{\ell-1,i} + b_{\ell,j} \right), \tag{7.3}$$

where $*$ is the cross-correlation operation. CNNs excel at processing spatially or temporally correlated feature spaces, such as those found in time-series data, images, and video frames. 1D (for time-series) and 2D (for flat images) CNNs are most common, though higher order CNNs are possible.

[1]technically, the operation being performed in convolutional neural networks is cross-correlation, which is very similar though not identical to convolution. The term cross-correlation will be used in this paper to describe the operations performed within convolutional neural networks.

7.2.3 Bayesian Neural Networks

BNNs treat the parameters of a NN, ω, as random variables with associated probability density functions (PDFs) representing the uncertainty or belief state on ω. When training data become available, these PDFs are updated via Bayes' theorem:

$$p(\omega|x_T, y_T) = \frac{p(\omega)p(y_T|x_T, \omega)}{p(y_T|x_T)}, \tag{7.4}$$

where $p(\omega)$ is the prior PDF on the weight parameters and we also assumed $p(w|x_T) = p(w)$ (i.e., the prior uncertainty should not change from knowing only the input values of the training data), $p(y_T|x_T, \omega)$ is the likelihood function, $p(w|x_T, y_T)$ is the posterior PDF, and $p(y_T|x_T)$ is the Bayesian evidence. Solving the Bayesian inference problem then entails computing the posterior $p(w|x_T, y_T)$—that is, our updated uncertainty on w given the training dataset (x_T, y_T).

7.2.4 Likelihood

The purpose of the Bayesian likelihood function is to assess the probability of observed data as a function of the parameters of a given model. The likelihood modifies the prior distribution in accordance with Bayes' theorem in order to arrive at the data-informed posterior distribution.

The exact likelihood distribution used to infer parameter distributions for a given statistical model is often a matter of user selection. Typically for regression-type problems, a normal distribution is selected to model the discrepancy between the model predictions and observed data. This is analogous to mean squared error (MSE) in a frequentist setting.

In our model, we assume additive noise in the form:

$$y_T = f(\omega, X_T) + \sigma_\epsilon$$

Thus, we assume a likelihood function based on the normal distribution and the error between the model and the observed data:

$$\mathcal{N}(0, \sigma_\epsilon^2; y_T - f(\omega, X_T))$$

Selection of the noise parameter, σ_ϵ, should be based on observed noise in the data. Because each exercise repetition is assessed by more than one PT, we can calculate a normal distribution that approximates the observed discrepancy between different PT ratings made on identical video footage of a given exercise.

7.2.5 Prior Selection

One challenge of Bayesian inference on the highly abstract parameter space of a deep neural network is that there is very little intuition as to what a prior distribution should look like. However, from a model regularization and performance perspective, certain priors can still be useful even if the distributions are not physically intuitive. Zero-mean Laplace priors, analogous to L_1 regularization in frequentist settings, is useful encouraging model parsimony. Zero-mean Gaussian priors, analogous to L_2 regularization in frequentist settings, is useful for avoiding model overfitting. When used together as a joint Laplace-Gaussian distribution, they induce an elastic net regularization.

We propose prior selection via a validation or cross validation scheme, analogous to frequentist regularization parameter tuning. Deterministic models are affordably trained via maximum *aposteriori* estimates (MAP) based on a selection of different priors along with the likelihood function above. The MAP estimates on model parameters are then used for producing model predictions on a held-out validation set. Final priors are selected based on relevant model performance metrics, such as mean squared error, on the validation set. This is different from the so-called "inverse crime" of choosing a prior based on knowledge of a posterior distribution.

7.3 Results

pBNN models were trained using the Nadam [138] optimizer for 400 epochs with a learning rate of 10^{-3} and batch size of 128. Computations were done using 8 GeForce RTX 2080 Ti GPUs in parallel from the University of Michigan Precision Health cluster. This was supported in part through computational resources and services provided by Advanced Research Computing (ARC), a division of Information and Technology Services at the University of Michigan.

Figure 7.3 shows posterior predictions (median predictions and along with confidence intervals) based on SVGD and pSVGD inference algorithms. In this application, we found pSVGD to produce more accurate median predictions while also producing predictions with greater uncertainty (high variance). We expect that the improved accuracy is due to the regularizing effect of projecting onto the likelihood informed subspace, truncating off components that are minimally influenced by the data. We expect that the greater variance on the predictions is due to the dimensionality reduction process, which in turn allowed for a higher ratio of number of particles to dimensions of inferred parameters, as compared to SVGD particles with the same computational budget. Because SVGD particles are known to underestimate posterior variance and collapse to the posterior mode, we believe that the higher variance of the pSVGD posterior distributions is a better approximation to the full Bayesian posterior, as the pSVGD posterior is less collapsed. As such, the greater variance of the pSVGD particles is indicative of higher fidelity Bayesian inference rather than poor

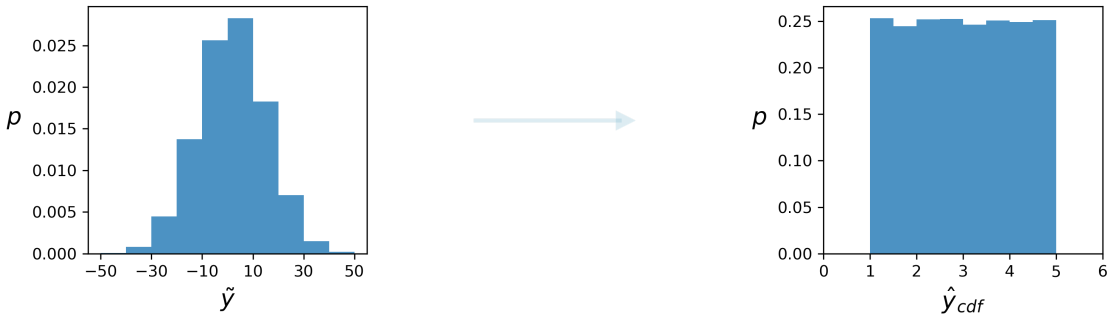


Figure 7.2: Illustration of example untransformed and transformed prior predictive distribution. On the left is the approximately Gaussian prior predictive distribution, created by sampling from the weight prior distributions and propagating the corresponding predictions through the NN with a linear output activation. On the right is predictions made from the same prior weight sample, but with a modified Gaussian CDF as the output activation function. The transformed prior predictive is now an approximately uniform distribution over the domain of the possible PT ratings. A uniform prior predictive is an intuitive distribution for this application, though transforming from the uniform distribution to any other PDF of our choosing would be trivial.

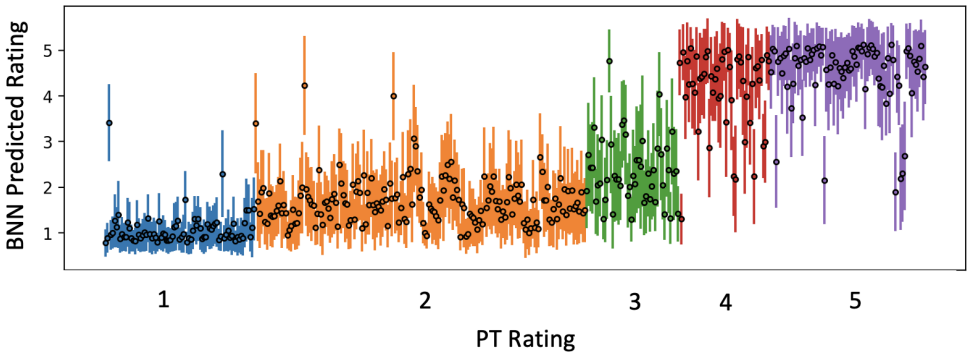
uncertainty quantification. This is supported by the trends shown in Figure 7.4, which shows a positive correlation between error and predictive uncertainty. Furthermore, the pSVGD approach shows a lower expected error than the vanilla SVGD approach, indicating pSVGD algorithm is better calibrated in this exemplar.

7.4 Conclusion and Future Work

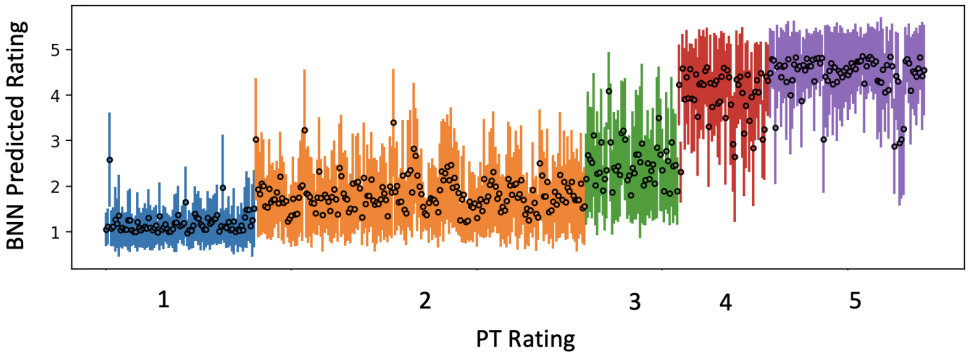
Automatic balance assessment tools have the potential to greatly improve the safety and effectiveness of home balance training programs. Deep neural networks are a promising tool for developing

Model	Log Likelihood	% ACC within 0.5	% ACC within 1.0
MEAN	-4.17 [-4.34, -3.98]	13 [11, 18]	46 [42, 50]
CNN	-0.90 [-0.97, -0.84]	63 [60, 65]	93 [91, 94]
BCNN	-0.76 [-0.79, -0.73]	68 [66, 70]	96 [94, 96]
pBCNN	**-0.73 [-0.75, -0.71]**	**71 [69, 73]**	**96 [95, 96]**
PT	-0.71 [-0.73, -0.69]	76 [74, 79]	97 [96, 98]

Table 7.2: Baseline PT performance (PT) is compared against four other data-driven models: Mean Label Predictor (MEAN); Cross-validated Deterministic CNN (CNN), Bayesian CNN (BCNN), Projected Bayesian CNN (pBCNN). 95% confidence intervals are calculated from bootstrap re-sampling the dataset 100 times. Metrics shown for the Bayesian methods are averaged across all SVGD / pSVGD particles.



(a) SVGD



(b) pSVGD

Figure 7.3: SVGD and pSVGD posterior predictive distributions for each exercise across all subjects. Predictions are color coded based on their underlying PT rating. Dots indicate the median predictions while error bars indicate the 95% credible interval based on the posterior predictive distributions propagated via SVGD and pSVGD respectively.

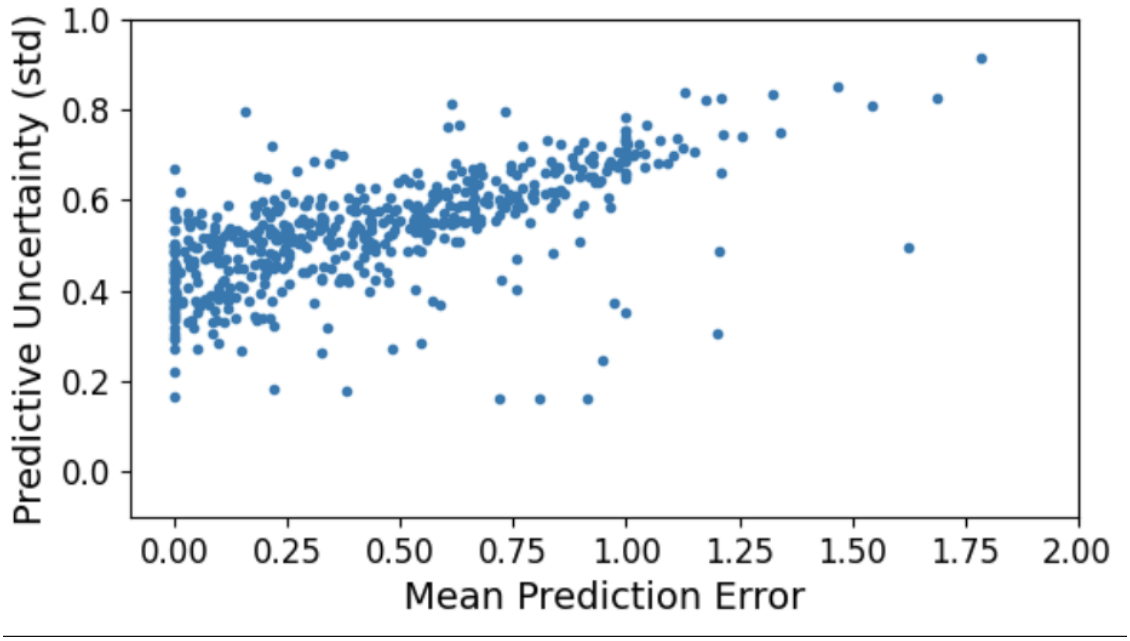


Figure 7.4: Predictive uncertainty, expressed as the standard deviation of the pSVGD posterior predictive distribution as a function of the error between the posterior predictive mean and the true label. Spearman $C = 0.68$.

accurate expert systems for assessing balance. However, high quality data for training these models is limited and expensive to acquire. As such, it is imperative that uncertainty be quantified in a high fidelity manner in these models, to ensure that models are not dangerously misused.

In this work, I investigated two state of the art algorithms, SVGD and pSVGD, for quantifying Bayesian uncertainty in DNN expert systems. These DNN architectures were able to map kinematic information from a single IMU along with an exercise encoding and step-out information to exercise assessment labels provided by professional PTs with accuracy the exceeded individuals own self assessments.

The models used in this work were convolutional neural network models featuring 1D time-series representations and 2D stabiligram representations. Both 1D time-series and 2D stabiligram representations yield convolutional models whose sizes are tied to the fineness of their discretization. The intuition is that both time-series and stabiligram representations have some innate, low dimensional structure that is independent of the discretization fineness and which can be exploited by active subspace techniques. I further found that although pSVGD yielded higher uncertainty bounds, the expected error of the posterior predictive distribution via pSVGD was comparable or lower than via SVGD. This suggests that in our application, pSVGD was able to better allocate probability density such that lower uncertainty was more often given to accurate model predictions while higher uncertainty was more often given to less accurate model predictions.

This work also made use of a novel predictive informed prior selection methodology. This prior selection approach leverages a hyperparameter search technique that is common in deep neural network literature in order too determine Bayesian prior scale parameters on the neural weights that induce the best predictive performance. However, this weight space is very abstract and encoding prior beliefs in these weights is a non-trivial task. A final transform is applied to the output layer in order to maintain prior predictive distributions that still capture physical intuition and expert opinion, while maintaining the desirable predictive benefits of the priors selected via hyperparameter search.

There are several limitations to this work. Firstly, the data was collected in a lab environment. This allowed for physical therapist intervention and correction of any IMU misalignment. These conditions are not representative of an at-home environment, where a physical therapy patient would have to manage their own stepouts or even losses of balance. This could impact confidence or lack thereof in a real world at-home setting.

Secondly, this dataset was relatively small, and thus more complex state-of-the- art deep learning architectures were not explored. The labels were also provided by a small number of physical therapists, who in turn showed a non-negligible amount of disagreement on individual exercise ratings. This would indicate noisy labels, and future work would do well to investigate this aleatoric noise. Hierarchical Bayesian formulations would be a promising route for automatically learning this noise as well as general model misspecification. The PT disagreement on individual exercises also calls into question whether a scalar rating system (1-5) is adequate for assessing a person's performance, as opposed to a ratings scale that captures balance across multiple dimensions of ability.

Lastly, the predictive informed prior selection methodology is currently limited by the ability to leverage a CDF transform to map the initial prior predictive to the informed prior predictive. Future work will focus on enabling high dimensional output spaces that may be difficult to accomplish with multidimensional CDFs. Future work will also focus on further developing conditional prior predictive distributions, in which the prior predictive can be customized for individual data points, based on prior belief.

CHAPTER 8

Conclusion

8.1 Summary

In recent years, development and implementation of deep learning (DL) algorithms have become ubiquitous in many fields of science and engineering. DL offers a promising approach for detecting complex trends in increasingly large databases of sensor measurements, physics-based computer simulations, and survey data, among many others. However, relatively few attempts exist to rigorously and systematically quantify the uncertainty in DL models, especially those related to the lack of understanding towards the model stemming from the quality and quantity of the available training dataset. In some domains, data might be rich enough, and the risks associated with misclassifications low enough, to not warrant the extra effort to quantify model uncertainty. However, in many scientific domains, such as those dealing with medical data, healthcare, safety and early warning devices, and the design of high-cost physical or computational experiments, the cost of poor machine learning (ML) predictions and false confidence in their appearance of 'super-human abilities' can lead to expensive or even dangerous mistakes.

This dissertation advances the state-of-the-art in large-scale Bayesian uncertainty quantification (UQ) in deep neural networks (DNNs)—that is, to build Bayesian neural networks (BNNs)—and draws new insights on model uncertainty in several unique science and engineering applications. Such insights on model uncertainty could help avoid some of the consequences of model overconfidence and by providing important metrics for decision making and new data collection.

Chapter 2 of this dissertation describes the current state-of-the-art of scalable Bayesian uncertainty quantification algorithms, including variational inference, Stein variational gradient descent (SVGD) and the projected SVGD (pSVGD). Chapter 3 and Chapter 4 demonstrate the implementation of these state-of-the-art algorithms to conduct Bayesian inference on various architectures of DNNs built from high-fidelity physics simulations. Chapter 5 presents a novel heuristic for reducing the Bayesian parameter space to create a 'partially Bayesian' model with minimal compromise on the fidelity of quantified uncertainty; as well as a novel approach for selecting intuitive, domain-

informed priors in the high dimensional model parameter space of DNNs.Chapter 6 demonstrates the implementation of this 'partially Bayesian neural network' (pBNN) in an automatic brain tumor segmentation application and draws insights on this uncertainty that may be of interest to radiologists, surgeons, and other clinicians. Chapter 7 combines the novel prior selection process with SVGD and pSVGD in the application of a precision health automatic balance assessment tool.

8.2 Contributions

8.2.1 Methodology

The first major contribution of this dissertation is to help facilitate widespread adoption of rigorous model UQ in DL models used for high-stakes science and engineering applications. This can and should be done in addition to other standard practices such as model selection, model validation, and model calibration that help ensure trustable and generalizable models. I achieved this through two methodological contributions, both described fully in Chapter 5.

The first methodological contribution is a heuristic for subselecting layers of weights in a DNN targeted for Bayesian inference, in a process called "partially Bayesian neural networks". Rather than performing the expensive task of conducting Bayesian inference on the entire parameter space of a DNN, one need only select a subset of high impact layers upon which to perform Bayesian inference that is most "uncertainty perserving", while using less costly approaches such as maximum likelihood estimation or maximum *a priori* estimation for the remaining layers. This work identified a strong inverse correlation between the sensitivity of a layer, measured through an index based on the locals gradients of layer parameters, and the divergence between the pBNN associated with that layer and the much more expensive fully Bayesian DNN.

The general approach of dimensionality reduction is not necessarily unique. One of the first Bayesian neural network techniques only performed inference on the final layer[118] in order to remain computationally feasible. Techniques such as [200] and [82] take a dimensionality reduction approach to Bayesian inference (with MCMC and SVGD respectively) via projections. My approach more heuristic-driven and less invasive, but is similar in that the dimensionality of the problem is greatly reduced compared to a hypothetical full BNN problem.

The second methodological contribution entailed developing a process for selecting Bayesian priors on abstract, high-dimensional DNN weights while preserving model predictive probability density functions that encode tangible, domain knowledge. This process also considers that Bayesian priors serve to not only encode domain knowledge but also serve important functions in model regularization. This process utilizes standard validation and hyperparameter selection procedures to test different weight priors and then identifies a final transform layer that ensures

that these weight priors correspond to a desired predictive prior. This process helps bridge the gap between two major Bayesian interpretations of priors– one that sees priors as tools for improving model performance and one that sees priors as a way to encode knowledge or individual opinion of a system.

The search for better ways of selecting BNN priors is still very much an area of active research. More recent approaches include hyperpriors [201], wherein the authors use a hierarchical Bayesian prior formulation to learn prior variances from data. Another promising approach is to learn functional priors [202], where priors are tuned in order to encode Gaussian process functionals.

Importantly, both the pBNN and the intuitive prior selection approach outlined in Chapter 5 are general purpose and model agnostic, and can be applied to a wide range of neural network architecture.

8.2.2 BNN for Complex Physics-Based Simulations

The next major contribution of this dissertation is in implementing BNNs via SVGD and pSVGD in two applications to efficiently map between quantities of interest identified in high-fidelity physics simulations. The first, described in Chapter 3, develops a novel process for mapping rotorcraft acoustic signatures and quantities of interest relating to the flight performance of the rotorcraft. Importantly, this flight performance can deteriorate substantially in the presence of ice accretion. However, as ice builds up on the leading edge of a rotor, the rotor acoustic signature is also modified substantially. Advances in high fidelity computational aeroacoustics and computational fluid dyanamics (CFD) allow accurate simulation of these icing scenarios. BNNs are then implemented to then map between observed acoustic signals, which can be observed directly with microphone-type sensors, and flight performance quantities such as coefficients of drag, lift, and moment, and are not directly observable. Furthermore, BNN can predict these mappings along with valuable model uncertainty information at a rate compatible with real-time monitoring. This process bypasses the expensive and intractable process of running CFD analysis iteratively on different icing geometries in hopes of identifying the geometry responsible for an observed acoustic signature. The second application, described in Chapter 4, develops a process for using BNN to map a polycrystaline material and a given force function to the material stress response. This process allows for near instantaneous predictions of how a complex polycrystaline material would react to a given force. This has implications for a range of science and engineering applications, where expensive physics simulations, primarily finite element analysis (FEA), are used to aid in design and validation. The BNN mapping is substantially faster than FEA and also allows for the consideration of minute changes in the material microstructure.

In many ways, Chapter 4 builds directly on the advancements made in Chapter 3. Both

projects utilized time-series data from high fidelity physics experiments and Bayesian inference was conducted using SVGD and pSVGD algorithms. However, data used in the icing CFD and CAA dataset were simplified and binned into discrete frequency bins. BNNs were then implemented exclusively via fully-connected, dense layers. While dense layers achieved good performance on this small dataset, a larger dataset would likely require a more sophisticated architecture. On the other hand, Chapter 4 investigated a much larger and complex dataset of poly-crystalline stress response. Rather than binning into discrete frequencies, the time-series data was processed directly using GRU layers. Automatic feature detection was further conducted using graph convolutional layers. These architectures are much more inline with the current state-of-the-art for analyzing complex time-series data. The size and complexity of this data presented unique challenges that were not encountered in the helicopter icing dataset, particularly with respect to identifying an appropriate active subspace for pSVGD.

8.2.3 BNN for Real-World Healthcare Data

Chapter 6 and Chapter 7 diverge from physics simulations and focus on data collected from the real-world healthcare domain with human expert labels. Data is limited and expensive to acquire, labels are noisy and conditional on the subjective labeling of different domain experts. Above all, UQ is an essential component of these expert systems due to the risk of injury or misdiagnosis. Each of these chapters demonstrated a different novel methodological contribution toward making BNNs more practical for use in large neural network models. Chapter 6 demonstrates the novel pBNN to sub-select parameters for Bayesian inference while minimizing divergence from a more expensive, fully Bayesian neural network. The pBNNs were used to gain useful uncertainty information on the state-of-the-art U-Net tumor segmentation model. Additional insights were gained on how physicians and radiologists might use this predictive uncertainty in a clinical setting. Chapter 7 demonstrated a novel approach for transforming expert-informed priors on a BNN output and encoding this information in the parameter space of a BNN. The BNN was then effectively used to map directly from kinematic data from a single IMU to ratings provided by professional PTs. Insight was given on how uncertainty could be used to guide future data collection.

Both applications will require additional research to make use of the predictive uncertainty in the larger healthcare decision processes, which is elaborated upon in Section 8.3.

8.3 Limitations and Future Work

8.3.1 Methological Limitations

While this dissertation significantly advances the state-of-the-art in Bayesian uncertainty quantification for deep neural networks (DNNs), it is essential to acknowledge certain limitations that warrant further consideration and investigation.

Firstly, the claim that Bayesian neural networks (BNNs) are now completely solved would be overly optimistic. Though the dissertation successfully compares small test problems to reference solutions, there remains a need for more in-depth discussions and diagnostics, especially in larger settings. Improved diagnostics are crucial for a better understanding of how accurately uncertainty is being calculated in comparison to the "true uncertainty". Identifying the thresholds for under/over predicting uncertainty is an essential avenue for future research, particularly in complex scenarios.

Moreover, the dissertation touches upon the issue of numerical stability for large systems. This was particularly evident in the context of active subspace eigenvectors in the GCNN case. This limitation emphasizes the need for further exploration and development of methodologies that ensure numerical stability in the face of larger and more intricate systems. [203] introduces the randomized numerical linear algebra class of algorithms that offer fast, stable, and scalable methods that would be useful for generating a high quality Hessian decomposition for the purpose to determining the active subspace. [204] also alludes to randomized methods as being promising approaches for the Hessian decomposition.

Another limitation pertains to the exclusive focus on uncertainty in the parameters of the DL model. The dissertation could benefit from extended discussions on model-form uncertainty, considering the expressiveness of the chosen architecture and potential implications. Additionally, examining scenarios where incorrect likelihood models are assumed could provide valuable insights into the broader landscape of uncertainty. [205] provides an excellent introduction to this field of Bayesian model comparison and appropriately weighing different likelihood models.

Computational resource management poses a practical challenge that demands attention. The trade-off between the number of particles used in SVGD and the number of training epochs requires careful consideration. Striking a balance to avoid allocating excessive resources to one aspect over the other, and developing mechanisms for automatic resource management, stands out as a pertinent limitation that invites further research.

In summary, the achievements of this dissertation, while notable, prompt the recognition of several limitations that highlight the complexity and nuances inherent in the field of Bayesian uncertainty quantification for DNNs. Addressing these limitations will not only refine the existing methodologies but also pave the way for more robust and applicable solutions in the dynamic landscape of SciML.

8.3.2 Active Learning for Model Improvement

In many scientific domains, there are opportunities to collect new data and to adapt data collection strategies in response to insights gains from previous data collection. However, in selecting new data to collect, not all data are equally useful. In the context of predictive modeling, not all training data has the same impact on improving model performance. Active learning is an area of research that develops strategies for iteratively selecting new data to collect or label, typically with the goal of maximally improving model performance for a given number of data samples. [206] gives an extensive survey of different types of active learning, as well as related topics such as semi-supervised learning and reinforcement learning. Invariably, these active learning meta-strategies must balance the opposing strategies of exploration and exploitation. That is to say, they must balance exploring data domains where data is sparse and performance is unknown, and exploiting data domains where model performance is known to be poor.

Future work to leverage model uncertainty as a metric in quantifying the exploration-exploitation trade-off is an interesting avenue, approaches such as expected improvement specifically encode uncertainty in weighing the best candidates for new data collection [207, 208]. For example, an active learning strategy might exploit selecting data points similar to existing training data that has high uncertainty An active learning strategy might also explore hypothetical feature sets and determine which correspond to the greatest uncertainty, and therefore which feature sets are most worthy of collecting actual data from. This is not a trivial task, as feature sets, particularly high dimensional image or time-series based features, might not be directly tuneable when selecting new data collection parameters. Instead, a user would need to select from these data collection parameters, likely low dimensional, and map to their most likely feature sets. This approach would likely be quite computationally expensive, but would be worthwhile in situations where new data is extremely costly to collect and/or opportunities to collect new data are very limited; any insight on how to collect the most useful data would be worthwhile. In this regard, there will likely be insights to be gained from the literature of optimal experimental design, a field which shares a similar goal of maximizing the impact or informativeness of a limited amount of new data.

8.3.3 Decision Boundaries with Uncertainty

In several of the applications that were investigated in this dissertation, DL predictions are just one part of a larger decision making process. For example, when assessing the flight performance coefficients of an iced helicopter rotor, there is an implied decision that must be made by the pilot as to whether the flight performance is too poor or the uncertainty too high to justify flying. Should the pilot seek an immediate emergency landing or just proceed with caution? At what threshold does ‘somewhat-uncertain’ become ‘too uncertain’? Similarly, when segmenting an MRI scan for

the presence of tumors, a natural decision follows of whether to perform an invasive surgery or not; if surgery, then where along the fuzzy gradient of 'definitely tumor' to 'possibly tumor' to 'definitely not tumor' would a surgeon want to make an incision. When using IMU data to assess a physical therapy patient's ability to perform on a given exercise, the natural question raised is whether they are doing the right exercise and which exercise should they do next? If a model is highly uncertain about the patient's current balance performance, how should the progression plan proceed?

In each of these applications, there is a weighing that needs to be made regarding how to incorporate model uncertainty into the corresponding decision. Most likely, the end user (whether pilot or brain surgeon or physical therapy patient) would not be able to make use of a raw probability density function, as informative as it is. Instead, this probability will likely need to be mapped to a set of tangible decision boundaries and discrete actions. [209] makes important contributions toward understanding neural network classification boundaries and investigates some Bayesian approaches for this problem. This process can draw heavily from the literature of risk analysis. Recently, [210] investigates approaches for bridging Bayesian probabilistic models with decision risk.

BIBLIOGRAPHY

[1] Nathan Baker, Frank Alexander, Timo Bremer, Aric Hagberg, Yannis Kevrekidis, Habib Najm, Manish Parashar, Abani Patra, James Sethian, Stefan Wild, and Karen Willcox. Workshop Report on Basic Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence. Technical report, U.S. Department of Energy Office of Science (SC), 2019.

[2] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

[3] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *Ieee Access*, 8:42200–42216, 2020.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[5] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[6] Erik Gawehn, Jan A. Hiss, and Gisbert Schneider. Deep learning in drug discovery. *Molecular Informatics*, 35(1):3–14, 2016.

[7] Matthew D. Schwartz. Modern Machine Learning and Particle Physics. *Harvard Data Science Review*, 3(2), may 13 2021. https://hdsr.mitpress.mit.edu/pub/xqle7lat.

[8] Ricardo Vinuesa and Steven L. Brunton. Enhancing computational fluid dynamics with machine learning. *Nature Computational Science*, 2(6):358–366, June 2022.

[9] Ehsan Haghighat, Maziar Raissi, Adrian Moure, Hector Gomez, and Ruben Juanes. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 379:113741, 2021.

[10] Kamal Choudhary, Brian DeCost, Chi Chen, Anubhav Jain, Francesca Tavazza, Ryan Cohn, Cheol Woo Park, Alok Choudhary, Ankit Agrawal, Simon J. L. Billinge, Elizabeth Holm,

Shyue Ping Ong, and Chris Wolverton. Recent advances and applications of deep learning methods in materials science. *npj Computational Materials*, 8(1):59, April 2022.

[11] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, apr 2016.

[12] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, and fnm Prabhat. Deep learning and process understanding for data-driven earth system science. *Nature*, 566(7743):195–204, 2019.

[13] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[14] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, February 2019.

[15] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[16] Ralph C Smith. *Uncertainty quantification: theory, implementation, and applications*, volume 12. Siam, 2013.

[17] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.

[18] Edmon Begoli, Tanmoy Bhattacharya, and Dimitri Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 2019.

[19] Richard T. Cox. Probability, frequency and reasonable expectation. 14(2):1–13, 1946.

[20] Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007):453–461, 1946.

[21] Edwin T Jaynes. Prior probabilities. *IEEE Transactions on systems science and cybernetics*, 4(3):227–241, 1968.

[22] Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

[23] Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*, 33:6514–6527, 2020.

[24] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087, 1953.

[25] WK Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[26] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer New York, New York, NY, 2004.

[27] Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An Introduction to MCMC for Machine Learning. *Machine Learning*, 50:5–43, 2003.

[28] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors. *Handbook of Markov Chain Monte Carlo*. Chapman and Hall, 2011.

[29] Gareth O Roberts and Richard L Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363, 1996.

[30] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[31] Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

[32] Michael Betancourt. A Conceptual Introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.

[33] Michael I. Jordan, Tommi S. Jaakkola, Lawrence K. Saul, and Florham Park. An Introduction to Variational Methods for Graphical Models An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37:183–233, 1999.

[34] Martin J. Wainwright and Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2007.

[35] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[36] Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 2378–2386, Barcelona, Spain, 2016.

[37] Jingwei Zhuo, Chang Liu, Jiaxin Shi, Jun Zhu, Ning Chen, and Bo Zhang. Message passing stein variational gradient descent. In *International Conference on Machine Learning*, pages 6018–6027. PMLR, 2018.

[38] Peng Chen and Omar Ghattas. Projected stein variational gradient descent. *Advances in Neural Information Processing Systems*, 33:1947–1958, 2020.

[39] Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366:415–447, 2018.

[40] Jing Chang Zhuge, Zhi Jing Yu, and Jian Shu Gao. Ice Detection Based on Near Infrared Image Analysis. In *Applied Mechanics and Materials*, volume 121, pages 3960–3964. Trans Tech Publ, 2012.

[41] Christopher E Bassey and Gregory R Simpson. Aircraft Ice Detection Using Time Domain Reflectometry With Coplanar Sensors. In *2007 IEEE Aerospace Conference*, pages 1–6. IEEE, 2007. DOI:10.1109/AERO.2007.352857.

[42] Derrick D Hongerholt, Gary Willms, and Joseph L Rose. Summary of Results From an Ultrasonic In-Flight Wing Ice Detection System. In *AIP Conference Proceedings*, volume 615, pages 1023–1028. American Institute of Physics, 2002.

[43] Fikret Caliskan and Chingiz Hajiyev. A review of in-flight detection and identification of aircraft icing and reconfigurable control. *Progress in Aerospace Sciences*, 60:12–34, 2013.

[44] Reese Jones, Cosmin Safta, and Ari Frankel. Deep learning and multi-level featurization of graph representations of microstructural data, 2022.

[45] Tom McLeod Logan, Terje Aven, Seth David Guikema, and Roger Flage. Risk science offers an integrated approach to resilience. *Nature Sustainability*, 5(9):741–748, September 2022.

[46] Frances Griffiths, Eileen Green, and Maria Tsouroufli. The nature of medical evidence and its inherent uncertainty for the clinical consultation: qualitative study. *BMJ*, 330(7490):511, January 2005.

[47] Alexander Campolo and Kate Crawford. Enchanted determinism: Power without responsibility in artificial intelligence. *Engaging Science, Technology, and Society*, 6:1–19, January 2020.

[48] Edmon Begoli, Tanmoy Bhattacharya, and Dimitri Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 2019.

[49] Susan J Herdman, Philip Blatt, Michael C Schubert, and Ronald J Tusa. Falls in patients with vestibular deficits. *Otology & Neurotology*, 21(6):847–851, 2000.

[50] Mary E Tinetti. Preventing falls in elderly persons. *New England journal of medicine*, 348(1):42–49, 2003.

[51] Victor Lun, Nancy Pullan, Nancy Labelle, Corey Adams, and Oksana Suchowersky. Comparison of the effects of a self-supervised home exercise program with a physiotherapist-supervised exercise program on the motor symptoms of parkinson's disease. *Movement disorders: official journal of the Movement Disorder Society*, 20(8):971–975, 2005.

[52] Chung-Lan Kao, Liang-Kung Chen, Chang-Ming Chern, Li-Chi Hsu, Chih-Chun Chen, and Shinn-Jang Hwang. Rehabilitation outcome in home-based versus supervised exercise programs for chronically dizzy patients. *Archives of gerontology and geriatrics*, 51(3):264–267, 2010.

[53] Kathleen H Sienko, M David Balkwill, LIE Oddsson, and Conrad Wall. Effects of multi-directional vibrotactile feedback on vestibular-deficient postural performance during continuous multi-directional support surface perturbations. *Journal of Vestibular Research*, 18(5-6):273–285, 2008.

[54] Kathleen H Sienko, Rachael D Seidler, Wendy J Carender, Adam D Goodworth, Susan L Whitney, and Robert J Peterka. Potential mechanisms of sensory augmentation systems on human balance control. *Frontiers in neurology*, 9:944, 2018.

[55] Tian Bao, Brooke N Klatt, Susan L Whitney, Kathleen H Sienko, and Jenna Wiens. Automatically evaluating balance: a machine learning approach. *IEEE transactions on neural systems and rehabilitation engineering*, 27(2):179–186, 2019.

[56] Fahad Kamran, Kathryn Harrold, Jonathan Zwier, Wendy Carender, Tian Bao, Kathleen H Sienko, and Jenna Wiens. Automatically evaluating balance using machine learning and data from a single inertial measurement unit. *Journal of NeuroEngineering and Rehabilitation*, 18(1):1–7, 2021.

[57] Michael Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.

[58] Vincent Fortuin. Priors in bayesian deep learning: A review. *International Statistical Review*, 90(3):563–591, 2022.

[59] Greg M Allenby and Peter E Rossi. Hierarchical bayes models. *The handbook of marketing research: Uses, misuses, and future advances*, pages 418–440, 2006.

[60] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987.

[61] Radford Neal. *MCMC Using Hamiltonian Dynamics*. CRC Press, May 2012.

[62] Radford M. Neal. MCMC Using Hamiltonian Dynamics. In *Handbook of Markov Chain Monte Carlo*, pages 113–162. 2011.

[63] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches, 2018.

[64] Ranganath Krishnan, Mahesh Subedar, and Omesh Tickoo. Specifying weight priors in bayesian deep neural networks with empirical bayes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4477–4484, Apr. 2020.

[65] Peng Chen, Keyi Wu, Joshua Chen, Thomas O'Leary-Roseberry, and Omar Ghattas. Projected stein variational newton: A fast and scalable bayesian inference method in high dimensions, 2019.

[66] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[67] Myles Morelli, Jeremiah Hauth, Alberto Guardone, Xun Huan, and Beckett Y. Zhou. A rotorcraft in-flight ice detection framework using computational aeroacoustics and bayesian neural networks. *Structural and Multidisciplinary Optimization*, 2023 [In review].

[68] Gareth D Padfield. *Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modelling*. Blackwell Publishing, Second Edition, 9600 Garsington Road, Oxford, OX4 2DQ, UK, 2008.

[69] MB Bragg, GM Gregorek, and JD Lee. Airfoil Aerodynamics in Icing Conditions. *Journal of Aircraft*, 23(1):76–81, 1986.

[70] RW Gent, NP Dart, and JT Cansdale. Aircraft Icing. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 358(1776):2873–2911, 2000. https://doi.org/10.1098/rsta.2000.0689.

[71] M Bragg, Tamer Basar, W Perkins, M Selig, P Voulgaris, J Melody, and N Sarter. Smart Icing Systems for Aircraft Icing Safety. In *40th AIAA Aerospace Sciences Meeting & Exhibit*, page 813, 2002.

[72] Vijay K Varadan, Vasundara V Varadan, and Xiao-Qi Bao. IDT, SAW, and MEMS Sensors for Measuring Deflection, Acceleration, and Ice Detection of Aircraft. In *Smart Structures and Materials 1997: Smart Electronics and MEMS*, volume 3046, pages 209–219. International Society for Optics and Photonics, 1997.

[73] Thomas Schlegl, Michael Moser, and Hubert Zangl. Wireless and Flexible Ice Detection on Aircraft. Technical report, SAE Technical Paper, 2015.

[74] Shuvo Roy, Alain Izad, Russell G DeAnna, and Mehran Mehregany. Smart Ice Detection Systems Based on Resonant Piezoelectric Transducers. *Sensors and actuators a: physical*, 69(3):243–250, 1998.

[75] James W Melody, T Başar, William R Perkins, and Petros G Voulgaris. Parameter Identification for Inflight Detection and Characterization of Aircraft Icing. *Control Engineering Practice*, 8(9):985–1001, 2000.

[76] Christoph Deiler and Nicolas Fezans. Performance-Based Ice Detection Methodology. *Journal of Aircraft*, 57(2):209–223, 2020.

[77] Baofeng Cheng, Yiqiang Han, Kenneth S Brentner, Jose Palacios, Philip J Morris, David Hanson, and Michael Kinzel. Surface Roughness Effect on Rotor Broadband Noise. *International Journal of Aeroacoustics*, 17(4-5):438–466, 2018.

[78] Xi Chen, Qijun Zhao, George N Barakos, and Alexander Kusyumov. Numerical Analysis of Rotor Aero-Acoustic Characteristics for Ice Detection. *International Journal of Aeroacoustics*, 18(6-7):596–620, 2019.

[79] Andy P Broeren and Michael B Bragg. Effect of Airfoil Geometry on Performance With Simulated Intercycle Ice Accretions. *Journal of Aircraft*, 42(1):121–130, 2005.

[80] Myles Morelli, Beckett Y Zhou, and Alberto Guardone. Acoustic characterization of glaze and rime ice structures on an oscillating airfoil via fully unsteady simulations. *Journal of the American Helicopter Society*, 65(4):1–12, 2020.

[81] Alexis Marbœuf, Marc Budinger, Valérie Pommier-Budinger, Valérian Palanque, and Lokman Bennani. Improving mechanical ice protection systems with topology optimization. *Structural and Multidisciplinary Optimization*, 65(5):1–13, 2022.

[82] Peng Chen and Omar Ghattas. Projected stein variational gradient descent. *Advances in Neural Information Processing Systems*, 33:1947–1958, 2020.

[83] Gori, G., and Zocca, M., and Garabelli, M., and Guardone, A., and Quaranta, G. PoliMIce: A Simulation Framework for Three-Dimensional Ice Accretion. *Applied Mathematics and Computation*, 267:96–107, September 2015. DOI: 10.1016/j.amc.2015.05.081.

[84] Gori, G., and Parma, G., and Zocca, M., and Guardone, A. Local Solution to the Unsteady Stefan Problem for In-Flight Ice Accretion Modeling. *Journal of Aircraft*, 55(1):251–262, September 2017. DOI: https://doi.org/10.2514/1.C034412.

[85] Myles Morelli and Alberto Guardone. A simulation framework for rotorcraft ice accretion and shedding. *Aerospace Science and Technology*, page 107157, 2021.

[86] M. L. Shur, P. R. Spalart, M. K. Strelets, and A. K. Travin. An Enhanced Version of DES with Rapid Transition from RANS to LES in Separated Flows. *Flow, Turbulence and Combustion*, 95(4):709–737, 2015.

[87] K. Kitamura and A. Hashimoto. Reduced Dissipation AUSM-family Fluxes: HR-SLAU2 and HR-AUSM+-up for High Resolution Unsteady Flow Simulations. *Computers & Fluids*, 126:41–57, 2016.

[88] Eduardo Molina. *Detached Eddy Simulation in SU2*. Ph.d. thesis, Aeronautical Institute of Technology, 2015.

[89] E. M. Molina, B. Y. Zhou, J. J. Alonso, M. Righi, and R. G. da Silva. Flow and Noise Predictions Around Tandem Cylinders using DDES approach with SU2. AIAA-2019-0326, 2019.

[90] Eduardo S. Molina, Daniel M. Silva, Andy P. Broeren, Marcello Righi, editor="Hoarau Yannick Alonso, Juan J.", Shia-Hui Peng, Dieter Schwamborn, Alistair Revell, and Charles Mockett. Application of DDES to Iced Airfoil in Stanford University Unstructured (SU2). In *Progress in Hybrid RANS-LES Modelling*, pages 283–293, Cham, 2020. Springer International Publishing.

[91] F. Farassat. Derivation of formulations 1 and 1a of farassat. In *NASA/TM-2007-214853. NASA Langley Research Center*, 2007.

[92] B. Y. Zhou, T. Albring, N. R. Gauger, C. R. Ilario, T. D. Economon, and J. J. Alonso. Reduction of airframe noise components using a discrete adjoint approach. AIAA-2017-3658, 2017.

[93] R. O Icke, O. Baysal, A. Moy, L. Lopes, B. Y. Zhou, and B. Diskin. Toward Adjoint-Based Aeroacoustic Optimization for Propeller and Rotorcraft Applications. In *AVIATION 2020 Forum, AIAA-2020-3140*, 2020.

[94] Yann A. LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient Back-Prop. In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–48. Springer-Verlag Berlin Heidelberg, 2012.

[95] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[96] Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic gradient descent. *ICLR: International Conference on Learning Representations*, 2015.

[97] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, New York, NY, 1996.

[98] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors. *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC, 2011.

[99] Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. Advances in Variational Inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):2008–2026, 2019.

[100] Jaiwon Shin and Thomas H Bond. Experimental and computational ice shapes and resulting drag increase for a naca 0012 airfoil. NASA Technical Memorandum No. 105743, January 1992.

[101] D Dussin, Marco Fossati, Alberto Guardone, and Luigi Vigevano. Hybrid grid generation for two-dimensional high-reynolds flows. *Computers & fluids*, 38(10):1863–1875, 2009.

[102] Tim G Myers. Extension to the Messinger Model for Aircraft Icing. *AIAA journal*, 39(2):211–218, 2001.

[103] Krzysztof Szilder and Edward P. Lozowski. Comparing experimental ice accretions on a swept wing with 3d morphogenetic simulations. *Journal of Aircraft*, 55(6):2546–2549, 2018.

[104] Ari L Frankel, Reese E Jones, Coleman Alleman, and Jeremy A Templeton. Predicting the mechanical response of oligocrystals with deep learning. *Computational Materials Science*, 169:109099, 2019.

[105] Ari Frankel, Kousuke Tachida, and Reese Jones. Prediction of the evolution of the stress field of polycrystals undergoing elastic-plastic deformation with a hybrid neural network model. *Machine Learning: Science and Technology*, 2020.

[106] Ari L Frankel, Cosmin Safta, Coleman Alleman, and Reese Jones. Mesh-based graph convolutional neural networks for modeling materials with microstructure. *Journal of Machine Learning for Modeling and Computing*, 3(1), 2022.

[107] Reese Jones, Cosmin Safta, and Ari Frankel. Deep learning and multi-level featurization of graph representations of microstructural data. *Computational Mechanics*, 72(1):57–75, 2023.

[108] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

[109] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[110] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

[111] Chayan Banerjee, Kien Nguyen, Clinton Fookes, and Maziar Raissi. A survey on physics informed reinforcement learning: Review and open problems. *arXiv preprint arXiv:2309.01909*, 2023.

[112] Ruben Villarreal, Nikolaos N Vlassis, Nhon N Phan, Tommie A Catanach, Reese E Jones, Nathaniel A Trask, Sharlotte LB Kramer, and WaiChing Sun. Design of experiments for the calibration of history-dependent models via deep reinforcement learning and an enhanced kalman filter. *Computational Mechanics*, 72(1):95–124, 2023.

[113] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[114] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[115] Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. *Advances in neural information processing systems*, 32, 2019.

[116] Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. An optimization-centric view on bayes' rule: Reviewing and generalizing variational inference. *The Journal of Machine Learning Research*, 23(1):5789–5897, 2022.

[117] Richard Kurle, Ralf Herbrich, Tim Januschowski, Yuyang Bernie Wang, and Jan Gasthaus. On the detrimental effect of invariances in the likelihood for variational inference. *Advances in Neural Information Processing Systems*, 35:4531–4542, 2022.

[118] Radford M Neal and Radford M Neal. Monte Carlo implementation. *Bayesian learning for neural networks*, pages 55–98, 1996.

[119] Qiang Liu. Stein variational gradient descent as gradient flow. *Advances in neural information processing systems*, 30, 2017.

[120] David Montes de Oca Zapiain and Surya R Kalidindi. Localization models for the plastic response of polycrystalline materials using the material knowledge systems framework. *Modelling and Simulation in Materials Science and Engineering*, 27(7):074008, 2019.

[121] Ari Frankel, Kousuke Tachida, and Reese Jones. Prediction of the evolution of the stress field of polycrystals undergoing elastic-plastic deformation with a hybrid neural network model. *Machine Learning: Science and Technology*, 1(3):035005, 2020.

[122] Nikolaos N Vlassis and WaiChing Sun. Sobolev training of thermodynamic-informed neural networks for interpretable elasto-plasticity models with level set hardening. *Computer Methods in Applied Mechanics and Engineering*, 377:113695, 2021.

[123] David Montes de Oca Zapiain, Hojun Lim, Taejoon Park, and Farhang Pourboghrat. Predicting plastic anisotropy using crystal plasticity and bayesian neural network surrogate models. *Materials Science and Engineering: A*, 833:142472, 2022.

[124] Nikolaos N Vlassis and WaiChing Sun. Geometric learning for computational mechanics part ii: Graph embedding for interpretable multiscale plasticity. *Computer Methods in Applied Mechanics and Engineering*, 404:115768, 2023.

[125] Paul R Dawson. Computational crystal plasticity. *International journal of solids and structures*, 37(1-2):115–130, 2000.

[126] Franz Roters, Philip Eisenlohr, Luc Hantcherli, Denny Dharmawan Tjahjanto, Thomas R Bieler, and Dierk Raabe. Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: Theory, experiments, applications. *Acta Materialia*, 58(4):1152–1211, 2010.

[127] UF Kocks. Laws for work-hardening and low-temperature creep. *Journal of engineering materials and technology*, 98(1):76–85, 1976.

[128] H Mecking, UF Kocks, and H Fischer. Hardening, recovery, and creep in fcc mono-and polycrystals. In *Presented at the 4th Intern. Conf. on Strength of Metals and Alloys, Nancy, 30 Aug.-3 Sep. 1976*, 1976.

[129] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[130] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[131] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[132] Khue-Dung Dang, Matias Quiroz, Robert Kohn, Tran Minh-Ngoc, and Mattias Villani. Hamiltonian Monte Carlo with energy conserving subsampling. *Journal of machine learning research*, 20, 2019.

[133] Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769, 2007.

[134] Gábor J Székely and Maria L Rizzo. Brownian distance covariance. *The annals of applied statistics*, pages 1236–1265, 2009.

[135] Snehal Prabhudesai, Jeremiah Hauth, Dingkun Guo, Arvind Rao, Nikola Banovic, and Xun Huan. Lowering the computational barrier: Partially bayesian neural networks for transparency in medical imaging ai. *Frontiers in Computer Science*, 5, 2023.

[136] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1613–1622, 2015.

[137] Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A. Saurous. Tensorflow distributions, 2017.

[138] Timothy Dozat. Incorporating Nesterov Momentum into Adam. *ICLR Workshop*, 2016.

[139] Haldun Akoglu. User's guide to correlation coefficients. *Turkish Journal of Emergency Medicine*, 18(3):91–93, September 2018.

[140] Marc C Kennedy and Anthony O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

[141] Pranav Rajpurkar, Emma Chen, Oishi Banerjee, and Eric J. Topol. AI in health and medicine. *Nature Medicine*, 28(1):31–38, January 2022.

[142] Martin Kocher, Maximilian I Ruge, Norbert Galldiks, and Philipp Lohmann. Applications of radiomics and machine learning for radiotherapy of malignant brain tumors. *Strahlentherapie und Onkologie : Organ der Deutschen Rontgengesellschaft ... [et al]*, 196(10):856–867, oct 2020.

[143] Umaira Nazar, Muhammad Attique Khan, Ikram Ullah Lali, Hong Lin, Hashim Ali, Imran Ashraf, and Junaid Tariq. Review of Automated Computerized Methods for Brain Tumor Segmentation and Classification. *Current medical imaging*, 16(7):823–834, nov 2020.

[144] Roger Stupp, Warren P Mason, Martin J Van Den Bent, Michael Weller, Barbara Fisher, Martin JB Taphoorn, Karl Belanger, Alba A Brandes, Christine Marosi, Ulrich Bogdahn, et al. Radiotherapy plus concomitant and adjuvant temozolomide for glioblastoma. *New England journal of medicine*, 352(10):987–996, 2005.

[145] Ferlay J Shin HR, F Bray, D Forman, C Mathers, and DM Parkin. Estimates of worldwide burden of cancer in 2008: Globocan 2008. *Int J Cancer*, 127(12):2893–917, 2010.

[146] Fonnet E Bleeker, Remco J Molenaar, and Sieger Leenstra. Recent advances in the molecular understanding of glioblastoma. *Journal of neuro-oncology*, 108(1):11–27, 2012.

[147] Eric J Topol. High-performance medicine: the convergence of human and artificial intelligence. *Nature Medicine*, 25(1):44–56, 2019.

[148] Stan Benjamens, Pranavsingh Dhunnoo, and Bertalan Meskó. The state of artificial intelligence-based FDA-approved medical devices and algorithms: an online database. *npj Digital Medicine*, 3(1), September 2020.

[149] Joep C. Stroom and Ben J.M. Heijmen. Geometrical uncertainties, radiotherapy planning margins, and the ICRU-62 report. *Radiotherapy and Oncology*, 2002.

[150] Mohammad Hesam Hesamian, Wenjing Jia, Xiangjian He, and Paul Kennedy. Deep learning techniques for medical image segmentation: Achievements and challenges. *Journal of digital imaging*, 32(4):582–596, 2019.

[151] Intisar Rizwan I Haque and Jeremiah Neubert. Deep learning approaches to biomedical image segmentation. *Informatics in Medicine Unlocked*, 18:100297, 2020.

[152] Mohammad Havaei, Axel Davy, David Warde-Farley, Antoine Biard, Aaron Courville, Yoshua Bengio, Chris Pal, Pierre-Marc Jodoin, and Hugo Larochelle. Brain tumor segmentation with deep neural networks. *Medical image analysis*, 35:18–31, 2017.

[153] Michael R Kaus, Simon K Warfield, Arya Nabavi, Peter M Black, Ferenc A Jolesz, and Ron Kikinis. Automated segmentation of mr images of brain tumors. *Radiology*, 218(2):586–591, 2001.

[154] Sana Tonekaboni, Shalmali Joshi, Melissa D. McCradden, and Anna Goldenberg. What clinicians want: Contextualizing explainable machine learning for clinical end use. In Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens, editors, *Proceedings of the 4th Machine Learning for Healthcare Conference*, volume 106 of *Proceedings of Machine Learning Research*, pages 359–380, Ann Arbor, Michigan, 09–10 Aug 2019. PMLR.

[155] Adrian Bussone, Simone Stumpf, and Dympna O'Sullivan. The role of explanations on trust and reliance in clinical decision support systems. In *2015 International Conference on Healthcare Informatics*, pages 160–169, 2015.

[156] Eliza Strickland. Ibm watson, heal thyself: How ibm overpromised and underdelivered on ai health care. *IEEE Spectrum*, 56(4):24–31, 2019.

[157] Dakuo Wang, Pattie Maes, Xiangshi Ren, Ben Shneiderman, Yuanchun Shi, and Qianying Wang. Designing ai to work with or for people? In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA, 2021. Association for Computing Machinery.

[158] Saleema Amershi. Toward responsible ai by planning to fail. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3607, New York, NY, USA, 2020. Association for Computing Machinery.

[159] Soumya Ghosh, Q. Vera Liao, Karthikeyan Natesan Ramamurthy, Jiri Navratil, Prasanna Sattigeri, Kush Varshney, and Yunfeng Zhang. Uncertainty quantification 360: A hands-on tutorial. In *5th Joint International Conference on Data Science &; Management of Data (9th ACM IKDD CODS and 27th COMAD)*, CODS-COMAD 2022, page 333–335, New York, NY, USA, 2022. Association for Computing Machinery.

[160] Umang Bhatt, Javier Antorán, Yunfeng Zhang, Q. Vera Liao, Prasanna Sattigeri, Riccardo Fogliato, Gabrielle Melançon, Ranganath Krishnan, Jason Stanley, Omesh Tickoo, Lama Nachman, Rumi Chunara, Madhulika Srikumar, Adrian Weller, and Alice Xiang. Uncertainty as a form of transparency: Measuring, communicating, and using uncertainty. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '21, page 401–413, New York, NY, USA, 2021. Association for Computing Machinery.

[161] Mahima Pushkarna, Andrew Zaldivar, and Oddur Kjartansson. Data cards: Purposeful and transparent dataset documentation for responsible ai, 2022.

[162] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, FAT* '19, page 220–229, New York, NY, USA, 2019. Association for Computing Machinery.

[163] Matthew Arnold, Rachel K. E. Bellamy, Michael Hind, Stephanie Houde, Sameep Mehta, Aleksandra Mojsilovic, Ravi Nair, Karthikeyan Natesan Ramamurthy, Darrell Reimer, Alexandra Olteanu, David Piorkowski, Jason Tsay, and Kush R. Varshney. Factsheets: Increasing trust in ai services through supplier's declarations of conformity, 2018.

[164] Ryan David Bowler, Benjamin Bach, and Larissa Pschetz. Exploring uncertainty in digital scheduling, and the wider implications of unrepresented temporalities in hci. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[165] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[166] Marzyeh Ghassemi, Tristan Naumann, Peter Schulam, Andrew L. Beam, and Rajesh Ranganath. Opportunities in machine learning for healthcare. *CoRR*, abs/1806.00388, 2018.

[167] Effy Vayena, Alessandro Blasimme, and I. Glenn Cohen. Machine learning in medicine: Addressing ethical challenges. *PLoS Medicine*, 2018.

[168] Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1), December 2017.

[169] Maia Jacobs, Jeffrey He, Melanie F. Pradier, Barbara Lam, Andrew C. Ahn, Thomas H. McCoy, Roy H. Perlis, Finale Doshi-Velez, and Krzysztof Z. Gajos. Designing ai for trust and collaboration in time-constrained medical decisions: A sociotechnical lens. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[170] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer New York, New York, NY, 1985.

[171] Jose M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. John Wiley & Sons, New York, NY, 2000.

[172] D. S. Sivia and J. Skilling. *Data Analysis: A Bayesian Tutorial*. Oxford University Press, New York, NY, 2nd edition, 2006.

[173] C. Tsagkaris, A.S. Papazoglou, D.V. Moysidis, M. Papadakis, and C. Trompoukis. Bayesian versus frequentist clinical research now and then: Lessons from the greco-roman medical scholarship. *Ethics, Medicine and Public Health*, 23:100805, August 2022.

[174] David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992.

[175] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, New York, NY, 1996.

[176] Alex Graves. Practical Variational Inference for Neural Networks. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, pages 2348–2356, Granada, Spain, 2011.

[177] Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

[178] Laurent Valentin Jospin, Wray Buntine, Farid Boussaid, Hamid Laga, and Mohammed Bennamoun. Hands-on Bayesian Neural Networks - a Tutorial for Deep Learning Users, 2020.

[179] Carlos Riquelme, George Tucker, Jasper Snoek, and Google Brain. Deep Bayesian Bandits Showdown. *NIPS 2017 Bayesian Deep Learning Workshop*, 2017.

[180] Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. Efficient exploration through Bayesian deep Q-networks. In *2018 Information Theory and Applications Workshop, ITA 2018*, 2018.

[181] Jiaming Zeng, Adam Lesnikowski, and Jose M. Alvarez. The Relevance of Bayesian Layer Positioning to Model Uncertainty in Deep Bayesian Active Learning, 2018.

[182] David Ojika, Bhavesh Patel, G. Anthony Reina, Trent Boyer, Chad Martin, and Prashant Shah. Addressing the memory bottleneck in AI model training. arXiv [preprint] Available at: http://arxiv.org/abs/2003.08732, 2020.

[183] Amber L. Simpson et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv preprint arXiv:1902.09063*, 2019.

[184] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv [preprint] Available at: http://arxiv.org/abs/1412.6980, 2015.

[185] Lee R. Dice. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302, jul 1945.

[186] W.R. Crum, Oscar Camara, and D.L.G. Hill. Generalized Overlap Measures for Evaluation and Validation in Medical Image Analysis. *IEEE Transactions on Medical Imaging*, 25(11):1451–1461, nov 2006.

[187] Michael C. Krygier, Tyler LaBonte, Carianne Martinez, Chance Norris, Krish Sharma, Lincoln N. Collins, Partha P. Mukherjee, and Scott A. Roberts. Quantifying the unknown impact of segmentation uncertainty on image-based simulations. *Nature Communications*, 12(1):5414, September 2021.

[188] Simon K Warfield, Kelly H Zou, and William M Wells. Simultaneous truth and performance level estimation (staple): an algorithm for the validation of image segmentation. *IEEE transactions on medical imaging*, 23(7):903–921, 2004.

[189] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[190] Samuel G. Finlayson, Adarsh Subbaswamy, Karandeep Singh, John Bowers, Annabel Kupke, Jonathan Zittrain, Isaac S. Kohane, and Suchi Saria. The clinician and dataset shift in artificial intelligence. *New England Journal of Medicine*, 385(3):283–286, July 2021.

[191] Anthony O'Hagan, Caitlin E. Buck, Alireza Daneshkhah, J. Richard Eiser, Paul H. Garthwaite, David J. Jenkinson, Jeremy E. Oakley, and Tim Rakow. *Uncertain Judgements: Eliciting Experts' Probabilities*. John Wiley & Sons, Ltd, Chichester, UK, 2006.

[192] Susan L Whitney, Ahmad H Alghadir, and Shahnawaz Anwer. Recent evidence about the effectiveness of vestibular rehabilitation. *Current treatment options in neurology*, 18(3):1–15, 2016.

[193] Courtney D Hall, Susan J Herdman, Susan L Whitney, Stephen P Cass, Richard A Clendaniel, Terry D Fife, Joseph M Furman, Thomas SD Getchius, Joel A Goebel, Neil T Shepard, et al. Vestibular rehabilitation for peripheral vestibular hypofunction: an evidence-based clinical practice guideline: from the american physical therapy association neurology section. *Journal of Neurologic Physical Therapy*, 40(2):124, 2016.

[194] KH Sienko, SL Whitney, WJ Carender, and C Wall III. The role of sensory augmentation for people with vestibular deficits: Real-time balance aid and/or rehabilitation device? *Journal of Vestibular Research*, 27(1):63–76, 2017.

[195] F Daniel Duffy and Eric S Holmboe. Self-assessment in lifelong learning and improving performance in practice: physician know thyself. *Jama*, 296(9):1137–1139, 2006.

[196] Leena Eskelinen, Antti Kohvakka, Tuula Merisalo, Heikki Hurri, and Gustav Wägar. Relationship between the self-assessment and clinical assessment of health status and work ability. *Scandinavian journal of work, environment & health*, pages 40–47, 1991.

[197] Umberto Michelucci and Francesca Venturini. Estimating neural network's performance with bootstrap: A tutorial. *Machine Learning and Knowledge Extraction*, 3(2):357–373, 2021.

[198] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 29, 2016.

[199] Stanley Kaplan and B John Garrick. On the quantitative definition of risk. *Risk analysis*, 1(1):11–27, 1981.

[200] Tiangang Cui, Youssef M. Marzouk, and Karen E. Willcox. Scalable posterior approximations for large-scale Bayesian inverse problems via likelihood-informed parameter and state reduction. *Journal of Computational Physics*, 315:363–387, 2016.

[201] Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E Turner, Jose Miguel Hernandez-Lobato, and Alexander L Gaunt. Deterministic variational inference for robust bayesian neural networks. *arXiv preprint arXiv:1810.03958*, 2018.

[202] Ba-Hien Tran, Simone Rossi, Dimitrios Milios, and Maurizio Filippone. All you need is a good functional prior for bayesian deep learning. *The Journal of Machine Learning Research*, 23(1):3210–3265, 2022.

[203] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[204] Peng Chen, Keyi Wu, Joshua Chen, Tom O'Leary-Roseberry, and Omar Ghattas. Projected stein variational newton: A fast and scalable bayesian inference method in high dimensions. *Advances in Neural Information Processing Systems*, 32, 2019.

[205] Udo Von Toussaint. Bayesian inference in physics. *Reviews of Modern Physics*, 83:943–999, 2011.

[206] Burr Settles. Active Learning Literature Survey. Technical report, University of Wisconsin-Madison, Madison, WI, 2009.

[207] Dawei Zhan and Huanlai Xing. Expected improvement for expensive optimization: a review. *Journal of Global Optimization*, 78(3):507–544, 2020.

[208] Deng Huang, Theodore T Allen, William I Notz, and R Allen Miller. Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32:369–382, 2006.

[209] Kagan Tumer and Joydeep Ghosh. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern recognition*, 29(2):341–348, 1996.

[210] Terje Aven. Bayesian analysis: Critical issues related to its scope and boundaries in a risk context. *Reliability Engineering & System Safety*, 204:107209, 2020.