Review

A Survey of Uncertainty Quantification in Machine Learning for Space Weather Prediction

Talha Siddique [1,†], Md Shaad Mahmud [1,*,†], Amy M. Keesee [2], Chigomezyo M. Ngwira [3] and Hyunju Connor [4]

1 Department of Electrical and Computer Engineering, University of New Hampshire, Durham, NH 03824, USA; talha.siddique@unh.edu
2 Department of Physics & Astronomy and Space Science Center, University of New Hampshire, Durham, NH 03824, USA; amy.keesee@unh.edu
3 Science Group, Atmospheric and Space Technology Research Associate, Louisville, CO 80027, USA; cngwira@astraspace.net
4 Department of Physics and Geophysical Institute, University of Alaska-Fairbanks, Fairbanks, AK 99775, USA; hkconnor@alaska.edu
* Correspondence: mdshaad.mahmud@unh.edu; Tel.: +1-603-862-1358
† These authors contributed equally to this work.

Abstract: With the availability of data and computational technologies in the modern world, machine learning (ML) has emerged as a preferred methodology for data analysis and prediction. While ML holds great promise, the results from such models are not fully unreliable due to the challenges introduced by uncertainty. An ML model generates an optimal solution based on its training data. However, if the uncertainty in the data and the model parameters are not considered, such optimal solutions have a high risk of failure in actual world deployment. This paper surveys the different approaches used in ML to quantify uncertainty. The paper also exhibits the implications of quantifying uncertainty when using ML by performing two case studies with space physics in focus. The first case study consists of the classification of auroral images in predefined labels. In the second case study, the horizontal component of the perturbed magnetic field measured at the Earth's surface was predicted for the study of Geomagnetically Induced Currents (GICs) by training the model using time series data. In both cases, a Bayesian Neural Network (BNN) was trained to generate predictions, along with epistemic and aleatoric uncertainties. Finally, the pros and cons of both Gaussian Process Regression (GPR) models and Bayesian Deep Learning (DL) are weighed. The paper also provides recommendations for the models that need exploration, focusing on space weather prediction.

Keywords: artificial intelligence; uncertainty quantification; deep learning; machine learning; bayesian statistics

Citation: Siddique, T.; Mahmud, M.S.; Keesee, A.M.; Ngwira, C.M.; Connor, H. A Survey of Uncertainty Quantification in Machine Learning for Space Weather Prediction. *Geosciences* **2022**, *12*, 27. https://doi.org/10.3390/geosciences12010027

Academic Editors: Valerie Livina and Jesus Martinez-Frias

Received: 22 November 2021
Accepted: 23 December 2021
Published: 7 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Data are generated and collected through the web and sensors in the modern world due to the increased usage and advancements in electronic devices. The International Data Corporation (IDC) reported that, in 2011, the overall data volume generated in the world was 1.8 Zeta Bytes (ZB), and, within five years, it increased by nearly nine times [1]. Due to this availability and usage of data, the term Big Data has emerged, and a variant of its definition is as such: high volume, velocity, and variety of data that demand cost-effective, innovative forms of processing for enhanced insight and decision-making [2]. Compared with traditional datasets, big data typically include masses of unstructured data that need more real-time analysis [3]. The value chain of big data can be roughly broken down into several phases, i.e., data generation, data acquisition, data storage, data analysis, and development of models that aid in decision-making through the usage of such data [2,3].

Data analysis techniques transform big data into smart data to obtain critical information regarding large datasets. With the advantage of increased data availability in such complex domains and the advancement in computational hardware, there is a growing focus to improve decision-making using machine learning (ML). ML is a branch of artificial intelligence (AI) that is based on the idea that computer systems can learn from data by identifying patterns and then carry out classifications or predictions to aid humans in decision-making [4,5]. At its core, machine learning consists of computational methods or techniques which can be used to solve analytical problems [5]. For instance, ML techniques are being extensively used for the study and detection of space weather phenomena. Ref. [6] (2019) argues that the field of space weather is appropriately suited for ML because of the large and freely available datasets of in situ and remote observations collected over several decades of space missions [6]. Examples include the OMNI dataset, which collects data on plasma and solar quantities and geomagnetic indices in the frequencies of both hours and minutes. In addition to space-based measurements, ground-based magnetometers are used to collect changes in the Earth's magnetic field on time resolutions of a minute and a second (e.g., SuperMAG data). Another ground-based example is the capturing of auroral images using All-Sky Cameras (ASC), which are specialized devices designed to capture images of the full hemispherical sky [7]. Throughout the literature, such datasets have been used to develop and train ML models for Space Weather forecasting, such as solar flare occurrence, coronal mass ejection propagation time, solar wind speed, classification of aurora images, Geomagnetically Induced Currents (GICs), etc. [6].

While ML holds much promise, the results from such models can become unreliable due to the challenges introduced by uncertainty. Uncertainty refers to situations involving unknown or imperfect knowledge and is inherent in stochastic and partially observable environments [8]. Furthermore, uncertainty can be embedded in the entire analytic process (e.g., collecting, organizing, and analyzing big data) [2]. Dealing with incomplete and imprecise information is a critical challenge for most ML techniques [2]. Ref. [9] (2020) focused on how uncertainty impacts the performance of learning from big data, whereas a particular concern lies in mitigating uncertainty inherent within a massive dataset [9]. When scaling these issues up to the big data level, even minor errors accumulate through positive feedback loops and effectively compound any errors in the entire analytic process [2]. Therefore, while an ML model could generate an optimal solution based on its training data, if the uncertainty in the data and the model parameters are not considered, then such optimal solutions have a high risk of real-world deployment failure. Decisions based upon incorrect predictions can cause significant losses; therefore, having confidence in the predicted solution's quality is essential. For the predicted solutions from such models to be reliable, datasets of appropriate size and accuracy are required. Often, despite data availability, the required volume and accuracy cannot be met, and, sometimes, the data themselves are subject to change due to future environmental variability. For example, in space weather prediction, although data on solar flare, solar wind, and changes in the Earth's magnetosphere are available, historical occurrences of events, such as geomagnetic storms, are scarce. Therefore, incorporating data and model parameter uncertainty should be at the forefront during the development of such ML models.

In recent years, there has been increased interest within the scientific community to quantify the uncertainty related to ML models. Figure 1 shows the total number of research publications with the keywords "Machine Learning, Uncertainty Quantification" within 2010–2020. In the context of ML, there are two types of major uncertainty that can be modeled: epistemic uncertainty and aleatoric uncertainty [10–13]. Epistemic uncertainty accounts for the uncertainty in the model parameters. Aleatoric uncertainty is related to the noise inherent in the training dataset itself. The literature consists of examples of different modeling techniques to quantify aleatoric or epistemic uncertainty or both when employing ML models for a specific problem domain. It must be noted that, although quantification of uncertainty can allow its minimization, uncertainty cannot be diminished completely due to the stochastic nature of physical and environmental systems. Uncertainty Quantification (UQ) captures this stochasticity in the form of a probability distribution. Therefore, UQ does not always lead to an increase in accuracy; rather, given the probabilistic nature of UQ, it allows the placement of a confidence interval in the current model output, ensuring reliability.

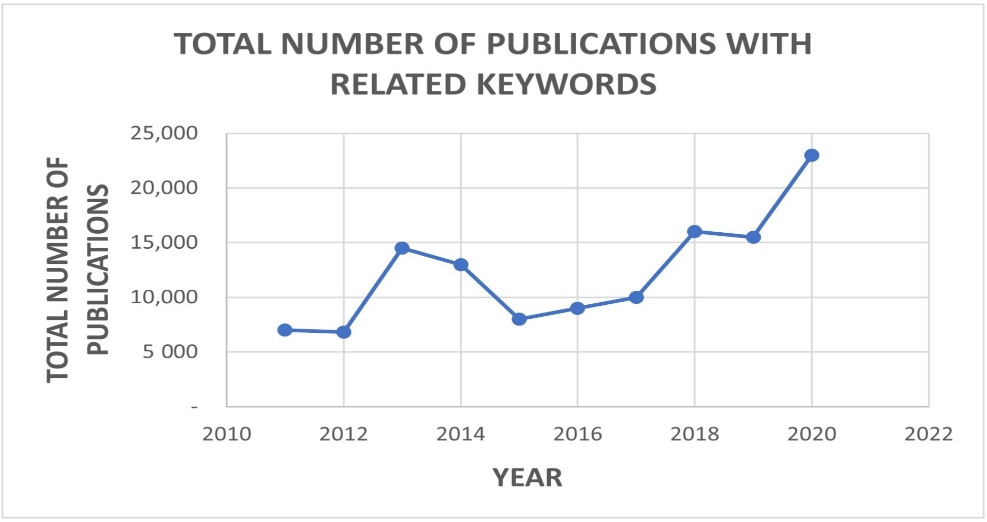

Figure 1. Number of research publications with keywords: "Machine Learning, Uncertainty Quantification", published within the time period of 2010–2020.

Based on the examination of existing research, little work has been done to survey and generalize the different ML methods that exist for quantifying uncertainty. In addition to this, the literature lacks a formal methodology to explicitly quantify uncertainty when applying ML in space weather forecasting. To this end, this paper will provide a survey of the existing methods for Uncertainty Quantification (UQ) using ML models and will exhibit, using two example case studies, how ML methods can quantify uncertainty within the domain of space physics.

The remainder of this paper is arranged in the following manner. Section 2 provides a general overview of UQ, discussing uncertainty, its many dimensions, and its propagation across a model. Section 3 provides an introduction to ML and discusses Gaussian Process Regression (GPR), a popular ML technique for UQ. It also elaborates on how GPR can be used to solve physics-informed models. In Section 4, Deep Learning (DL) and some of its popular architectures are discussed, along with a variant of DL called Physics-Informed Neural Networks (PINN). The section also describes how Bayesian DL can be used to quantify uncertainty. Section 5 presents two case studies, which motivate the use of Bayesian DL for forecasting and uncertainty quantification within the domain of space physics, namely auroral image classification and Geomagnetic Induced Current (GIC) prediction. The section further discusses the pros and cons of each model's approach and the future research possibilities within the context of space physics.

2. Uncertainty Quantification Overview

2.1. Uncertainty and Its Dimensions

In past literature, different definitions of uncertainty can be found specific to the domain in focus. According to Reference [14] (2006), information and concepts of uncertainty are interconnected [14]. This relationship exists because uncertainty manifests due to a deficiency of information, and information is a means to mitigate uncertainty. Uncertainty should not be deemed as an error. The difference lies in that uncertainty is a potential deficiency in the experimental or modeling process due to the lack of knowledge itself, whereas error is a deficiency in the modeling activity not due to a lack of knowledge [10]. Ref. [15] (1991) defines uncertainty along the same line as Reference [14] (2006). They define uncertainty based on the level of available information as a situation of inadequate information. They further classified this level of information into three categories: inexactness, unreliability, and border with ignorance. This level of uncertainty can also increase in situations where there is a surplus of information [16]. The increase is because, when more information is available about the experimental system in focus, the current understanding about the functionalities of the system in focus can come under scrutiny, leading to more uncertainty [17].

Ref. [10] (2003) gives a general definition of uncertainty as being "any departure from the unachievable ideal of complete determinism" [10]. According to them, uncertainty can arise not only because of inadequate knowledge but also due to the system's inherent variability under focus. Uncertainty within a scientific process can be classified into two categories: measurement uncertainty and model uncertainty. Measurement uncertainty is a statistical parameter which describes the possible fluctuations of the result of a measurement collected during an experiment [18,19]. Model uncertainty deals with the imperfections made while formulating a physical system as a mathematical model. It occurs due to limitations in capturing the systems method of functioning and its environmental variability [20,21].

Ref. [10] (2003) wrote a survey on defining uncertainty from the point of view of the modelers [10]. According to them, experts in past literature have agreed on three dimensions of uncertainty. Ref. [10] (2003) generalized these dimensions into: the location of uncertainty, the level of uncertainty, and the nature of uncertainty [10]. The authors made the distinctions with decision support systems for policy prediction in focus. The focus of this survey paper is uncertainty quantification in ML models, and this paper deems ML models as a subset of the generic decision support systems described by Reference [10] (2003). As such, this paper is adopting these distinctions with the focus of ML models in mind.

Location of uncertainty is an identification of where uncertainty can manifest within the system which is being modeled. This particular dimension is an umbrella term for the different sources of uncertainties that occur due to limitations in the instruments or the observations which are used for data collection; the inherent variability in the environmental systems and, therefore, in the collected data; the structural or mathematical relationships used to represent the physical system in focus; and calibration and estimation of the model parameters which is dependent on the collected uncertain data or the mathematical constraints. Modern instrumentation systems rarely report or display uncertainty explicitly. Usually, measurement and calibration uncertainties are interpreted using records from instrument handbooks and calibration records. For example, the time period of capturing aurora images by All-Sky Camera (ASC) devices is limited by 2$\times$ the exposure time [7]. The measurement using the ASC is limited by instrumental issues, such as misting of the camera lens; and environmental variability, such as local weather conditions (e.g., rain, snow, or cloudy weather), moon or sun positions, etc. [22]. These factors can lead to a noisy dataset. For instance, sensors in the ASC allow pictures to be taken at different time exposures and signal amplifications. The ASC takes a set of raw images at a specified time period to have enough pixel signals, and these pixel signals are prone to uncertainty [7]. As mentioned above, uncertainty can also arise due to model structure, i.e., the assumptions that are made about the constraints and the relationship between the model variables during implementation. According to Reference [23] (2014), a model has structural uncertainty if the model dynamics differ from the dynamics in the target system [23]. They further elaborate that, especially for non-linear systems, small differences in the choice of model structure can compromise the ability to generate decision-relevant predictions. While training, the model parameter estimation depends on the noisy dataset, as well as the model structure; therefore, the parameters themselves can suffer from uncertainty. All of these uncertainties accumulate and propagate forward into the prediction made by the ML model.

Ref. [10] (2003) expands on the three levels of uncertainty defined by Reference [15] (1991). In the context of ML, the two most relevant levels are determinism and statistical uncertainty. Determinism is a scenario where everything about the system being modeled is known. Although determinism is unattainable in real life, the literature does consist of deterministic ML models, which are developed for their simplicity [24,25]. In ML, statistical uncertainty can deal with the quantification of probability in a stochastic model. It can also deal with measurement uncertainties, such as sampling error, where the sample dataset does not completely represent the population in focus.

In addition to the level of uncertainty, the nature of uncertainty must also be taken into consideration [10]. In the context of ML, knowledge or information can be categorized into two factors: the background knowledge required to determine the model structure and the dataset which is used to train the ML models. This nature of uncertainty can be classified into two categories within the context of ML [10,13,26]:

- Epistemic uncertainty (parameter uncertainty): This uncertainty accounts for the uncertainty in the model parameters. With a better understanding of the model structure, accurate model relationships and constraints can be captured. In addition, with more data, the model can be trained better; therefore, this kind of uncertainty can be reduced with more knowledge about the system in focus. Moreover, epistemic uncertainty is higher in regions with little or no training data and relatively lower in regions with more training data.
- Aleatoric uncertainty (data uncertainty): This uncertainty is related to the noise inherent in the training dataset itself. Such uncertainty cannot be reduced, even if we get more data to train the model.

These two natures of uncertainty capture the different uncertainties contributed by the sources discussed above. The presence of epistemic uncertainty indicates whether the model will require more data for better training. It also indicates whether the structure of the model is a sound representation of the system. Aleatoric uncertainty tells us about the noise level in the dataset, which can be due to the reasons elaborated above.

2.2. Uncertainty Quantification and Propagation

Uncertainty quantification (UQ) is the end to end study of the reliability of scientific inferences (U.S. Department of Energy, 2009). UQ in ML primarily deals with model uncertainty over measurement uncertainty [20]. In the context of modeling, it is concerned with the estimation of the impact that uncertain input data has on the model parameter and prediction [26]. UQ covers the different dimensions of uncertainty and aims to enhance the model's reliability by producing the output in a probabilistic framework, where a confidence interval can be placed to estimate the model's robustness [26]. UQ problems usually consist of a mathematical model representing the system in focus, subject to uncertainties through the input values and estimated model parameters. It also involves identifying the propagation of these uncertainties through the entire model. These uncertainties are then quantified via the means of a probabilistic framework [27].

In past literature, the propagation of uncertainty across the model is addressed through forward modeling and backward modeling [26,28,29]. Forward modeling, or forward propagation problem, deals with the use of input data and parameter estimates to predict an output variable [29,30]. The model's accuracy is determined by calculating a difference metric, such as Mean Squared Error (MSE), of the predicted outcome against test or observation data. For such a modeling approach, sensitivity analysis is a means to determine the effects of uncertain parameter estimates on model predictions [26,28]. On the other hand, backward modeling or inverse propagation problem deals with the optimal setup for the input data and the model parameters so that the model generates predictions of a certain desired accuracy [26,29]. The models in such settings achieve this by using a form of optimization technique. The observational or test data is used to refine the data and parameter uncertainties [28]. Typically, simulation models are set up as a forward propagation problem, and ML models are set up as a backward propagation problem. UQ in backward propagation problems are generally classified based on their underlying concept of probability inference: Frequentist or Bayesian. An example of Frequentist inference for UQ in ML is the standard error of the parameter estimates in least square regression problems can be represented as a quantification of parameter uncertainty [31]. Bayesian inference for ML deals with Bayes theorem and is elaborated in the next section.

3. Machine Learning and Uncertainty Quantification

3.1. Overview of Machine Learning

In recent years, machine learning (ML) methods have been adopted across various studies, specifically for data analysis, prediction, and classification-based tasks. As a discipline, ML tackles the question of how computer systems can learn from experience. It also deals with the statistical, informational, and theoretical laws that govern the learning process in humans, machines, and physical systems [32]. ML algorithms follow different learning strategies. These learning strategies can roughly be classified into three classes: supervised learning, unsupervised learning, and semi-supervised learning [33]. Supervised learning systems generally form their predictions via a learned mapping $f(x)$ [32,34]. The system learns from example inputs and outputs from past (training datasets) relevant to the problem in focus. The goal of the ML system is to learn a general rule that maps inputs x to outputs y or a probability distribution over y given x. This mapping of f is done by different ML algorithms, e.g., linear or logistic regression, decision trees, neural networks, etc. In the case of unsupervised learning, in contrast to supervised learning, the examples in the training dataset do not consist of any labels [32]. A popular use of unsupervised learning is to discover hidden patterns in the data. It can also be used for feature learning, where the algorithm detects the variables or features from the dataset which best represents the system in focus. Although these two categories of learning are widely in use, there are still applications that require a blend across these categories. Semi-supervised learning uses unlabeled data to supplement labeled data in a supervised learning context. It can also blend algorithms developed for unsupervised learning with optimization methods that use training labels to determine decision boundaries for classification or regression.

The application and success of ML techniques depend heavily on datasets, algorithms, and layered structures, so it is necessary to quantify the uncertainty contributions of each component in ML modeling [35]. In ML modeling, quantifying uncertainty is a crucial task to improve the validity and predictability of ML model applications. According to Reference [36] (2020), uncertainty sources concerning ML can be mainly divided into three types: sample set partitioning, ML approach selection, and ML architectures design [36]. Studies in the previous literature usually focus on a single source of uncertainty.

3.2. Gaussian Process Regression for Uncertainty Quantification

Ref. [37] (2019) investigated the impact of dataset uncertainties on the performance of the ML model predictions [37]. They focused on the case study of the glass transition temperature of polymers. The glass transition temperature is a material property that exhibits large variations during the experimental measurement process. The authors curated a training dataset consisting of the glass transition temperature of 751 polymers. They adopted the ML method called Gaussian Process Regression (GPR), which was used in a similar study by Reference [38]. GPR is a non-parametric, Bayesian approach to regression, and its advantage over other supervised learning is that it can work well on small datasets and provide uncertainty measurements on the predictions.

Given a functional form, $y = wx + \epsilon$, traditional or non-Bayesian supervised machine learning algorithms give a point estimate for the parameter w. As opposed to this, the Bayesian approach specifies a posterior probability distribution $p(w)$, on the parameter w based on the observed data. The Bayesian theorem states the following:

$$p(w|y,x) = \frac{p(y|x,w)p(w)}{p(y|x)}. \quad (1)$$

The updated posterior distribution $p(w|y,x)$ incorporates information from both the prior distribution $p(w)$ and the dataset $D(x,y)$. The terms $p(y|x,w)$ and $p(y|x)$ represent the likelihood and marginal likelihood, respectively. If the test observation is x^*, and the prediction label is f^*, the predictive distribution can be calculated as:

$$p(f^*|x^*, y, x) = \int p(f^*|x^*, w)p(w|y, x)dw. \tag{2}$$

For the integration to be tractable, the prior and likelihood are generally assumed to be Gaussian. Solving for the predictive distribution generates a Gaussian distribution. The mean of the distribution represents a point estimate for the model parameter, and the variance of the distribution is a quantification of the parameter uncertainty.

GPR is nonparametric and, therefore, does not restrict the probability distribution of parameters to a specific functional form. In GPR, a range of functions that fit the data are used to calculate the probability distribution. A Gaussian process prior is first assumed using a mean function, $m(x)$, and covariance function, $k(x, x')$. The labels that are drawn from the mean and covariance function can be denoted as:

$$f(x) \sim GP(m(x), k(x, x')). \tag{3}$$

In a Gaussian process, the labels from the dataset are joint Gaussian distributed. The mean and covariance functions can be used to incorporate the Gaussian process prior. When the process is assumed to be independently, identically distributed (i.i.d), the functional form is as follows:

$$f(x) \sim GP(m(x), k(x, x') + \delta\sigma^2). \tag{4}$$

The function $k(x, x')$ represents the covariance kernel. The functional form of mean and covariance is selected during the model selection step. Usually, the mean function is a constant based on the mean of the training dataset. The covariance kernel can take various functional forms depending on the need (e.g., linear, square, exponential, etc.), but one of the most widely used kernels is the radial basis function (RBF) kernel [39]. The RBF kernel has the following functional form:

$$k(x, x') = \delta^2 exp(-\frac{1}{2l^2}||x - x'||^2). \tag{5}$$

The RBF kernel has two hyperparameters: the signal variance, δ^2, and lengthscale, l. Hyperparameters are parameters whose values are tuned to control the learning process. In order to tune the hyperparameters of the RBF kernel function, the log marginal likelihood of the training data can be optimized. As mentioned above, since the prior is Gaussian, it is tractable to estimate the predictive distribution. This generates a normal distribution with mean, $\bar{f}^*$, and covariance matrix, Σ^*. The variances which quantify the uncertainty can be obtained from the diagonal of the covariance matrix. The accuracy of the model is calculated using the root mean square error (RMSE). The normal distribution which is obtained is as follows:

$$f^*|x, y, x^* \sim N(\bar{f}^*, \Sigma^*). \tag{6}$$

3.3. Physics-Informed Models and Gaussian Process Regression

In past literature, there has been an overlap between the application of Gaussian process (GP) models with deterministic and stochastic differential equations in linear form [40]. Ref. [41] (2011) proposed a general framework of how linear differential equations can be incorporated into GPR to encode physical and other background information into the models [41]. In physics, the models of nature (e.g., law of electromagnetism) appear as partial differential equations (PDE) [42]. When PDE models include stochastic terms, they become stochastic partial differential equations (SPDE). SPDE can be used to model spatial, physical processes with unknown sub-phenomena [43]. An SPDE has a solution of a GP when a GP drives the linear PDE.

Based on the approach proposed by Reference [41], to incorporate linear PDE into the GP, we take the general form of the GPR as that of Equation (3). Let $y = Hf(x) + \epsilon$, where H is a deterministic linear function, and the objective is to estimate the linear operator transformation of the posterior $d = \mathcal{L}f(x)$, where $\mathcal{L}$ is a linear operator. Here, y denotes the target value from the dataset. The error term ϵ is i.i.d, and $\epsilon \sim N(0, \Sigma)$, where Σ represents the joint covariance matrix. The joint distribution of y and the posterior $d(x)$ is a Gaussian with zero mean and covariance:

$$Cov\begin{bmatrix} y \\ d(x) \end{bmatrix} = \begin{bmatrix} Hk(x,x')H^T & \Sigma Hk(x,x')\mathcal{L}^T \\ \mathcal{L}k(x,x')H^T & \mathcal{L}k(x,x')\mathcal{L}^T \end{bmatrix}. \tag{7}$$

The conditional mean and covariance of the posterior $d(x)$ can be calculated as follows:

$$E[d(x)|y] = \mathcal{L}k(x,x')H^T[Hk(x,x')H^T + \Sigma]^{-1}y, \tag{8}$$

$$Cov[d(x)|y] = \mathcal{L}k(x,x')\mathcal{L}^T - \mathcal{L}k(x,x')H^T[Hk(x,x')H^T + \Sigma]^{-1}Hk(x,x')\mathcal{L}^T. \tag{9}$$

Moving on to the case of SPDEs, to motivate the problem, the functional form of the GPR model and y is assumed to be the same as before. Let $Dg(x) = f(x)$ represent the linear SPDE, where D is a linear differential operator, and $f(x)$ is a GP. However, this SPDE is incompatible with the GP model in this form. Therefore, Reference [41] introduced the Green's function $G(x,x')$, which is the inverse of the differential operator D. Therefore, the compatible form of the function y in the GPR model is as follows:

$$y = H\int G(x,x')f(x')dx' + \epsilon. \tag{10}$$

The estimation of the mean and covariance of the posterior can be performed using a formulation similar to PDEs.

Ref. [44] (2018) proposed the numerical GP method, which is defined as a Gaussian processes whose covariance function is determined by the temporal discretization of the time dependent PDEs [44]. Their method placed GP priors on spatial fields and quantified uncertainties in the solutions of the PDEs.

Probabilistic numeric methods (PNM) have been used with GPs to solve PDEs, along with uncertainty quantification. PNM deals with the quantification of errors and uncertainties arising from precision loss due to the limitations of time and hardware [45]. For example, Reference [46] (2016) proposed a variant called the probabilistic mesh-free method (PMM) for quantifying uncertainty over the solution space of linear PDEs [46]. In their model, the GP prior is constrained by the PDE dynamics and boundary conditions by a finite set of co-location points. In their work, the PMM method quantifies uncertainties in both the forward and inverse problems.

GPs have also been used to estimate uncertainties in the parameters of Ordinary Differential Equations (ODEs). ODEs have been used throughout different domains to model the behavior of a natural system [47]. Although the modelers have sufficient knowledge to model the natural system, the parameters estimated for the ODEs tend to suffer from uncertainty. In past literature, parameter inference methods based on experimental data have been proposed to deal with such uncertainty [48–50]. Ref. [47] (2013) proposed an inference scheme called the Adaptive Gradient Matching (AGM) technique [47]. In their scheme, both the hyperparameters of the GP and the ODE parameters are jointly inferred from the posterior distribution by taking into account their correlation. The GP model selection is adapted during the inference based on information from the ODE system.

4. Uncertainty Quantification, Neural Networks and Deep Learning

4.1. Overview of Deep Learning

Deep Learning (DL) is a subfield of ML [51]. The learning mechanism used by DL is inspired by the biological structure, which allows the human brain to process information. The human brain uses biological neurons to identify patterns and classify different types of information. Similarly, in DL, Neural Networks (NN) or Artificial Neural Networks (ANN) are used to simulate similar tasks on data. A NN generally consists of layers of connected nodes called neurons. It can be thought of as a graphical representation, where the layers of neurons are interconnected via weights, and both neurons and weights are numerical values. Usually, an NN consists of several layers. The first layer is the input layer, and the last layer is called the output layer. The number of neurons in the input layer is equal to the number of the feature variables, and the number of neurons in the output layer reflects the target variable [52]. The layers in between are called hidden layers. The goal of the NN is to attain the optimal set of weights for optimal connection between the layers, which, in turn, will ensure the most accurate prediction. During training, the feature variable data are fed into the input layer. The hidden layers consists of an activation function $f()$, which maps the dot product of the input $\vec{x}$, and the weight $\vec{w}$, to $\vec{h}$. The most common forms of activation functions are tanh, sigmoid, and Rectified Linear Unit (ReLU) [53]. The mathematical representation is as follows:

$$\vec{z} = \vec{x} \cdot \vec{w}, \tag{11}$$

$$\vec{h} = f(\vec{z}). \tag{12}$$

Forward propagation continues through the consecutive hidden layers until it reaches the output layer to generate the intermediate predicted target vector, $\vec{\hat{y}}$. This $\vec{\hat{y}}$ is compared against the ground truth of the target vector, $\vec{y}$, from the training dataset. The comparison is performed using a loss function. Two common types of loss functions are cross-entropy and mean-squared error (MSE) loss. The idea is to update the weights through backward propagation until the loss is minimized. The mathematical formulation of both the loss functions are exhibited below:

$$\mathcal{L}(w) = -\frac{1}{N}\sum_{i=1}^{N}(y_i \cdot \log \hat{y}_i), \tag{13}$$

$$\mathcal{L}(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \log \hat{y}_i)^2. \tag{14}$$

The choice of loss function depends on the task in focus. Minimizing the loss function gives the optimal set of weights $\vec{w}$, which, in turn, generates the most accurate predictions. The loss function is minimized through a mathematical process known as gradient descent. The loss $\mathcal{L}$ is a function of a particular weight w, and the objective is to determine the value of w, which gives the minimum loss. If our initial weight is $w_{initial}$, and the subsequent weight descending towards the minimum point of the loss function is w_{next}, then the gradient descent step is as follows:

$$w_{next} = w_{initial} - \epsilon \nabla_{w_{inital}} \mathcal{L}(w_{inital}). \tag{15}$$

The learning rate, ϵ, is a hyperparameter that determines how quickly the model will update the weight parameters. With each gradient descent step, the weight parameters are updated towards a more optimal value, leading to better final predictions.

There are various forms of network architectures used in DL. The choice of an appropriate DL approach is one of the most prominent activities for time-series forecasting [54]. A widely used DL architecture is Convolutional Neural Networks (CNNs), which are primarily used to solve image-driven pattern recognition tasks [55,56]. Through the application of relevant filters, CNNs can capture the spatial dimensionality of an image. In addition to the input layer, a CNN consists of three additional layers, stacked one after the other: the convolutional layer, the pooling layer, and fully connected layers. The input is in the form of an algebraic object, known as a tensor. A convolution layer abstracts the input to a feature map consisting of the shape (number of images) × (feature map height) × (feature map width) × (number of feature map channels). The objective of the convolutional layer is to extract high-level features, such as the edges of the input. The next layer is the pooling layer, where the feature map is further reduced in size to extract the most relevant features [57]. The fully connected layer flattens the reduced feature map into column vectors, and these feature vectors are then passed to a feed-forward NN, i.e., an NN where there is no feedback loop. The NN then classifies the images, usually using a softmax function, also known as a softmax classifier. Usually, cross-entropy (Equation (13)) is used as a loss function, and it is optimized to increase the likelihood of the class or label of the correct image. The functional form of softmax is exhibited below, where, given the input parameter matrix θ, the objective is to see if the trained feature vector x, along with its set of weights, belongs to an image class label j.

$$p(y = j|\theta) = \frac{\epsilon_\theta}{\sum\limits_{j=0}^{k} \epsilon_{\theta_j}}. \tag{16}$$

4.2. *Physics-Informed Models and Deep Learning*

In past literature, variants of DL models have been developed to solve physics-based or physics-informed models. These models, in their rudimentary NN forms, are known as Physics-Informed Neural Networks (PINNs). As discussed in the previous section, physics-based models are used to model natural systems using PDEs and SPDEs. This paper adopts the example given by Reference [58]. A form of PDE called Burgers equation is given below in one space dimension:

$$u_t + \lambda_1 u u_x - \lambda_1 u u_{xx} = 0. \tag{17}$$

Approximating $u(t,x)$ by an NN results in a PINN of the form $f(t,x)$, which is exhibited below:

$$f(t,x) = u_t + \lambda_1 u u_x - \lambda_1 u u_{xx}. \tag{18}$$

By minimizing the mean-squared error loss (MSE) (Equation (14)), the shared parameters of the NN, $u(t,x)$ and $f(t,x)$, along with the parameters of the PDE, λ_1 and λ_2, can be learned. The combined MSE for the PINN is given in Equation (19), where MSE_u corresponds to the training data of $u(t,x)$, while MSE_f ensures the structure imposed by Equation (18).

$$MSE = MSE_u + MSE_f. \tag{19}$$

4.3. *Bayesian Deep Learning and Uncertainty Quantification*

Despite the potential of such architectures, these models in their traditional form cannot consider uncertainty. In order for data and parameter uncertainty to be quantified, Bayesian inference is required. Bayesian DL in its most basic form consists of a Bayesian Neural Network (BNN). In the case of a classical Neural Network (NN), the weights are a point estimate. However, in BNN, using Bayesian statistics (Equation (1)), a probability distribution is assigned as the weights. The probability distributions capture the uncertainty in the data [13,59]. A BNN can be viewed as probabilistic model $p(y|x,w)$. Here, y is the variable to be predicted; x is the set of features; w is the weight parameter and $p(y|x,w)$. Given the training dataset $D = \{x,y\}$, the likelihood function can be constructed, which is a function of the parameter w [13]. The likelihood function is as follows:

$$p(D|w) = \prod p(y|x,w). \tag{20}$$

The maximum likelihood estimate (MLE) of w is obtained by maximizing the likelihood function, where the objective function is the negative log-likelihood. However, MLE gives point estimates for parameters due to which the uncertainty over the weights cannot be captured. Therefore, BNN averages predictions over a combination of NN, which are weighted by the posterior distribution of the parameter w. The mathematical formulation of the posterior predictive distribution is as follows:

$$p(y|x, D) = \int p(y|x, w)p(w|D)dw. \tag{21}$$

The determination of the posterior distribution, $p(w|D)$ is intractable. There are two common methods to approximate the probability distribution: Variational Inference (VI) and Markov Chain Monte Carlo (MCMC). In the case of VI, BNNs can use a variational distribution $q(w|\theta)$ of known functional form (e.g., Gaussian distribution) to approximate the true posterior distribution. In order to achieve this, the Kullback–Leibler (KL) divergence between $q(w|\theta)$ and the true posterior $p(w|D)$ w.r.t. θ is minimized. The corresponding objective function is as follows:

$$\mathrm{KL}(q(w|\theta)||p(w|D)) = \mathbb{E}[\log q(w|\theta)] - \mathbb{E}[\log p(w)] - \mathbb{E}[\log p(D|w)] + \log p(D). \tag{22}$$

Since the KL cannot be calculated, the negative of the KL divergence function called the evidence lower bound (ELBO) is used [13]. The ELBO does not contain the term $\log p(D)$. Since $\log p(D)$ is a constant term, it can be discarded; therefore, maximizing the ELBO function is equivalent to minimizing the KL divergence. The functional form of the ELBO function is stated below.

$$\mathrm{ELBO}(q) = \mathbb{E}[\log p(w)] + \mathbb{E}[\log p(D|w)] - \mathbb{E}[\log q(w|\theta)]. \tag{23}$$

Another popular approach to the approximate posterior distribution of the parameter in focus is MCMC methods [60,61]. It is a class of methods that achieves the approximation by random sampling in a probabilistic space [62]. The MCMC method can roughly be broken down into three steps: Markov Chain, Monte Carlo, and Transitions. Markov Chains can be thought of as a means for a random variable to change its state over time in a graph. Markov Chains follows the Markov property, which dictates that, if the current state is known, transitions to future states are independent of the states in the past. An example of a functional form that follows the Markov property is given below:

$$f(\theta_{t+1}|\theta_t) = f(\theta_{t+1}|\theta_t, \theta_{t-1}, \theta_{t-2}, \ldots). \tag{24}$$

During the Monte Carlo simulation, the model draws the values of θ from an arbitrary distribution, with a mean equal to the previous value of θ [62]. The transition step determines whether the sampled observation is in the correct direction and whether it should be accepted. A commonly used algorithm for this transition step is Metropolis-Hasting (M-H). The M-H algorithm calculates the acceptance probability $r(\theta_{new}, \theta_{t-1})$. The model accepts the new proposed value θ_{new} if the $min(r(\theta_{new}, \theta_{t-1}))$ equals 1. The acceptance probability is a ratio which is given below:

$$r(\theta_{new}, \theta_{t-1}) = \frac{PosteriorMean(\theta_{new})}{PosteriorMean(\theta_{t-1})}. \tag{25}$$

In the BNN, the distribution has the parameters $\theta = (\mu, \sigma)$, where μ and σ are the distribution's mean and standard deviation vector. Since σ consists of elements of a diagonal covariance matrix, the weights of the BNN can be assumed to be uncorrelated. As the model is parameterized with μ and σ, a BNN ends up with a more significant number of parameters compared to a classical NN.

As mentioned in Section 2 of this paper, there are two types of uncertainty: epistemic uncertainty and aleatoric uncertainty. The posterior weight distribution captures the epistemic uncertainty, and the likelihood function captures the aleatoric uncertainty. The UQ concepts relevant to ML/DL (backward propagation problem) that were discussed over this and former sections (Sections 3 and 4) are summarized in Figure 2.

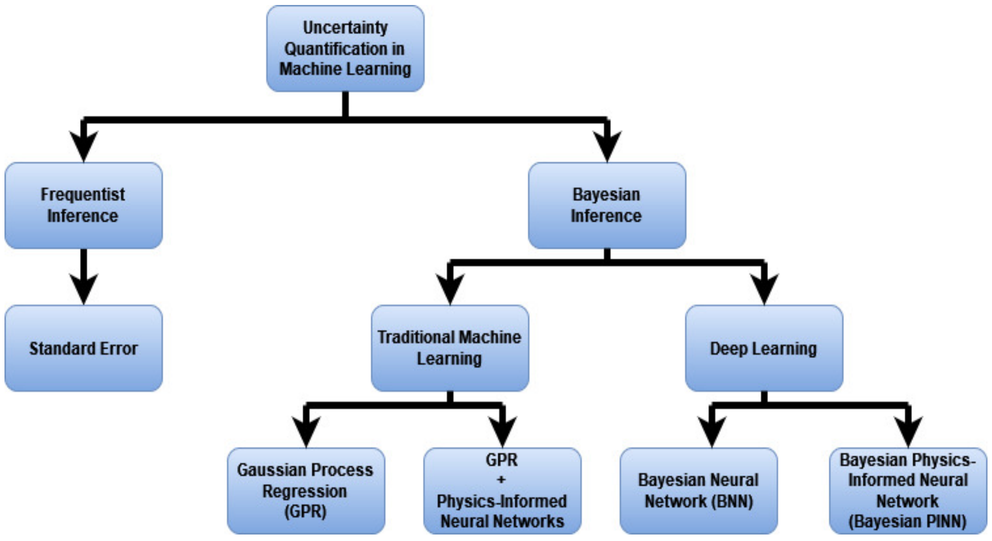
Uncertainty Quantification in Machine Learning
Frequentist Inference
Bayesian Inference
Standard Error
Traditional Machine Learning
Deep Learning
Gaussian Process Regression (GPR)
GPR + Physics-Informed Neural Networks
Bayesian Neural Network (BNN)
Bayesian Physics-Informed Neural Network (Bayesian PINN)

Figure 2. A graphical representation of uncertainty quantification models in machine learning.

5. Case Studies: Machine Learning in Space Physics Forecasting and the Need for Uncertainty Quantification

In past literature, ML techniques have been applied to study space physics phenomena, such as aurora and GICs. This is to better understand the complex space environment between the sun and the Earth, which, in turn, can help predict and mitigate the adverse effects it poses on human technology [63,64].

Both aurora and GICs are manifestations caused by geomagnetic storms [64]. A geomagnetic storm is a major disturbance of Earth's magnetosphere (the region of space dominated by Earth's magnetic field) that originates from solar disturbances [63,65,66]. At times, Coronal Mass Ejections, an explosive outburst of plasma (ionized gas) and magnetic fields from the solar surface, is released toward the Earth, causing a geomagnetic storm. When the CME comes with southward magnetic fields, it strongly interacts with the dayside of the Earth's magnetosphere through a physical process called magnetic reconnection [67–69]. This reconnection transfers energy from CME to the dayside magnetosphere. A portion of this energy is directly deposited into the dayside upper atmosphere, causing the dayside aurora. However, most of it continues to flow toward the nightside magnetosphere and then, after the nightside reconnection, enters the inner magnetosphere (the region in 6 earth radii radial distance) and the nightside upper atmosphere, causing strong geomagnetic disturbances and luminous nightside aurora.

The interaction of the CME with the Earth's geomagnetic field (the B field) increases currents in both the magnetosphere and the ionosphere (the Earth's atmospheric layer at an altitude of 80$\sim$1000 km that contains a high concentration of charged particles). The currents cause fluctuations in the B field on Earth's surface. Faraday's law of induction dictates that changing magnetic fields induce electric currents in conductors. Rapid changes in the B field, in the presence of Earth conductive structure, generate geoelectric fields (E fields) that drive GICs through these conductors.

The flow of GIC into power transmission lines can lead to other subsequent effects, such as relay misoperations, voltage dips, elevated reactive power demand, transformer overheating, disruptive harmonics, or a failure of the electric power devices [70]. In the worst-case scenario, it can even lead to a malfunction of the complete power grid. For example, on 13 March 1989, a strong Geomagnetic Disturbance (GMD) event or geostorm caused major disruptions to electrical equipment across Canada, Scandinavia, and the United States [9].

However, despite the extensive use of ML to predict and study space weather phenomena, little work exists which explores or reviews the need for UQ in this domain. To bridge this gap, two separate off-the-shelf Bayesian DL methods, which were reviewed in Section 4, have been implemented to classify aurora images and predict GIC values while quantifying uncertainty, respectively.

5.1. Machine Learning/Deep Learning Methods for Auroral Image Classification and GIC Prediction

Past literature consists of studies where ML techniques have been used to classify aurora images. For example, Reference [71] (2014) used Support Vector Machines (SVM) to classify aurora images into three predefined labels: aurora, no aurora, and cloudy [71]. They compared the accuracy of the SVM using different feature extraction methods. They used a texture-based feature extraction technique called Local Binary Pattern (LBP) and a point description technique called Scale Invariant Feature Transform (SIFT). Their study found that the SVM performs best with the SIFT feature extraction method. Ref. [72] (2012) used a Hidden Markov Model (HMM) to classify auroral images into four categories: arc, hot spot, drapery, and radial [72]. Due to the evolving nature of aurora over time, they explicitly included this temporal characteristic by using sequences of auroral images.

Deep Learning methods have also been used to classify auroral images. In fact, with the focus of auroral image classification, DL architectures have generally outperformed traditional ML approaches, such as SVM methods [73]. Ref. [74] (2020) used CNNs to classify the auroral images into the same labels as that of Reference [72]. First, they compared the performance of three CNN variants used for auroral image classification in past literature: AlexNet, VGG-16, and Inception-v4. Due to its superiority in auroral image classification, AlexNet was chosen as the backbone for their CNN. The Spatial Transformer Network (STN) was installed before the AlexNet, as STN can adaptively transform and align the images specific to the classification task. Finally, they used a Large Margin Softmax (L-Softmax) as their loss function. In the literature, the DL methods aim to segment the auroral images into aurora pixels from the background of the sky [75]. However, when human experts classify auroras, they focus on the Key Local Structures (KLS) from the images, e.g., arc, band, ray structures, etc. To this end, Reference [76] (2019) implemented a variant of DL called Cycle-consistent Adversarial Network (CAN) to classify auroral images by extracting these KLS [76]. ML/DL techniques have been used extensively to classify dayside auroral images. However, the same models cannot be applied to classify nightside auroral images due to the evolution of the auroral images during nighttime [73]. Therefore, Ref [77] (2018) classified nightside auroral images into six labels: arc, diffuse, discrete, cloudy, moon, and clear/non-aurora [77]. They used a pre-trained Deep NN to extract the features from the images. The extracted features were then used to classify the images using a ridge classifier. Based on this study, Reference [73] (2020) classified colored (RGB) nighttime auroral images with higher accuracy into a greater number of classification labels: arc aurora, auroral breakup, colored aurora, discrete irregular, patchy aurora, edge aurora, and faint clear [73]. They compared different Deep NN architectures, including the one used by Reference [77] (2018). The Deep NN architectures implemented by them were VGG, AlexNet, and ResNet. The authors found ResNet to give the highest accuracy on the given dataset, compared to the other architectures.

For the case of GIC prediction or detection, Reference [9] (2020) used feature extraction based on waveform analysis, along with DL, to detect GIC [9]. In their hybrid model, they incorporated two time-frequency analysis techniques, Wavelet Transforms (WT) and Short-Time Fourier Transform (STFT), with CNN. Ref. [78] (2015) developed a model to predict the 30 min maximum perturbation of the horizontal magnetic component, $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ [78]. They used a variant of NN called the Elman NN (Elman 1990). Ref. [79] (2020) implemented two separate models, namely a feed-forward NN and a Long-Short Term Memory (LSTM) NN, to predict the East and North component of the geomagnetic field, which, in turn, was used to derive the horizontal component and its change over time [79]. They proceeded to compare the accuracy of the two models. The literature discussed in this section are summarized in Table 1 below.

Table 1. Summary of ML models used for auroral image classification or GIC prediction.

Literature Citation	Implemented Model	Objective
[71] (2014)	Support Vector Machines (SVM); Used feature selection methods: Local Binary Pattern (LBP) and Scale Invariant Feature Transform (SIFT)	Auroral image classification into 3 labels
[72] (2012)	Hidden Markov Model (HMM)	Auroral image classification into 4 categories; Used Sequences of Auroral images to capture temporal properties
[74] (2020)	Three Convolutional Neural (CNN) Network Variants, AlexNet, VGG-16, and Inception-v4	Auroral image classification, into 4 labels
[76] (2019)	Cycle-consistent Adversarial Network (CAN)	Classify auroral images by extracting Key Local Structures (KLS)
[77] (2018)	Pre-trained Deep Neural Network for feature extraction; Ridge classifier for classification	Nightside Auroral image classification into 6 labels
[73] (2020)	Variants of Deep Neural Network Architectures: VGG, AlexNet, and ResNet	Nightside Auroral image classification into 7 labels
[9] (2020)	Hybrid model: Uses Wavelet Transforms (WT) and Short-Time Fourier Transform (STFT) for feature extraction, along with Convolutional Neural Network (CNN)	Detect GIC
[78] (2015)	Elman Neural Network	Predict the 30 min maximum perturbation of the horizontal magnetic component
[79] (2020)	Two separate models: feed-forward NN and Long-Short Term Memory (LSTM) Neural Network	Predict the East and North component of the geomagnetic field, which, in turn, was used to derive the change in horizontal component

Despite the advancement in the use of ML for both auroral image classification and GIC prediction, there are still challenges that need to be addressed. For example, the classification of auroral images using ML is difficult due to the morphological nature of aurora, i.e., no two auroras are alike. In addition, there is no consensus within the scientific community about the number and type of classes the nighttime auroral images can be classified into [73]. In the case of GIC, the measurement of GIC is not readily available, and $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ from magnetometer readings are used as a proxy to calculate GIC. However, the $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ data tends to be noisy, which makes it difficult to predict using ML models [79]. Therefore, both auroral image classification and GIC prediction tend to suffer from data and model parameter uncertainty. The literature lacks the implementation of probabilistic ML models in auroral image classification and GIC prediction that can quantify data and model parameter uncertainty. To this end, in this section, two separate case studies are presented: (1) Bayesian CNN is used to classify auroral images into predetermined labels, and (2) BNN is used to predict the rate of change in Earth's horizontal magnetic field component, $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$. The Bayesian nature of the models allows the quantification of uncertainty. Although a more sophisticated architecture for the NNs, along with the Bayesian inference, could lead to greater accuracy, this study aims to present a motivation for uncertainty quantification in the domain of space weather forecasting. These case studies form a base upon which future research can be pursued.

5.2. Data Acquisition, Model Setup, and Variables

The dataset used to classify the auroral images, were same as that of Reference [73] (2020). The dataset consists of 3846 images of the aurora, obtained from 2010 through 2019 from the all-sky camera near Kiruna in Sweden, which the Swedish Institute operates for Space Physics. Each image is in RGB format with a resolution of 128 $\times$ 128. The splitting of the dataset into training and test set was done as per the instructions in Reference [73] (2020). The training dataset consists of 3000 images, and the testing dataset consists of the remaining 846 images. This classified the auroral images into the same labels used by Reference [73] (2020). As mentioned in Section 5.2, the seven labels are: arc aurora, auroral breakup, colored aurora, discrete irregular, patchy aurora, edge aurora, and faint clear (refer to Reference [73] (2020) for the definitions of each label).

When an auroral image is given as input to the implemented model, it will classify the image to label y, where $y \epsilon \{0, 1, 2, 3, 4, 5, 6\}$. Two models were implemented for the auroral image classification: a traditional CNN and a Bayesian CNN. Both CNNs consisted of five hidden layers. As described in Section 4.1, the traditional CNN uses a softmax classifier, and the Bayesian CNN uses Markov Chain Monte Carlo (MCMC).

In case of the GIC prediction case study, similar to Reference [79] (2020), the rate of change in local horizontal magnetic component ($\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$) is used as a proxy measure for GIC. The dataset used in this paper consisted of ground magnetometer data from Ottawa (OTT), obtained from SuperMAG [80]. SuperMAG is an association of organizations and national agencies from around the world that currently operate more than 300 ground-based magnetometers. The dataset also consisted of solar wind data from OmniWeb, NASA's Space Physics Data Facility. For this study, the data from the years 2001–2018 was taken into consideration. The input data was the vector $\vec{x}$, consisting of $\mathbf{B}_N$, $\mathbf{B}_E$, and $\mathbf{B}_Z$ components of the solar wind magnetic field; and flow pressure, proton density, speed, and solar wind temperature, obtained from the OmniWeb dataset. The output variable, y, is the rate of change of local horizontal magnetic component ($\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$). The local horizontal magnetic component, $\mathbf{B}_x$, was calculated from the dataset, as shown in Equation (26). The rate of change of local horizontal magnetic component, $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$, is the difference between consecutive $\mathbf{B}_x$ over a time frame of 1 min. The training dataset consisted of the years 2001–2010, and the test dataset consisted of the remaining years. Given our dataset, the analysis focuses on dayside GIC. A BNN model was implemented to predict $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$, where it makes the prediction with 95% confidence interval. The model consisted of five hidden layers, and it performs Variational Inference (VI).

$$\mathbf{B}_x = \sqrt{\mathbf{B}_N^2 + \mathbf{B}_E^2}. \tag{26}$$

5.3. Results and Discussion

In Bayesian DL, the model parameter and output are derived as a posterior probability distribution instead of a point estimate. Given a probability distribution, theories in statistics dictate that a confidence interval can be placed. This confidence interval ensures the reliability of the model prediction. The Bayesian CNN and BNN implemented in the two case studies exhibits the point mentioned above. Figure 3a,b represent the posterior probability distribution for an instance of a weight parameter of the Bayesian CNN and BNN, respectively. Both were obtained during a training iteration of their respective models. The mean weight for Bayesian CNN and BNN is 0.015 and -0.057, respectively. In both cases, the standard deviation is around ± 2. The standard deviation is a quantification of the level of model parameter uncertainty (epistemic uncertainty) that the models incur during that particular iteration of training. The 94% Highest Density Interval (HDI) represents the credible interval for the weight parameter. A credible interval can be thought of as the Bayesian counterpart of frequentist probability theory's confidence intervals. An interpretation of this credible interval is that 94% of the area under the distribution will lie within these credible bounds. Therefore, a smaller credible interval is preferred to a larger one, as it will ensure lower variance or dispersion of the distribution.

When the Bayesian CNN receives an observation from the test set, it calculates the posterior probability of that particular observation being part of a specific label. It reports the mean probability of a particular label within the 98% probability. Figure 4 shows the result of an instance when the test image belonged to label-4. Figure 5a illustrates an instance of the aurora image label-6 (faint clear), and Figure 5b represents a noisy image. After both the standard and Bayesian CNN model is trained, when the image in Figure 5a is the input, both the aforementioned implemented CNNs could classify it to the correct label. However, when the noisy image in Figure 5b is the input, the Bayesian CNN model generates the mean probability of the input image belonging to a particular label, and the probabilities tend to be very low. In contrast, when the noisy is the input to the standard CNN, it classifies the noisy image as one of the labels. This exercise exhibits the unreliability of a Non-Bayesian DL approach due to its inability to make probabilistic inferences about uncertainty. Within the context of a classification problem, the mean classification probability represents the level of confidence in the model results. The Bayesian CNN model has a classification accuracy of 83.04%, where the classification accuracy is calculated as shown in Equation (27).

$$ClassificationAccuracy = \frac{TruePositive + TrueNegative}{TotalSample}. \tag{27}$$

For the case of predicting $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$, using the BNN, a predictive posterior distribution is obtained, which captures the model uncertainty, as well as a confidence interval of 95%, and is placed on the prediction to attain the desired reliability. In Figure 6, the dotted red lines represents the confidence interval of 95%, the dark blue line is the predicted mean $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ at each time step, and the cyan lines represents variance associated with each predicted mean $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$. The variance represents the level of uncertainty in the prediction and is an accumulation of the epistemic uncertainty and the aleatoric uncertainty. It can be observed that the uncertainty is higher for particular time steps relative to others. Uncertainty is higher in those regions where observations similar to the test sample were scarce in the training dataset. The accuracy of the BNN model was measured using RMSE, which has a value of 81.12%. The RMSE was calculated by comparing the predicted mean $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ value against the ground truth.

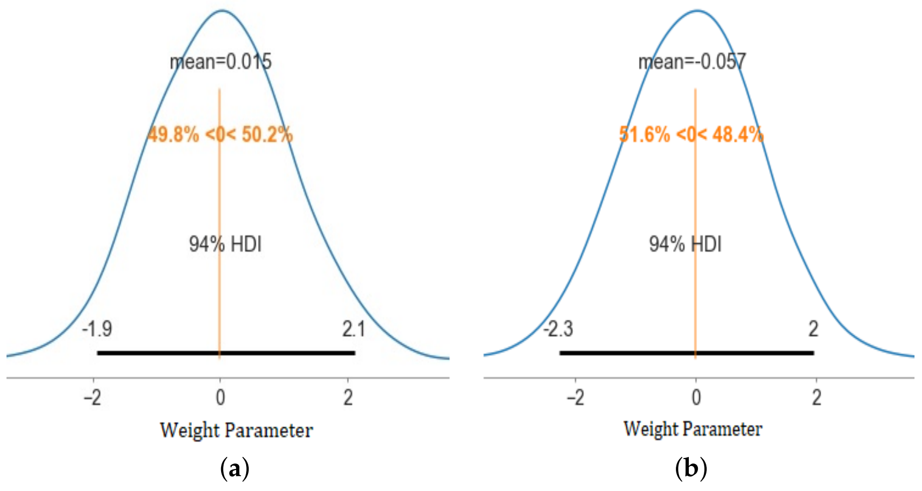

Figure 3. (**a**,**b**) An instance of the posterior probability distribution of the weight parameter for the Bayesian CNN and BNN, respectively. Both were obtained during an iteration of the models' training.

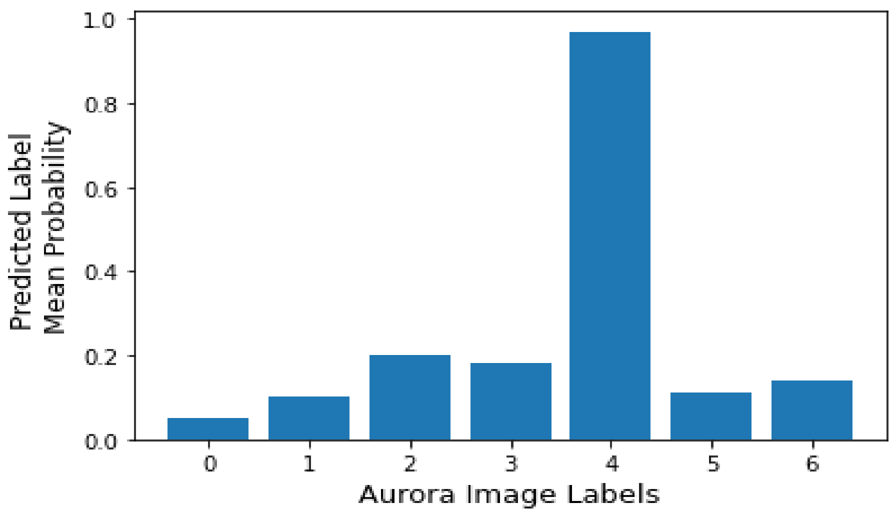

Figure 4. An instance of predicted label mean probability with test observation belonging to label-4. Each bar corresponds to the mean probability of a test input image belonging to a particular label.

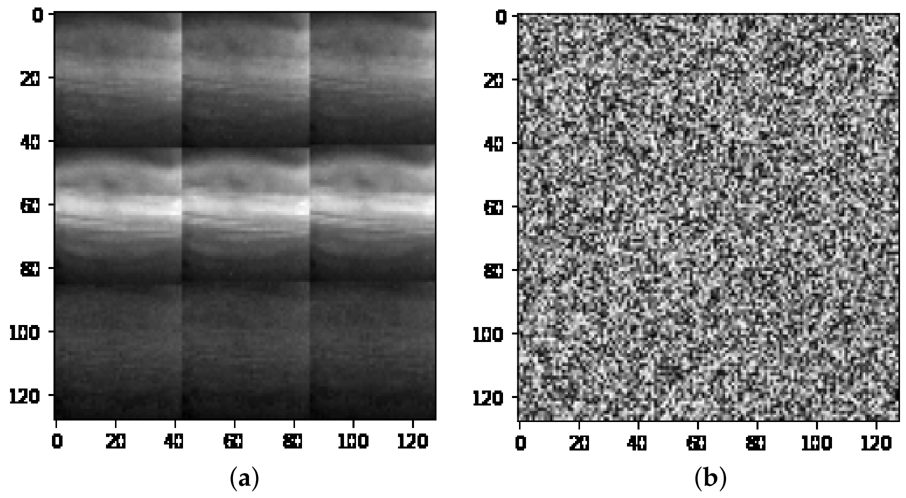

Figure 5. (**a**) The aurora label-6 (faint clear). (**b**) A noisy image.

It is to be noted that the confidence level which is achieved through uncertainty quantification does not necessarily increase the model accuracy, but, rather, it provides a quantification of the level of reliability that can be placed on the model's existing output and accuracy. Given the "off the shelf" nature of the implemented models, the accuracy of the models could be improved using more sophisticated DL architectures, e.g., Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) Networks, etc. A popular ML methodology that has gained traction in recent years is Ensemble Learning (EL) [81,82]. Ensemble learning involves multiple machine learning models in the prediction process. The member models forming the ensemble are trained on subsets of the total training dataset. The predictions from the member models are then synthesized to form a final prediction by taking a form of average or weighted sum of the individual predictions. This form of learning is aimed to increase model accuracy.

The case studies presented in this paper deal with ground magnetic component, solar wind, and auroral image data, which are a means of studying space weather phenomena, such as geomagnetic storms. In recent years, data on relativistic (galactic and solar) cosmic rays have been used in space weather storms forecasting, specification of magnetic properties of coronal mass ejections (CMEs), and ground level enhancements (GLEs) [83]. Data on such high-energy cosmic rays are recorded using a network of ground-based instruments called global neutron monitors [84]. Neutron monitors detect secondary cosmic rays created by interactions of predominantly protons and heavier nuclei with the Earth's atmosphere [84]. In past literature, ML has been used to study cosmic rays. Ref. [85] (2018) used Convolutional Neural Networks to reconstruct cosmic ray properties using data from simulated cosmic ray induced air showers. Ref. [86] (2020) trained two separate Convolutional Neural Network models to study the propagation of cosmic rays. The first model learns how to infer the propagation and source parameters from the energy spectra of cosmic rays. The second model differs from the first in that it is capable of learning from the data with added artificial fluctuations. Bayesian analysis, along with ML, has been used to study propagation and injection parameters for cosmic ray. Ref. [87] (2016) carried out a Bayesian search augmented with a DL variant to determine posterior distribution of propagation parameters. Although ML has been used to study such phenomenons, there is still room for development and research into uncertainty quantification and model reliability.

The field of uncertainty quantification (UQ) has become a central topic in engineering research and applications over recent decades [88]. However, within the realm of space physics, there is still a need for exploring UQ and its applications [89]. For example, physics-based models have become the core of space weather forecasting [90,91]; therefore, a probabilistic interpretation of simulation outputs will be more informative and reliable than a traditional point estimate [92]. In turn, such a probabilistic interpretation requires at least two ingredients: a correct specification of the uncertainties attributed to input parameters and the study of the propagation of such uncertainties through the physics-based model. The case studies presented in this survey paper touches on the first point by showing how a probabilistic framework, such as Bayesian statistics and NNs, can be implemented to address data and parameter uncertainties. Regarding the second point, the paper surveys how ODEs and PDEs have been used in past literature to model physics-based natural systems and the importance of incorporating a probabilistic framework to address parameter uncertainty.

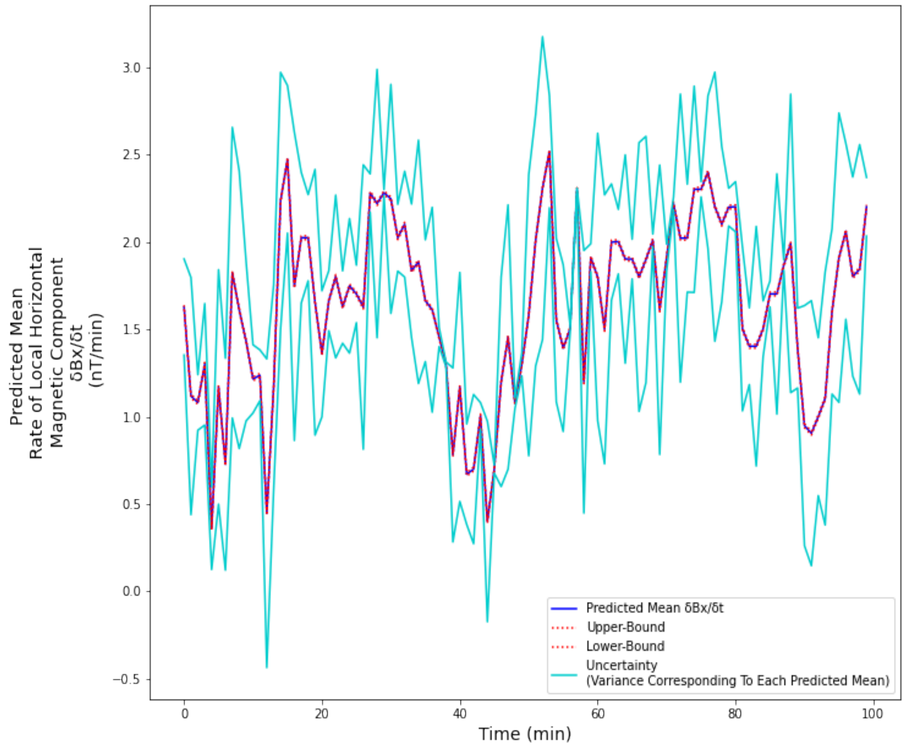

Figure 6. Predicted mean $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ values for a set of test observations (from Ottawa, Canada), with 95% confidence interval boundary and corresponding uncertainty (variance) for each predicted mean.

A general classification divides UQ techniques between those that are intrusive and those that are non-intrusive [30]. The former term shows how underlying PDEs are recast in stochastic equations and can be solved using Gaussian Process models while considering uncertainty. The latter shows how DL methods and NNs can be used as a black-box simulator that can run multiple times, and, by leveraging Bayesian statistics, data and parameter uncertainty can be taken into account.

As surveyed in this paper, both GPR models and Bayesian DL have been used with great success across various fields to address the issue of uncertainty. Both models have their roots in Bayesian inference and can quantify the uncertainty and the confidence level in their generated results. The advantage of GP models lies in their ability to incorporate physics-based PDEs and SPDEs, which can model natural systems. Proven frameworks have been developed in past literature to solve problems focused on different disciplines. Another advantage lies in their non-parametric nature. In contrast to parametric models, GPs are not distilling the training data into a finite set of parameters. Therefore, the results generated by GP models are not limited by the number of model parameters, as they take into account the whole training data during each inference. However, the latter advantage comes at the cost of scalability, i.e., the computational cost of running such models increases with high-dimensional datasets. A related drawback of GP models is that different feature selection methods need to be applied to determine those variables deemed to be most relevant for the model to make accurate predictions. A common approach for feature selection in GPs is to use an Automatic Relevance Determination kernel, and then feature selection can be achieved by tuning the kernel parameters to maximize the marginal likelihood. However, the pitfall of using such a step is, it might lead to the over-fitting of the marginal likelihood. This results in a model that performs poorly compared to a model that uses a simple radial basis covariance function. In addition to this, feature selection methods tend to perform better with linear models. In contrast, DL methods can deal with high-dimensional data without the intervention of external feature selection methods ahead of time. DL methods can learn high-level features from the data incrementally and are optimally tuned for the desired outcome. The same NN approach can be applied to different data types and, therefore, can be adopted for solving a wider array of problems. DL methods in their traditional form do not naturally adopt physics-informed PDEs or SPDEs to generate solutions. As discussed in Section 4, literature consisting of studies have emerged, where physics-informed NN and DL frameworks have been developed [93–95]. Several physics constraints need to be factored in with the model when modeling space weather phenomena. For example, the propagation of solar wind energy from the dayside magnetosphere to the nightside magnetosphere and to the upper atmosphere needs to be better understood when it comes to analyzing the relation between solar wind and GICs because there are complex internal processes in the near-Earth space environment that deposit and distribute the solar wind energy with different spatial and temporal scales. For example, the propagation of solar wind energy from the dayside magnetosphere to the nightside magnetosphere and to the upper atmosphere needs to be better understood when it comes to analyzing the relation between solar wind and GICs because there are complex internal processes in the near-Earth space environment that deposit and distribute the solar wind energy with different spatial and temporal scales. ML models in their standard form will not incorporate this information, but it will be incorporated if the model is physics-informed. However, the existing physics-informed models do not apply any probabilistic methods, such as Bayesian inference, and cannot quantify data and parameter uncertainty. Given the scope of auroral image classification and GIC prediction, this paper highlights the need to explore and develop models that leverage physics-based PDEs/SPDEs and Bayesian inference in an NN or DL setting. Such a model will generate predictions that are constrained by the laws of physics and provide a confidence level on the predictions based on its ability to quantify data and parameter uncertainty.

6. Conclusions

This paper surveys the different approaches used in ML to quantify uncertainty. There are different dimensions of uncertainty, and its propagation across a model can be thought of as either a forward propagation problem or an inverse propagation problem. The paper elaborates on GPR and BNN, which have been predominantly used in past literature, for UQ in traditional ML and DL, respectively. Both the approaches have their roots in Bayesian inference, and both can be used to solve physics-informed PDEs and SPDEs. The advantages and disadvantages of GPR versus BNN models have also been weighed. Finally, the paper presents two case studies for UQ by implementing Bayesian DL focusing on auroral image classification and GIC prediction. The parameters of both the models are estimated as a posterior probability distribution, where each distribution have a standard deviation of ± 2, which is equal to the models' parameter uncertainty. The Bayesian CNN model which was used for auroral image classification has an accuracy of 83.04%, and the classification probability acts as the model's level of confidence. The mean $\frac{\delta \mathbf{B}_x}{\delta \mathbf{t}}$ at each time step was predicted with 95% confidence interval, along with the corresponding variance. The regression model has an accuracy of 81.12%. The paper highlights that UQ and confidence interval on parameters and output posterior distribution does not necessarily increase model accuracy but, rather, quantifies a level of reliability on the model's existing accuracy. The primary aim of the two case studies is to motivate research into the development of reliable ML models through UQ, within the domain of space weather forecasting.

Author Contributions: M.S.M. and T.S. was the primary authors of the paper. M.S.M. provided guidance on model development and data analysis also participated in interpretation. T.S. conducted data preparation, model development, analysis, and assisted with writing. A.M.K. provided guidance on data preparation and model development, assisted with data preparation. C.M.N. and H.C. assisted with overall project discussions and provided editorial comments. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by NSF EPSCoR Award OIA-1920965.

Data Availability Statement: The solar wind, IMF, and Sym-H index data are available from OMNIWeb at https://omniweb.gsfc.nasa.gov, accessed on 1 September 2020, Replication Data for: Auroral Image Classification with Deep Neural Networks available at https://dataverse.no/, accessed on 1 September 2020, and the ground magnetometer data are available from SuperMAG at http://supermag.jhuapl.edu, accessed on 1 September 2020.

Acknowledgments: We thank members of the the "Machine-learning Algorithms for Geomagnetically Induced Currents In Alaska and New Hampshire" (MAGICIAN) team. We thank OMNIWeb and SuperMAG for providing the data.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. McAfee, A.; Brynjolfsson, E. Big Data: The Management Revolution. *Harv. Bus. Rev.* **2012**, *90*, 60–66,68,128.
2. Hariri, R.H.; Fredericks, E.M.; Bowers, K.M. Uncertainty in big data analytics: Survey, opportunities, and challenges. *J. Big Data* **2019**, *6*, 44. [CrossRef]
3. Chen, M.; Mao, S.; Liu, Y. Big data: A survey. *Mob. Netw. Appl.* **2014**, *19*, 171–209. [CrossRef]
4. Liakos, K.G.; Busato, P.; Moshou, D.; Pearson, S.; Bochtis, D. Machine learning in agriculture: A review. *Sensors* **2018**, *18*, 2674. [CrossRef]
5. Siddique, T. Agrobiodiversity for Pest Management: An Integrated Bioeconomic Simulation and Machine Learning Approach. 2019. Available online: https://www.semanticscholar.org/paper/Agrobiodiversity-For-Pest-Management3A-An-Integrated-Siddique/1c2075401bb28b826c9ce12969d46ae4b4fed13e (accessed on 20 November 2021).
6. Camporeale, E. The Challenge of Machine Learning in Space Weather: Nowcasting and Forecasting. *Space Weather* **2019**, *17*, 1166–1207. [CrossRef]
7. Antuña-Sánchez, J.C.; Román, R.; Cachorro, V.E.; Toledano, C.; López, C.; González, R.; Mateos, D.; Calle, A.; de Frutos, Á.M. Relative sky radiance from multi-exposure all-sky camera images. *Atmos. Meas. Tech.* **2021**, *14*, 2201–2217. [CrossRef]
8. Ayyub, B.M.; Klir, G.J. *Uncertainty Modeling and Analysis in Engineering and the Sciences*; Chapman and Hall/CRC: London, UK, 2006; pp. 1–378. [CrossRef]
9. Wang, S.; Dehghanian, P.; Li, L.; Wang, B. A Machine Learning Approach to Detection of Geomagnetically Induced Currents in Power Grids. *IEEE Trans. Ind. Appl.* **2020**, *56*, 1098–1106. [CrossRef]

10. Walker, W.E.; Harremoes, P.; Rotmans, J.; Van Der Sluijs, J.P.; Van Asselt, M.B.A.; Janssen, P.; Krayer Von Krauss, M.P. Defining Uncertainty. *Integr. Assess.* **2003**, *4*, 5–17. [CrossRef]
11. Lele, S.R. How Should We Quantify Uncertainty in Statistical Inference? *Front. Ecol. Evol.* **2020**, *8*. [CrossRef]
12. Abdar, M.; Pourpanah, F.; Hussain, S.; Rezazadegan, D.; Liu, L.; Ghavamzadeh, M.; Fieguth, P.; Cao, X.; Khosravi, A.; Acharya, U.R.; et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *arXiv* **2020**, arXiv:2011.06225.
13. Siddique, T.; Mahmud, M.S. Classification of fNIRS Data Under Uncertainty: A Bayesian Neural Network Approach 2021. Available online: https://ieeexplore.ieee.org/document/9398971 (accessed on 20 November 2021)
14. Klir, G.J. *Uncertainty and Information: Foundations of Generalized Information Theory*; Wiley: Hoboken, NJ, USA, 2006; Volume 35, pp. 1297–1299. [CrossRef]
15. Ravetz, J.; Funtowicz, S. *Uncertainty and Quality in Knowledge for Policy*; Springer: Berlin, Germany 1991; p. 229.
16. Van Asselt, M.B.; Rotmans, J. Uncertainty in Integrated Assessment modelling. From positivism to pluralism. *Clim. Chang.* **2002**, *54*, 75–105. [CrossRef]
17. Sluijs, J.V.D. *Anchoring Amid Uncertainty on the Management of Uncertainties in Risk Assessment of Anthropogenic Climate Change*; Ludy Feyen: Utrecht, The Netherlands, 1997.
18. Meyer, V.R. Measurement uncertainty. *J. Chromatogr. A* **2007**, *1158*, 15–24. [CrossRef]
19. International Bureau of Weights and Measures; International Organization for Standardization. (Eds.) *Guide to the Expression of Uncertainty in Measurement*, 1st ed.; International Organization for Standardization: Geneve, Switzerland, 1993.
20. Volodina, V.; Challenor, P. The importance of uncertainty quantification in model reproducibility. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2021**, *379*, 20200071. [CrossRef]
21. Bai, Y.; Jin, W.L. Chapter 33—Random Variables and Uncertainty Analysis. In *Marine Structural Design*, 2nd ed.; Bai, Y., Jin, W.L., Eds.; Butterworth-Heinemann: Oxford, UK, 2016; pp. 615–625. [CrossRef]
22. Mandat, D.; Pech, M.; Hrabovsky, M.; Schovanek, P. A TMO HEAD W ORKSHOP, 2013 All Sky Camera Instrument for Night Sky Monitoring. 2014. Available online: https://arxiv.org/abs/1402.4762 (accessed on 20 November 2021)
23. Frigg, R.; Bradley, S.; Du, H.; Smith, L.A. Laplace's Demon and the Adventures of His Apprentices. *Philos. Sci.* **2014**, *81*, 31–59. [CrossRef]
24. Icke, I.; Bongard, J.C. Improving genetic programming based symbolic regression using deterministic machine learning. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, 20–23 June 2013; pp. 1763–1770. [CrossRef]
25. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. *Int. Conf. Mach. Learn.* **2014**, *1*, 605–619.
26. Zhang, J.; Yin, J.; Wang, R. Basic framework and main methods of uncertainty quantification. *Math. Probl. Eng.* **2020**, *2020*, 6068203. [CrossRef]
27. Sullivan, T. *Introduction to Uncertainty Quantification*; Texts in Applied Mathematics; Springer International Publishing: Cham, Switzerland, 2015; Volume 63. [CrossRef]
28. Iskandarani, M.; Wang, S.; Srinivasan, A.; Carlisle Thacker, W.; Winokur, J.; Knio, O.M. An overview of uncertainty quantification techniques with application to oceanic and oil-spill simulations. *J. Geophys. Res. Ocean* **2016**, *121*, 2789–2808. [CrossRef]
29. Peckham, S.D.; Kelbert, A.; Hill, M.C.; Hutton, E.W. Towards uncertainty quantification and parameter estimation for Earth system models in a component-based modeling framework. *Comput. Geosci.* **2016**, *90*, 152–161. [CrossRef]
30. Camporeale, E.; Shprits, Y.; Chandorkar, M.; Drozdov, A.; Wing, S. On the propagation of uncertainties in radiation belt simulations. *Space Weather* **2016**, *14*, 982–992. [CrossRef]
31. Hibbert, D.B. The uncertainty of a result from a linear calibration. *Analyst* **2006**, *131*, 1273–1278. [CrossRef]
32. Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260. [CrossRef] [PubMed]
33. Cohen, S. Chapter 2 - The basics of machine learning: Strategies and techniques. In *Artificial Intelligence and Deep Learning in Pathology*; Cohen, S., Ed.; Elsevier: Amsterdam, The Netherlands, 2021; pp. 13–40. [CrossRef]
34. Schmidt, J.; Marques, M.R.; Botti, S.; Marques, M.A. Recent advances and applications of machine learning in solid-state materials science. *NPJ Comput. Mater.* **2019**, *5*, 83. [CrossRef]
35. Senel, O. Infill Location Determination and Assessment of Corresponding Uncertainty. Ph.D. Thesis, Texas A & M University, College Station, TX, USA, 2009.
36. Song, T.; Ding, W.; Liu, H.; Wu, J.; Zhou, H.; Chu, J. Uncertainty quantification in machine learning modeling for multi-step time series forecasting: Example of recurrent neural networks in discharge simulations. *Water* **2020**, *12*, 912. [CrossRef]
37. Jha, A.; Chandrasekaran, A.; Kim, C.; Ramprasad, R. Impact of dataset uncertainties on machine learning model predictions: The example of polymer glass transition temperatures. *Model. Simul. Mater. Sci. Eng.* **2019**, *27*, 024002. [CrossRef]
38. Kim, H.Y.; Won, C.H. Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Syst. Appl.* **2018**, *103*, 25–37. [CrossRef]
39. Benoudjit, N.; Verleysen, M. On the kernel widths in radial-basis function networks. *Neural Process. Lett.* **2003**, *18*, 139–154. [CrossRef]

40. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*; The MIT Press: Cambridge, MA, USA, 2005.
41. Särkkä, S. Linear operators and stochastic partial differential equations in Gaussian process regression. In *International Conference on Artificial Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 151–158. [CrossRef]
42. Griffiths, D.J. *Introduction to Electrodynamics*, 4th ed.; Pearson: Boston, MA, USA, 2013.
43. Holden, H. (Ed.) *Stochastic Partial Differential Equations: A Modeling, White Noise Functional Approach*, 2nd ed.; Universitext: London, UK; Springer: New York, NY, USA, 2010.
44. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Numerical Gaussian processes for time-dependent and nonlinear partial differential equations. *SIAM J. Sci. Comput.* **2018**, *40*, A172–A198. [CrossRef]
45. Hennig, P.; Osborne, M.A.; Girolami, M. Probabilistic numerics and uncertainty in computations. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2015**, *471*, 20150142. [CrossRef]
46. Cockayne, J.; Oates, C.; Sullivan, T.; Girolami, M. Probabilistic Meshless Methods for Partial Differential Equations and Bayesian Inverse Problems. *arXiv* **2016**, arXiv:abs/1605.07811.
47. Dondelinger, F.; Filippone, M.; Rogers, S.; Husmeier, D. ODE parameter inference using adaptive gradient matching with Gaussian processes. *J. Mach. Learn. Res.* **2013**, *31*, 216–228.
48. Ashyraliyev, M.; Fomekong-Nanfack, Y.; Kaandorp, J.A.; Blom, J.G. Systems biology: Parameter estimation for biochemical models: Parameter estimation in systems biology. *FEBS J.* **2009**, *276*, 886–902. [CrossRef] [PubMed]
49. Girolami, M.; Calderhead, B. Riemann manifold Langevin and Hamiltonian Monte Carlo methods: Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2011**, *73*, 123–214. [CrossRef]
50. Conrad, P.R.; Girolami, M.; Särkkä, S.; Stuart, A.; Zygalakis, K. Statistical analysis of differential equations: Introducing probability measures on numerical solutions. *Stat. Comput.* **2017**, *27*, 1065–1082. [CrossRef]
51. Sevakula, R.K.; Au-Yeung, W.M.; Singh, J.P.; Heist, E.K.; Isselbacher, E.M.; Armoundas, A.A. State-of-the-Art Machine Learning Techniques Aiming to Improve Patient Outcomes Pertaining to the Cardiovascular System. *J. Am. Heart Assoc.* **2020**, *9*, e013924. [CrossRef]
52. Pantoja, M.; Behrouzi, A.; Fabris, D. An introduction to deep learning. In Proceedings of the 2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA), Montreal, QC, Canada, 2–5 July 2018.
53. Guarascio, M.; Manco, G.; Ritacco, E. Deep Learning. In *Encyclopedia of Bioinformatics and Computational Biology*; Ranganathan, S., Gribskov, M., Nakai, K., Schönbach, C., Eds.; Academic Press: Oxford, UK, 2019; pp. 634–647. [CrossRef]
54. Gamboa, J.C.B. Deep Learning for Time-Series Analysis. *arXiv* **2017**, arXiv:1701.01887.
55. McDermott, P.L.; Wikle, C.K. Bayesian Recurrent Neural Network Models for Forecasting and Quantifying Uncertainty in Spatial-Temporal Data. *Entropy* **2019**, *21*, 184. [CrossRef] [PubMed]
56. O'Shea, K.; Nash, R. An Introduction to Convolutional Neural Networks. *arXiv* **2015**, arXiv:1511.08458.
57. Sharma, V. Deep Learning Algorithms. Available online: https://www.datarobot.com/wiki/deep-learning/ (accessed on 25 December 2021)
58. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. 2017. Available online: http://xxx.lanl.gov/abs/1711.10566 (accessed on 20 November 2021)
59. Ghahramani, Z. Probabilistic machine learning and artificial intelligence. *Nature* **2015**, *521*, 452–459. [CrossRef]
60. Salimans, T.; Kingma, D.; Welling, M. Markov chain monte carlo and variational inference: Bridging the gap. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 1218–1226.
61. Andrieu, C.; De Freitas, N.; Doucet, A.; Jordan, M.I. An introduction to MCMC for machine learning. *Mach. Learn.* **2003**, *50*, 5–43. [CrossRef]
62. Liang, F.; Liu, C.; Carroll, R. *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples*; John Wiley & Sons: Hoboken, NJ, USA, 2011; Volume 714.
63. Camporeale, E.; Johnson, J.R.; Wing, S. (Eds.) *Machine Learning Techniques for Space Weather*; Elsevier: Amsterdam, The Netherlands; Cambridge, MA, USA, 2018.
64. Piersanti, M.; Di Matteo, S.; Carter, B.A.; Currie, J.; D'Angelo, G. Geoelectric Field Evaluation during the September 2017 Geomagnetic Storm: MA.I.GIC. Model. *Space Weather* **2019**, *17*, 1241–1256. [CrossRef]
65. Salman, T.M.; Lugaz, N.; Farrugia, C.J.; Winslow, R.M.; Galvin, A.B.; Schwadron, N.A. Forecasting Periods of Strong Southward Magnetic Field Following Interplanetary Shocks. *Space Weather* **2018**, *16*, 2004–2021. [CrossRef]
66. Tsurutani, B.T.; Lakhina, G.S.; Hajra, R. The physics of space weather/solar-terrestrial physics (STP): What we know now and what the current and future challenges are. *Nonlinear Process. Geophys.* **2020**, *27*, 75–119. [CrossRef]
67. Watari, S.; Nakamura, S.; Ebihara, Y. Measurement of geomagnetically induced current (GIC) around Tokyo, Japan. *Earth Planets Space* **2021**, *73*, 102. [CrossRef]
68. de Villiers, J.S.; Kosch, M.; Yamazaki, Y.; Lotz, S. Influences of various magnetospheric and ionospheric current systems on geomagnetically induced currents around the world. *Space Weather* **2017**, *15*, 403–417. [CrossRef]
69. Salman, T.M.; Lugaz, N.; Farrugia, C.J.; Winslow, R.M.; Jian, L.K.; Galvin, A.B. Properties of the Sheath Regions of Coronal Mass Ejections with or without Shocks from STEREO in situ Observations near 1 au. *Astrophys. J.* **2020**, *904*, 177. [CrossRef]
70. Rajput, V.N.; Boteler, D.H.; Rana, N.; Saiyed, M.; Anjana, S.; Shah, M. Insight into impact of geomagnetically induced currents on power systems: Overview, challenges and mitigation. *Electr. Power Syst. Res.* **2021**, *192*, 106927. [CrossRef]

71. Rao, J.; PhD, N.P.; Amariutei, O.; Syrjäsuo, M.; Van De Sande, K.E. Automatic auroral detection in color all-sky camera images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 4717–4725. [CrossRef]
72. Yang, Q.; Liang, J.; Hu, Z.; Zhao, H. Auroral sequence representation and classification using hidden markov models. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 5049–5060. [CrossRef]
73. Kvammen, A.; Wickstrøm, K.; McKay, D.; Partamies, N. Auroral Image Classification with Deep Neural Networks. *J. Geophys. Res. Space Phys.* **2020**, *125*, e2020JA027808. [CrossRef]
74. Yang, Q.; Zhou, P. Representation and Classification of Auroral Images Based on Convolutional Neural Networks. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 523–534. [CrossRef]
75. Gao, X.; Fu, R.; Li, X.; Tao, D.; Zhang, B.; Yang, H. Aurora image segmentation by combining patch and texture thresholding. *Comput. Vis. Image Underst.* **2011**, *115*, 390–402. [CrossRef]
76. Yang, Q.; Tao, D.; Han, D.; Liang, J. Extracting Auroral Key Local Structures From All-Sky Auroral Images by Artificial Intelligence Technique. *J. Geophys. Res. Space Phys.* **2019**, *124*, 3512–3521. [CrossRef]
77. Clausen, L.B.; Nickisch, H. Automatic Classification of Auroral Images From the Oslo Auroral THEMIS (OATH) Data Set Using Machine Learning. *J. Geophys. Res. Space Phys.* **2018**, *123*, 5640–5647. [CrossRef]
78. Wintoft, P.; Wik, M.; Viljanen, A. Solar wind driven empirical forecast models of the time derivative of the ground magnetic field. *J. Space Weather Space Clim.* **2015**, *5*, A7. [CrossRef]
79. Keesee, A.M.; Pinto, V.; Coughlan, M.; Lennox, C.; Mahmud, M.S.; Connor, H.K. Comparison of Deep Learning Techniques to Model Connections Between Solar Wind and Ground Magnetic Perturbations. *Front. Astron. Space Sci.* **2020**, *7*, 1–8. [CrossRef]
80. Gjerloev, J.W. The SuperMAG data processing technique: TECHNIQUE. *J. Geophys. Res. Space Phys.* **2012**, *117*. [CrossRef]
81. Rokach, L. *Ensemble Learning: Pattern Classification Using Ensemble Methods*; World Scientific: Singapore, 2019.
82. Tang, Y.; Wang, Y.; Cooper, K.M.; Li, L. Towards Big Data Bayesian Network Learning—An Ensemble Learning Based Approach. In Proceedings of the 2014 IEEE International Congress on Big Data, Anchorage, AK, USA, 27 June 2014; pp. 355–357. [CrossRef]
83. Mavromichalaki, H.; Souvatzoglou, G.; Sarlanis, C.; Mariatos, G.; Plainaki, C.; Gerontidou, M.; Belov, A.; Eroshenko, E.; Yanke, V. Space weather prediction by cosmic rays. *Adv. Space Res.* **2006**, *37*, 1141–1147. [CrossRef]
84. Kuwabara, T.; Bieber, J.W.; Clem, J.; Evenson, P.; Pyle, R.; Munakata, K.; Yasue, S.; Kato, C.; Akahane, S.; Koyama, M.; et al. Real-time cosmic ray monitoring system for space weather. *Space Weather* **2006**, *4*. [CrossRef]
85. Erdmann, M.; Glombitza, J.; Walz, D. A deep learning-based reconstruction of cosmic ray-induced air showers. *Astropart. Phys.* **2018**, *97*, 46–53. [CrossRef]
86. Tsai, Y.L.S.; Chung, Y.; Yuan, Q.; Cheung, K. Inverting cosmic ray propagation by Convolutional Neural Networks. *arXiv* **2020**, arXiv:2011.11930.
87. Jóhannesson, G.; de Austri, R.R.; Vincent, A.; Moskalenko, I.; Orlando, E.; Porter, T.; Strong, A.; Trotta, R.; Feroz, F.; Graff, P.; et al. Bayesian analysis of cosmic ray propagation: Evidence against homogeneous diffusion. *Astrophys. J.* **2016**, *824*, 16. [CrossRef]
88. Smith, R.C. *Uncertainty Quantification: Theory, Implementation, and Applications*; Siam: Bangkok, Thailand, 2013.
89. Knipp, D.J. Advances in Space Weather Ensemble Forecasting: SPACE WEATHER ENSEMBLE FORECASTING. *Space Weather* **2016**, *14*, 52–53. [CrossRef]
90. Tóth, G.; van der Holst, B.; Sokolov, I.V.; De Zeeuw, D.L.; Gombosi, T.I.; Fang, F.; Manchester, W.B.; Meng, X.; Najib, D.; Powell, K.G.; et al. Adaptive numerical algorithms in space weather modeling. *J. Comput. Phys.* **2012**, *231*, 870–903. [CrossRef]
91. Schunk, R.W.; Scherliess, L.; Eccles, V.; Gardner, L.C.; Sojka, J.J.; Zhu, L.; Pi, X.; Mannucci, A.J.; Butala, M.; Wilson, B.D.; et al. Space weather forecasting with a Multimodel Ensemble Prediction System (MEPS). *Radio Sci.* **2016**, *51*, 1157–1165. [CrossRef]
92. Morley, S.K.; Welling, D.T.; Woodroffe, J.R. Perturbed Input Ensemble Modeling With the Space Weather Modeling Framework. *Space Weather* **2018**, *16*, 1330–1347. [CrossRef]
93. Guo, Y.; Cao, X.; Liu, B.; Gao, M. Solving partial differential equations using deep learning and physical constraints. *Appl. Sci.* **2020**, *10*, 5917. [CrossRef]
94. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
95. Shin, Y.; Darbon, J.; Karniadakis, G.E. On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs. *Commun. Comput. Phys.* **2020**, *28*, 2042–2074. [CrossRef]