Journal of Physics D: Applied Physics

PAPER

Application of machine learning for the estimation of electron energy distribution from optical emission spectra

To cite this article: Kamran Shojaei and Lorenzo Mangolini 2021 *J. Phys. D: Appl. Phys.* **54** 265202

IOP Publishing Journal of Physics D: Applied Physics

J. Phys. D: Appl. Phys. **54** (2021) 265202 (8pp) https://doi.org/10.1088/1361-6463/abf61e

Application of machine learning for the estimation of electron energy distribution from optical emission spectra

Kamran Shojaei[1] **and Lorenzo Mangolini**[1,2,*]

[1] Department of Mechanical Engineering, University of California Riverside, Riverside, CA, United States of America
[2] Materials Science and Engineering Program, University of California Riverside, Riverside, CA, United States of America

E-mail: lmangolini@engr.ucr.edu

Received 22 February 2021, revised 24 March 2021
Accepted for publication 8 April 2021
Published 23 April 2021

Abstract

This paper discusses the use of probabilistic deep neural networks for the prediction of the electron energy probability function in low-temperature non-thermal plasmas. The neural networks are trained using optical emission spectroscopy and Langmuir probe measurements, with the goal of providing a reliable estimate of the electron energy probability function solely from optical emission data. The performance of both non-Bayesian and Bayesian networks is evaluated. It is found that Bayesian models are preferable as they assign a higher level of uncertainty to their prediction especially when the dataset used to train them is small. This work describes one of the many potential applications of machine learning in plasma science and technology.

Supplementary material for this article is available online

Keywords: machine learning, Langmuir probe, spectroscopy, non-thermal plasmas

(Some figures may appear in color only in the online journal)

1. Introduction

Machine learning (ML) refers to a broad set of computational tools that enable a machine to perform tasks by learning from data. ML has been used in a wide range of fields from skin cancer detection and medical image segmentation [1] to transportation in form of self-driving cars [2]. ML models demonstrate physician-level accuracy at diagnostic tasks such as identifying cardiovascular risk, diabetic retinopathy, moles from melanomas [3, 4], breast lesion detection in mammograms [5], and analysis of spine with magnetic resonance imaging [6]. Using radiological images, object detection and segmentation are employed for flagging large-artery occlusion in the brain [7, 8]. With respect to autonomous vehicles, ML models clone driving behavior and perform tasks such as detection [9, 10], segmentation [11], classification [12, 13] and tracking of objects on the road [14]. In plasma science, ML has been applied for reconstructing the plasma radiation profile from bolometer measurements in fusion reactors [15]. The promising applications of ML also include inference of plasma and surface properties from spectral data in non-equilibrium plasmas [16] and predicting in-flight particle characteristics of an atmospheric plasma spray process for coating applications [17]. Despite these recent demonstrations, ML is still very much an under-utilized tool in plasma science. This consideration has motivated us to investigate its use for the characterization of low-temperature plasmas, in particular

* Author to whom any correspondence should be addressed.

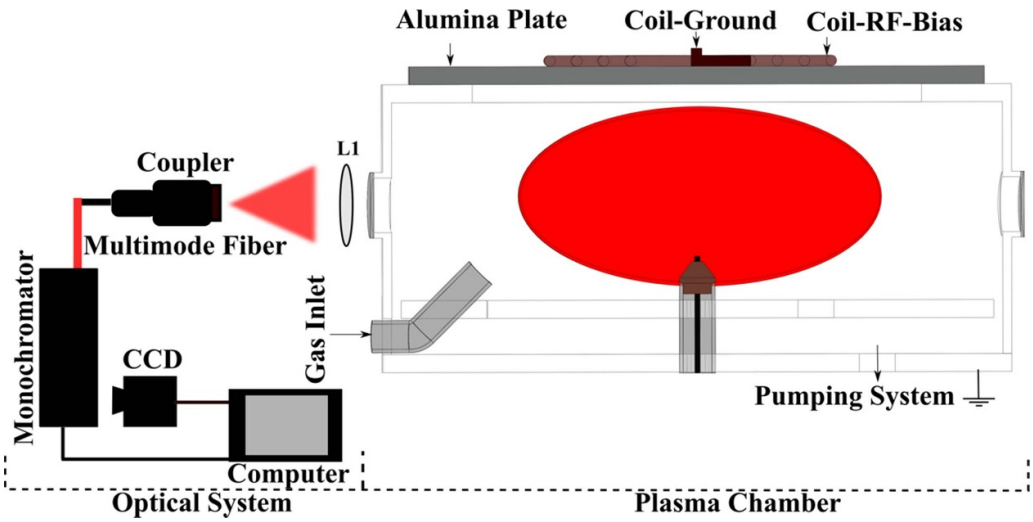

Figure 1. Schematic of the setup employed to perform the Langmuir probe and spectroscopy measurements showing the optical system and the plasma chamber.

for the estimation of the electron energy distribution function (EEDF) using easily obtainable optical emission spectra. In this work, we have performed both Langmuir probe and optical emission spectroscopy (OES) measurements to establish a training dataset, which we have then used to develop ML models that are capable of determining the electron energy distribution of a low-pressure non-thermal plasma starting simply from OES data.

Langmuir probe and OES are two common techniques used to diagnose nonthermal plasmas. A Langmuir probe consists, in its simplest form, of a wire that is inserted into the plasma. Analysis of the current–voltage (I–V) characteristic allows determining fundamental plasma properties such as electron temperature (T_e), electron energy distribution and probability function (f_d, f_p), and plasma density ($n_{e,i}$) [18–22]. Despite being powerful, this characterization technique is also non-trivial to set up and utilize. It would be desirable to use simpler and more accessible tools for the characterization of f_d. OES, on the other hand, is a useful tool for determining, for instance, the density of certain excited states or the roto-vibrational state of excitation when molecular species are present in the discharge. One class of physics-based methods that can be used to obtain plasma properties from emission lines are collisional radiative (CR) model. These are effectively semi-empirical micro-kinetic models in which radiative and collisional process are balanced to extract fundamental plasma properties [23–29]. CR models can provide us with information about deviations of f_d from a Maxwellian case [30], but the true shape of f_d cannot be predicted via CR models alone. In addition, theoretical solutions of f_d can be obtained via a two-term approximation of the Boltzmann transport equation under steady-state [31] and Monte Carlo approach [32]. However, these approaches rely on a zero-dimensional assumption and therefore deviate from laboratory and industrial plasmas. Here we show that ML can be successfully utilized to estimate the electron energy distribution starting from OES measurements. We have developed probabilistic deep neural networks (PDNNs) to map line intensities in the plasma emission spectrum to f_p and T_e while quantifying the uncertainty about models and predictions. This approach bypasses the complex physics and inherent uncertainty of CR models to predict plasma parameters such as f_p and T_e without the need of a Langmuir probe system. We build PDNNs, train them on the data, and improve their performance via hyperparameter tuning, regularization, and through advanced optimization algorithm such as the Adam [33, 34].

2. Experimental methods, database collection, and pre-processing

Langmuir probe measurements were conducted in a large 42.0 cm diameter chamber at pressures ranging from 20 to 100 mTorr. The setup is identical to the one utilized for our previous studies on dusty plasmas, although we are focusing on a pristine (dust-free) plasma here [18, 19]. The distance between the grounded base plate and the bottom of alumina plate is 10.0 cm. The plasma is sustained using a radio frequency (13.56 MHz) power supply, with the electrical power coupled into the chamber via an inductive antenna, as shown in figure 1. The plasma powers ranged from 55 to 115 W. The discharge was generated in different gas mixtures containing argon, argon–hydrogen or argon–helium at various compositions. The concentration of gases is controlled by varying their flow rates. The argon flow rate is varied from 50 to 110 sccm in steps of 20 sccm while helium and hydrogen flow rates altered from 2 to 12 sccm in steps of 2 sccm. The stationary Langmuir probe is positioned at the center of the chamber perpendicular to the planar coil. The tungsten probe tip has a length and a diameter of 5.0 mm and 127 μm respectively. The Langmuir probe is passively compensated with a large auxiliary electrode connected to the probe tip via a 200 pF capacitor. A 100 μH inductor resonating at 13.56 MHz is also used as a notch filter to block the first harmonic caused by plasma oscillations. A Wavetek 182 signal generator provides the sweeping voltage required for the Langmuir probe measurements by

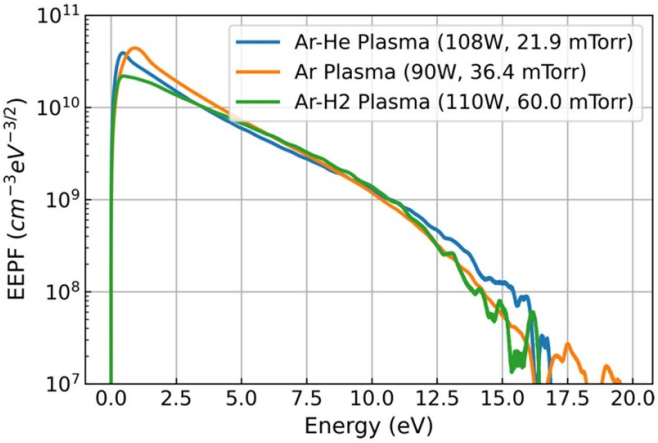

Figure 2. Example of measured electron energy probability functions (EEPFs) for Ar, Ar–He and Ar–H_2 mixtures, at various pressures and input powers.

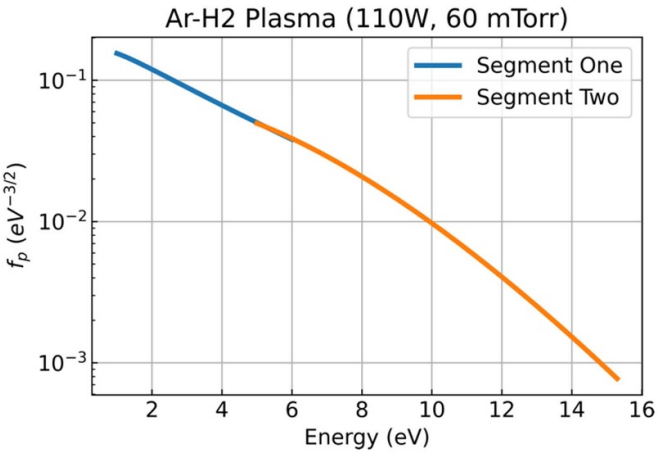

Figure 3. Example of a post-processed and fitted electron energy probability function f_p.

generating a triangular wave at 10 Hz that is amplified using a Kepco BOP 100-1M. A Teledyne DA1855A differential amplifier is used to measure the voltage drop across a 30 Ω resistor and to monitor the probe current. A PicoScope 5442A digital oscilloscope is used to record the IV curves. The current and voltage traces are averaged over 667 waveforms, with the IV acquisition taking approximately 3 min. The Langmuir probe current–voltage waveforms are smoothed via Savitzky Golay averaging. The actual electron energy probability function $\mathrm{EEPF}(E)$ and electron temperature are determined following the Druyvestyn method [35]

$$\mathrm{EEPF}(E) = \frac{\sqrt{8m_e}}{e^3 A_p} \frac{\mathrm{d}{I_e}^2}{d^2 |V_p - V|} \tag{1}$$

$$T_e = \frac{2}{3} \frac{\int_0^\infty E \times \mathrm{EEPF}(E)\,\mathrm{d}E}{\int_0^\infty \mathrm{EEPF}(E)\,\mathrm{d}E} \tag{2}$$

where E is the electron kinetic energy, e the unit electron charge, m_e the electron mass, A_p the probe surface area, V_p the plasma potential, I_e the electron current, and V is the applied sweeping voltage to the probe tip. Adjacent averaging is applied to the second derivative to obtain the EEPF. Figure 2 shows the measured EEPFs for various powers, pressures and gas compositions. The EEPF needs to be encoded into a set of parameters that are learnable and then decoded in post processing to describe the f_p. The choice of fitting parameters must guarantee robustness in the solution, so that any error in the fitting of one parameter does not propagate to other parameters.

The data points below 10^8 cm^{-3} $\mathrm{eV}^{-3/2}$ are discarded as they are below the detection limit of our Langmuir probe setup. The EEPF is converted to actual EEDF (cm^{-3} eV^{-1}) and then normalized to f_d (eV^{-1}). We have found that further dividing f_d by E, and separating it into two segments in the 1–6 eV range and 5 E_{cutoff} eV range, makes the function learnable. The part of the distribution in the 0–1 eV range can be fit to a polynomial function, but those fitting parameters are not learnable meaning there is no correlation between the emission spectra and the parameters. As shown in figure 3, both segments are fitted to an exponential function with three variables given by

$$f_{\mathrm{fit}} = c^{1.5} \times a \times \exp\left(-\left(\frac{E}{c}\right)^b\right). \tag{3}$$

Overall, a total of six fitting parameters describes distribution function. In the post-processing step, $f_{\mathrm{p1,p2}} = f_{\mathrm{fit1,fit2}} \times E^{0.5}$ for each segment is computed using the predicted fitting parameters. In addition to these six fitting parameters, we also add the measured electron temperature as an output of the ML algorithm. The fitting parameters and the electron temperature need to undergo a linear transformation before being used as the outputs to PDNNs. Each output is individually translated and scaled to be between zero and one to ensure equal treatment from PDNNs during training given by

$$Y_{\mathrm{norm}} = \frac{Y - Y_{\mathrm{min}}}{Y_{\mathrm{max}} - Y_{\mathrm{min}}} \tag{4}$$

where Y_{min} and Y_{max} are the minimum and maximum possible values of each feature.

The emission from the plasma is collected through a side port of the plasma chamber. An air-spaced doublet collimator (Thorlabs) connected to a triple grating imaging spectrometer (Acton Spectra Pro, Princeton Instruments) is employed to focus the light into a multimode glass fiber. A CCD camera (1024 × 256) placed at the exit of monochromator's slit with a total recording of 200 frames and an acquisition time of 10 ms is used to capture the emission spectra. The emission spectra in the 300–1000 nm wavelength range are monitored and recorded. A broadband light source (SLS201L, Thorlabs) is used to calibrate the spectral response of the optical system. A correction file is constructed using the calibrated light source to correct for the non-uniform spectral response of the detector. The line intensities are then normalized to the total emission intensity in the 300–1000 nm wavelength range as

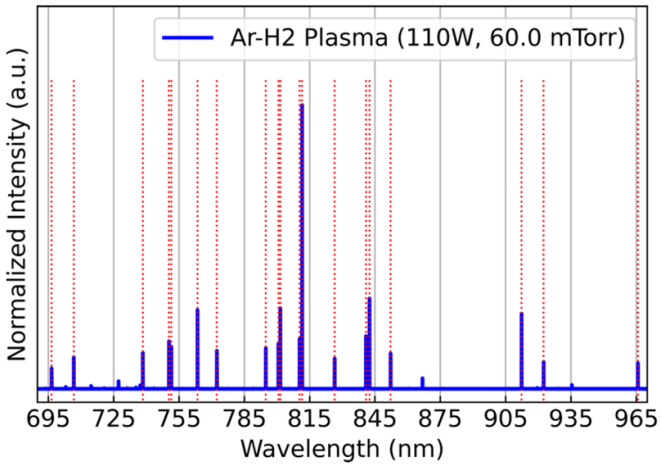

Figure 4. Normalized emission spectrum and the line intensities chosen as input features to PDNNs. The line positions are listed in table 1.

Table 1. The wavelengths and atomic transitions of argon I emission lines used as the input to PDNNs.

Wavelength (nm)	E_i–E_j (eV)
696.54	11.548 – 12.327
706.72	11.548 – 13.302
738.39	11.623 – 13.302
750.38	11.828 – 13.479
751.46	11.623 – 13.273
763.51	11.548 – 13.171
772.42	11.723 – 13.327
794.81	11.723 – 13.282
800.61	11.623 – 13.171
801.47	11.548 – 13.094
810.36	11.623 – 13.153
811.53	11.548 – 13.075
826.45	11.828 – 13.327
840.82	11.828 – 13.302
842.46	11.623 – 13.094
852.14	11.828 – 13.282
912.29	11.548 – 12.907
922.44	11.828 – 13.171
965.77	11.828 – 13.094

shown in figure 4. Nineteen pronounced argon I line intensities are chosen as input features to PDNNs which are located at the following wavelengths (nm) as shown in table 1.

The input (X) is a batch of 19-dimensional vector representing the 19 normalized argon I line intensities, and the output (y) is a batch of seven-dimensional vector representing six f_p fitting parameters and electron temperature T_e. A total of 1400 Langmuir probe and OES measurements were performed for this study.

3. ML methods and architectures

3.1. Neural networks

In general, neural networks effectively recognize hidden patterns and correlations in data. The neural network architecture comprises of three parts: the input layer, the output layer, and series of layers in between called hidden layers. The basic unit of a neural network is a neuron, and each layer consists of a certain number of neurons. Each node or neuron computes a linear function $Z = W^T X + b$ where W and b are the weight and bias matrices, and X is the input matrix. Then, it feeds the output Z to an activation function which is described in the SI document.

The neural network computations are completed in terms of forward and backward propagation steps [36]. Initially, the weight matrices (W) are randomly initialized, and biases (b) are initialized to be zero. In the forward pass, the input matrix (X) is fed into the densely connected network, and the outputs are computed from the last layer based on the initialized network parameters. A loss function is created by comparing the discrepancy between this predicted output and the desired output. A cost function is then generated by the averaging of the loss function over the entire training set. The appropriate cost functions for a deterministic neural network are the mean absolute error (MAE) and mean square error (MSE). Their exact definition can be found in the SI document. The ML algorithm minimizes the cost function by tuning W and b of all neurons in the network by implementing gradient descent algorithm [37] in backward propagation step. The gradients of the cost function with respect to weights W and biases b of every layer are computed. W and b are iteratively updated accordingly at a learning rate α given by

$$\begin{cases} (W_{[l]} = W_{[l]} - \alpha \times \frac{\mathrm{d(Cost)}}{\mathrm{d}W_{[l]}} \\ b_{[l]} = b_{[l]} - \alpha \times \frac{\mathrm{d(Cost)}}{\mathrm{d}b_{[l]}}). \end{cases} \tag{6}$$

Each training cycle is called an epoch (or iteration). This iterative process continues until the cost function is minimized. The hyperparameter α also needs to be tuned to minimize the error. The gradients of the cost function with respect to weights and biases can be noisy as they can oscillate through this iterative process. In this work, Adam optimization algorithm which is an extension to gradient descent is used instead to damp out the oscillations and make the model reach a faster convergence towards minimized cost [33, 34]. This algorithm is built into TensorFlow [38] which is an end-to-end open-source ML platform developed by Google. We use TensorFlow to develop and train our ML models. The available dataset is split into a training set (60%), a development set (20%), and a test set (20%). It is important that both the development and test sets comes from the same data set. This gives an unbiased estimate of the performance of the neural network and ensure the model has a good generalization ability, i.e. it can apply the knowledge gained from the training set onto other data.

3.2. Non-Bayesian neural networks: maximum likelihood principle

Statistical variability in experimental dataset is inevitable. Therefore, a neural network should assign a level of uncertainty to its predictions. Uncertainty can be categorized as either aleatoric or epistemic. The aleatoric uncertainty is due to the inherent noise in the measured Langmuir probe and

spectroscopic data. Epistemic uncertainty is due to the finite size of the experimental dataset. For example, the Langmuir probe measurements discussed in this manuscript were performed in the 20–100 mTorr range. The model needs to assign a higher level of uncertainty to its predictions while encountering inputs outside of this pressure range, as it was not trained on them. We account only for the aleatoric uncertainty by implementing the maximum likelihood principle [39, 40]. In this approach, each output is described not by a single value, but rather by a Gaussian distribution having a mean and a standard deviation value. In this probabilistic model, the negative log-likelihood which is described in SI document is used as cost function instead of MAE or MSE. The weights and biases of the model are trained to predict the appropriate mean and standard deviation for each output feature by simply minimizing the negative-log-likelihood.

3.3. Bayesian neural networks: variational inference (VI) and Monte Carlo dropout (MCD)

An approach to handling both aleatoric and epistemic uncertainties associated with neural networks, is to move away to associating fixed values to the weight and bias of each neuron, but rather describe them via a distribution. Effectively, this approach uses the knowledge of many networks instead a single neural network. The distribution of all weights and biases can be learned from the training data using Bayes' theorem. However, solving these complex distributions is computational expensive. Two approaches have been tested to solve this problem: the VI and the MCD methods.

In the VI method, the distributions of all weights and biases are approximated to Gaussian distributions. Upon convergence, these Gaussian distributions are as close as possible to complex distributions of all weights and biases. A metric to describe the difference between the two is called Kullback–Leibler (KL) divergence [40] which we use as the cost function to train this model as [41, 42]. The exact definition can be found in the SI document.

Alternatively, the MCD approach [43, 44] approximates the complex distribution of weights and biases with a binary distribution. The idea is to train an ensemble of thin version of neural network that share weights and biases using dropout technique [37]. During the training process, dropout technique is applied by going through each of the layers of the neural network and randomly eliminating a number of output of nodes per layer [37]. So, every iteration, a thin version of network is used to train the model. At the end of this iterative process, we have trained many thin versions of neural network that can make predictions. By using dropout during test time, we can make predictions with a different thin version of neural network meaning we can use knowledge of many networks instead of one. We used negative-log-likelihood as the cost function to train this model.

4. Implementation

We have implemented the non-Bayesian and two Bayesian models in TensorFlow. The architecture of models is described in the supplementary section (available online at stacks.iop.org/JPD/54/265202/mmedia), and the codes can be accessed on our GitHub page [45]. For the non-Bayesian model, the Adam optimization algorithm is used with initial learning rate of $\alpha_0 = 10^{-3}$ and further decreased according to

$$\alpha = \alpha_0 \times \left(1 - 4.15 \times \frac{t}{N_t}\right)^{0.75} \tag{7}$$

where t and N_t are the current epoch and total number of epochs which is 10 000 in this case. For the Bayesian model approximated via VI technique, the Adam optimization algorithm is used with a constant $\alpha = 4 \times 10^{-4}$, and the model is trained for 10 000 epochs. For the Bayesian model approximated by the MCD method, the Adam optimization algorithm is used with a constant $\alpha = 2 \times 10^{-4}$, and the model is trained for 25 000 epochs. We have evaluated the performance of our models by comparing the measured and predicted mean f_p, T_e while considering the uncertainty contours.

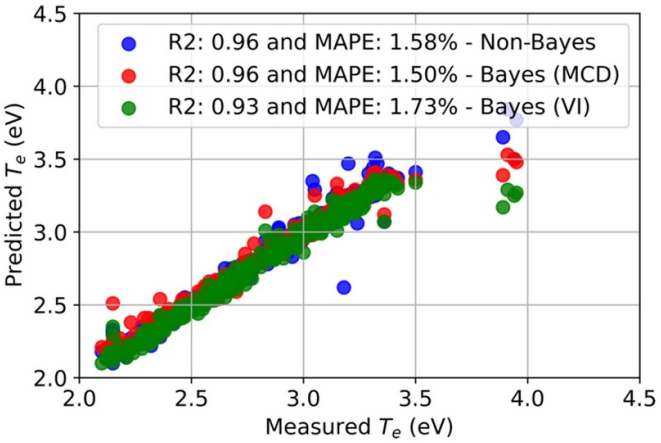

Figure 5. The relation between the measured and predicted T_e with non-Bayesian and Bayesian (variational inference and Monte Carlo dropout) models for all the testing data.

Table 2. The coefficient of determination R^2 between the predicted and measured f_p fitting parameters for all testing data.

	Model		
R^2 score	Non-Bayes	Bayes (MCD)	Bayes (VI)
a	0.73	0.86	0.85
b	0.74	0.82	0.78
c	0.74	0.77	0.79
d	0.76	0.87	0.87
e	0.89	0.89	0.89
g	0.92	0.95	0.94
T_e	0.96	0.96	0.94

5. Results

The architecture of all PDNNs is optimized over many iterations to ensure models have low bias and variance meaning the

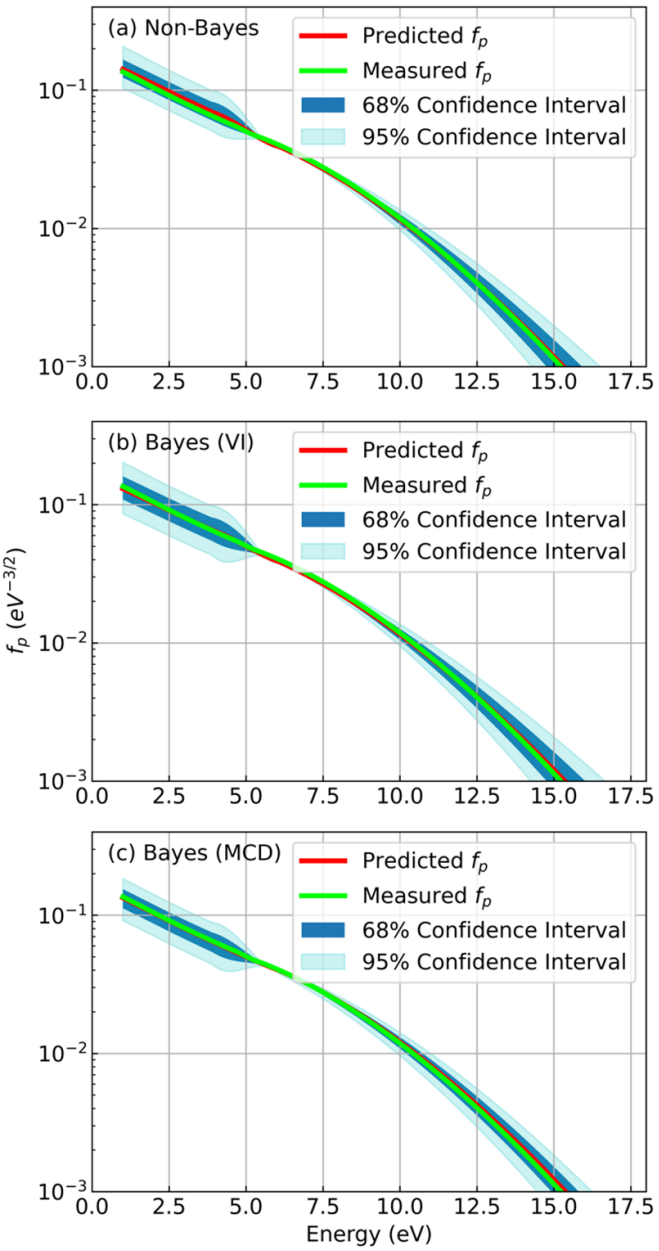

Figure 6. Measured and predicted f_p with 68% and 95% confidence intervals, predicted by a non-Bayesian (a) and Bayesian models approximated via variational inference (b) and Monte Carlo dropout (c) methods. The distributions are obtained using a test data that is similar to the data they were trained on.

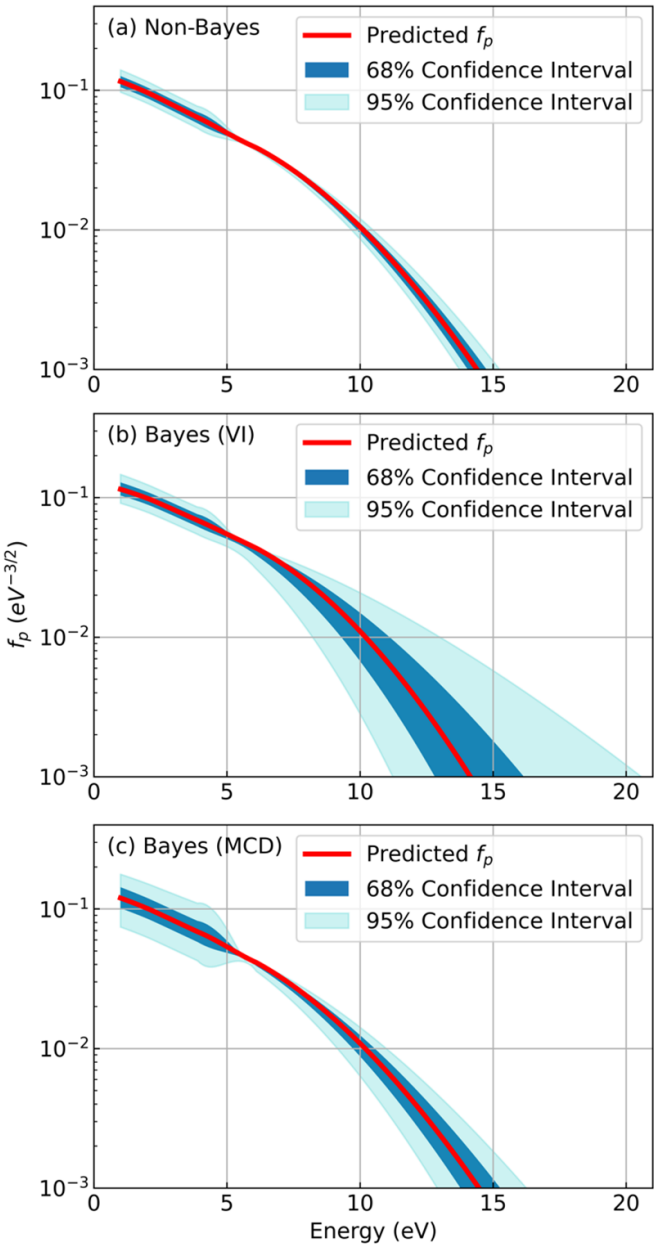

Figure 7. Predicted f_p with 68% and 95% confidence intervals by non-Bayesian (a) and Bayesian models approximated via variational inference (b) and Monte Carlo dropout (c) methods, using a test data dissimilar from the data they were trained on.

training and validation errors are as small as possible. Methods to reduce bias and variance of the model are described in SI document. To better analyze the performance of our models, the coefficient of determination R^2 and mean absolute percentage error (MAPE) between the measured and predicted mean output features are computed. In figure 5, we show the relation between the measured and predicted electron temperatures for all testing data. The R^2 scores are 0.962, 0.960 and 0.934 and MAPE are 1.58%, 1.50% and 1.73% for the non-Bayesian and Bayesian models approximated via VI and MCD techniques. The R^2 scores for all three cases suggest a high level of fit. The models are generalizing well as they can examine more general input circumstances and do not simply memorize the training samples.

The coefficients of determination for the six fitting parameters used to estimate the shape of the distribution function (a, b, c, d, e, g) are also computed as presented in table 2. a, b and c are used in the exponential function to generate segment one (in the 1–5 eV range), and d, e and g are utilized to create segment two (for energies larger than 5 eV). The R^2 scores of parameters fitting the tail of the distribution (d, e, g) are larger than (a, b, c) implying models did a better job at predicting the tail of the distribution than the 1–5 eV range segment of f_p. Overall, high R^2 scores and

small MAPE indicate all models are predicting f_p with high accuracy.

Next, we compared the capability of the non-Bayesian and Bayesian models in capturing both the epistemic and aleatoric uncertainties. All non-Bayesian and Bayesian networks describe the uncertainty in their predictions. For the non-Bayesian neural network, the model parameters are fixed meaning it yields identical results for a given input, and the standard deviation of each output feature represents the aleatoric uncertainty in the predictions. Bayesian neural networks, approximated either by VI or by MCD methods, have model parameters that are characterized by Gaussian and binary distributions. Therefore, the model prediction for an input is different every run since a different thin version of a network is used to make predictions each time. To obtain the empirical outcome distribution for the non-Bayesian and Bayesian models, we did run each model 500 times and sampled from the predicted Gaussian distribution of each output feature. All the 500 results of each output feature are summarized by a mean, the 95% confidence and the 68% confidence interval. The corresponding f_p are generated and the contours describing the uncertainty in the predicted f_p are drawn for the non-Bayesian and Bayesian models. Using one sample of inputs drawn from the testing data, all models yield similar results as expected, as shown in figure 6. All models capture the aleatoric uncertainty. To test the ability of the model in capturing the epistemic uncertainty, we fed into the models an input which is dissimilar from the data they were trained on, i.e. with input values that are far from the average range of the training set. As shown in figure 7, the non-Bayesian approach assigned unrealistically narrow uncertainty contours to f_p. However, the Bayesian models correctly assign larger uncertainty contours to f_p.

6. Conclusion

In this manuscript, we report for the first time the use of ML for the estimation of the electron energy distribution in a low pressure, low temperature non-thermal plasmas. We have completed 1400 Langmuir probe and OES measurements. We pre-processed, engineered, and used the data to train ML models. We have explored different model architectures for the development of the probabilistic deep learning networks. Non-Bayesian and Bayesian neural networks are evaluated to quantify and manipulate uncertainty in the predictions. We found that the best performing approach to predict electron energy distribution is using Bayesian models approximated via VI and MCD methods as they can correctly capture both aleatoric and epistemic uncertainties. Further improvements in accuracy can be obtained by accumulating larger data set, thus reducing the epistemic uncertainty of the models. We envision and advocate for a more widespread utilization of these models, since they can provide reliable estimates of fundamental plasma parameters. One challenge associated with this approach is the need for large experimental data sets to train the neural network. As a consequence, ML models will likely find broader application, for instance, in industrial settings where large data sets are available. However, Bayesian deep neural networks can quantify the degree of uncertainty in the models and their predictions, so that academic research groups can still build probabilistic networks while accounting for aleatoric and epistemic uncertainty. Further reduction in the level of uncertainty can be achieved over extended period of time by expanding the datasets and by sharing of data among different research groups.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

This work was supported by the US Department of Energy, Office if Science, Early Career Research Program under Award No. DESC0014169.

ORCID iD

Lorenzo Mangolini https://orcid.org/0000-0002-0057-2450

References

[1] Hatamizadeh A, Terzopoulos D and Myronenko A 2019 End-to-end boundary aware networks for medical image segmentation *Machine Learning in Medical Imaging* (Berlin: Springer) pp 187–94
[2] Levinson J *et al* 2011 Towards fully autonomous driving: systems and algorithms *2011 IEEE Intelligent Vehicles Symp. (IV)* (IEEE) pp 163–8
[3] Esteva A, Kuprel B, Novoa R A, Ko J, Swetter S M, Blau H M and Thrun S 2017 Dermatologist-level classification of skin cancer with deep neural networks *Nature* **542** 115–8
[4] Haenssle H A *et al* 2018 Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists *Ann. Oncol.* **29** 1836–42
[5] Kooi T, Litjens G, van Ginneken B, Gubern-Mérida A, Sánchez C I, Mann R, den Heeten A and Karssemeijer N 2017 Large scale deep learning for computer aided detection of mammographic lesions *Med. Image Anal.* **35** 303–12
[6] Jamaludin A, Kadir T and Zisserman A 2016 SpineNet: automatically pinpointing classification evidence in spinal MRIs *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016* (Berlin: Springer) pp 166–75
[7] Barreira C M, Bouslama M, Haussen D C, Grossberg J A, Baxter B, Devlin T, Frankel M and Nogueira R G 2018 Abstract WP61: automated large artery occlusion detection IN stroke imaging—ALADIN study *Stroke* **49** AWP61
[8] Esteva A, Robicquet A, Ramsundar B, Kuleshov V, DePristo M, Chou K, Cui C, Corrado G, Thrun S and Dean J 2019 A guide to deep learning in healthcare *Nat. Med.* **25** 24–29
[9] Ren S, He K, Girshick R and Sun J 2015 Faster R-CNN: towards real-time object detection with region proposal networks (available at: http://arxiv.org/abs/1506.01497)

[10] He K, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (IEEE) pp 770–8
[11] Noh H, Hong S and Han B 2015 Learning deconvolution network for semantic segmentation pp 1520–8 (available at: https://openaccess.thecvf.com/content_iccv_2015/html/Noh_Learning_Deconvolution_Network_ICCV_2015_paper.html) (Accessed 15 February 2021)
[12] Krizhevsky A, Sutskever I and Hinton G E 2017 ImageNet classification with deep convolutional neural networks *Commun. ACM* **60** 84–90
[13] He K, Zhang X, Ren S and Sun J 2015 Delving deep into rectifiers: surpassing human-level performance on ImageNet classification pp 1026–34 (available at: https://openaccess.thecvf.com/content_iccv_2015/html/He_Delving_Deep_into_ICCV_2015_paper.html) (Accessed 15 February 2021)
[14] Fridman L *et al* 2019 MIT advanced vehicle technology study: large-scale naturalistic driving study of driver behavior and interaction with automation *IEEE Access* **7** 102021–38
[15] Ferreira D R, Carvalho P J and Fernandes H 2020 Deep learning for plasma tomography and disruption prediction from bolometer data *IEEE Trans. Plasma Sci. IEEE Nucl. Plasma Sci. Soc.* **48** 36–45
[16] Mesbah A and Graves D B 2019 Machine learning for modeling, diagnostics, and control of non-equilibrium plasmas *J. Phys. D: Appl. Phys.* **52** 30LT02
[17] Choudhury T A, Hosseinzadeh N and Berndt C C 2011 Artificial neural network application for predicting in-flight particle characteristics of an atmospheric plasma spray process *Surf. Coat. Technol.* **205** 4886–95
[18] Woodard A, Shojaei K, Nava G and Mangolini L 2018 Langmuir probe characterisation of an Ar–H_2 non-thermal plasma loaded with carbon nanoparticles *Plasma Sources Sci. Technol.* **27** 104003
[19] Woodard A, Shojaei K, Berrospe-Rodriguez C, Nava G and Mangolini L 2020 Electron emission from particles strongly affects the electron energy distribution in dusty plasmas *J. Vac. Sci. Technol.* A **38** 023005
[20] Maresca A, Orlov K and Kortshagen U 2002 Experimental study of diffusive cooling of electrons in a pulsed inductively coupled plasma *Phys. Rev.* E **65** 056405
[21] Godyak V A, Piejak R B and Alexandrovich B M 2002 Electron energy distribution function measurements and plasma parameters in inductively coupled argon plasma *Plasma Sources Sci. Technol.* **11** 525–43
[22] Godyak V A, Piejak R B and Alexandrovich B M 1992 Evolution of the electron-energy-distribution function during rf discharge transition to the high-voltage mode *Phys. Rev. Lett.* **68** 40–43
[23] Zhu X-M and Pu Y-K 2007 A simple collisional–radiative model for low-pressure argon discharges *J. Phys. D: Appl. Phys.* **40** 2533–8
[24] Thorne A, Litzen U and Johansson S 1999 *Spectrophysics: Principles and Applications* (Berlin: Springer)
[25] Fujimoto T 2004 *Plasma Spectroscopy* (Oxford: Clarendon)
[26] Fantz U 2004 Emission spectroscopy of molecular low pressure plasmas *Contrib. Plasma Phys.* **44** 508–15
[27] Fantz U 2005 Molecular diagnostics of fusion and laboratory plasmas *AIP Conf. Proc.* (AIP)
[28] Behringer K and Fantz U 1994 Spectroscopic diagnostics of glow discharge plasmas with non-Maxwellian electron energy distributions *J. Phys. D: Appl. Phys.* **27** 2128–35
[29] Vinogradov I P 1999 Development and applications of spectroscopic determinations of the electron distribution function in discharges *Plasma Sources Sci. Technol.* **8** 299–312
[30] Fantz U 2006 Basics of plasma spectroscopy *Plasma Sources Sci. Technol.* **15** S137–47
[31] Hagelaar G J M and Pitchford L C 2005 Solving the Boltzmann equation to obtain electron transport coefficients and rate coefficients for fluid models *Plasma Sources Sci. Technol.* **14** 722–33
[32] Capitelli M *et al* 2016 Kinetic and Monte Carlo approaches to solve Boltzmann equation for the electron energy distribution functions *Fundamental Aspects of Plasma Chemical Physics* (New York: Springer) pp 79–111
[33] Ruder S An overview of gradient descent optimization algorithms* (Arxiv.org) (available at: http://arxiv.org/abs/1609.04747v2) (Accessed 15 February 2021)
[34] Kingma D P and Ba J 2014 Adam: a method for stochastic optimization (available at: http://arxiv.org/abs/1412.6980)
[35] Druyvesteyn M J 1930 Der Niedervoltbogen *Eur. Phys. J.* A **64** 781–98
[36] Ruder S 2016 An overview of gradient descent optimization algorithms (available at: http://arxiv.org/abs/1609.04747)
[37] Schmidhuber J 2015 Deep learning in neural networks: an overview *Neural Netw.* **61** 85–117
[38] Abadi M *et al* 2016 TensorFlow: a system for large-scale machine learning *12th {USENIX} Symp. on Operating Systems Design and Implementation ({OSDI} 16)* pp 265–83
[39] Bengio Y Le Cun Presenter: Marilyn Vazquez Y. Scaling learning algorithms towards AI (Semanticscholar.org) (available at: https://pdfs.semanticscholar.org/f01e/080777b59d6978e412ded8995edabbaa62f0.pdf) (Accessed 15 February 2021)
[40] Dürr O, Sick B and Murina E 2020 *Probabilistic Deep Learning: With Python, Keras, and TensorFlow Probability* (Shelter Island, NY: Manning Publications)
[41] Blundell C, Cornebise J, Kavukcuoglu K and Wierstra D 2015 Weight uncertainty in neural networks (available at: http://arxiv.org/abs/1505.05424)
[42] Wen Y, Vicol P, Ba J, Tran D and Grosse R 2018 Flipout: efficient pseudo-independent weight perturbations on mini-batches (available at: http://arxiv.org/abs/1803.04386)
[43] Gal Y and Ghahramani Z 2015 Dropout as a Bayesian approximation: appendix (available at: http://arxiv.org/abs/1506.02157)
[44] Ghahramani Z 2015 Probabilistic machine learning and artificial intelligence *Nature* **521** 452–9
[45] Shojaei K (Available at: https://github.com/kshojaei/MachineLearning-Plasma.git)