Uncertainty quantification for virtual diagnostic of particle accelerators

Owen Convery,[1] Lewis Smith,[2] Yarin Gal,[2] and Adi Hanuka[1,*]

[1]*SLAC National Accelerator Laboratory, Menlo Park, California 94025, USA*

[2]*Department of Computer Science, University of Oxford, Oxford, OX1 3QD, United Kingdom*

(Received 28 January 2021; accepted 14 June 2021; published 21 July 2021)

Virtual diagnostic (VD) is a computational tool based on deep learning that can be used to predict a diagnostic output. VDs are especially useful in systems where measuring the output is invasive, limited, costly or runs the risk of altering the output. Given a prediction, it is necessary to relay how reliable that prediction is, i.e., quantify the uncertainty of the prediction. In this paper, we use ensemble methods and quantile regression neural networks to explore different ways of creating and analyzing prediction's uncertainty on experimental data from the Linac Coherent Light Source at SLAC National Lab. We aim to accurately and confidently predict the current profile or longitudinal phase space images of the electron beam. The ability to make informed decisions under uncertainty is crucial for reliable deployment of deep learning tools on safety-critical systems as particle accelerators.

DOI: 10.1103/PhysRevAccelBeams.24.074602

I. INTRODUCTION

Particle accelerators serve a wide variety of applications ranging from chemistry, physics to biology experiments. Those experiments require increased accuracy of diagnostics tools to measure the beam properties during its acceleration, transport, and delivery to users. Diagnostics must keep pace with the advance of extreme beam conditions and the increased experiments' complexity, which presents challenges to the current state-of-the-art [1,2]. Machine learning (ML) applications for accelerator diagnostics have been recently shown to improve beam stability [3], identify glitchy accelerator components [4], perform optics corrections and detect faulty beam position monitors [5,6], and control in real-time by combining adaptive feedback and ML [7].

Given readily available input data, virtual diagnostic (VD) tools provide a shot-to-shot noninvasive measurement of the beam in cases where the diagnostic has limited resolution or limited availability [8–10]. VDs have the potential to be useful in an experiment's design, setup, and optimization while saving valuable operation time. They could also aid in interpreting experimental results, especially in cases in which current diagnostics cannot provide necessary information.

Current VD provides predictive models based on training a neural network mapping between noninvasive diagnostic input to invasive output measurements [8,9,11]. This type of mapping is known as supervised regression. Previous work has demonstrated VD to predict the electron beam current profile and longitudinal phase space (LPS) distribution [12] along the accelerator using either scalar controls [8] or spectral information [9] as the noninvasive input to the VD. However, while current VD methods provide mean prediction only, an essential component in deploying the virtual diagnostic tool is to quantify the confidence in the prediction, i.e., estimate an interval presenting the uncertainty in prediction. A metric of uncertainty and its reliable presentation provide a way of making informed decisions, that becomes crucial in safety-critical systems such as particle accelerators.

In general, there are two classes of uncertainty: epistemic and aleatoric uncertainty [13]. Aleatoric, or statistical irreducible uncertainty, is the uncertainty in the dataset that arises from experimental error or inherent measurement noise. Given the same set of inputs, we may observe slightly different results. Epistemic uncertainty, also known as systematic reducible uncertainty, is produced when the model's knowledge is limited or hindered, e.g., from missing data. We aim to capture both types of uncertainty in the VD tool and to incorporate them into the final prediction. While the neural network can only make point predictions about beam properties, we can use tools from deep learning to better understand the uncertainty of the predictions.

There are various ways to quantify uncertainty of machine learning models for supervised regression problems. These include Bayesian model averaging approaches such as Gaussian processes [14] and Bayesian neural

[*] Corresponding author.
adiha@slac.stanford.edu

networks (BNNs) [15], which learn distributions over the parameters of the network. Compared to Gaussian processes, BNNs have the advantage that they can easily scale to high dimensional inputs. However, BNNs are computationally expensive and require modifications to the training procedure. Other non-Bayesian approaches widely used are bootstrapping, ensembling [16,17] and quantile regression [18,19]. Those non-Bayesian approaches are simple to implement, could be easily parallelized to scale with large amount of data, and yields high quality predictive uncertainty estimates [20,21]. Ensembles are closely related to Bayesian model averaging, and ensembles of neural networks can be seen as an *ad hoc* approximation of Bayesian neural networks or Gaussian processes for specific architectures and training conditions [22–24]. When each ensemble component is trained with random initialization *and* a random data subset, the procedure is known as bootstrap aggregation, or bagging. Bootstrapping is well studied in the frequentist statistics literature and is closely related to Bayesian statistics with certain priors [16,25,26].

In this paper, we apply deep learning tools to provide confidence intervals for virtual diagnostic predictions. We compare the predicted uncertainty learned by multiple ensemble methods as well as quantile regression using experimental data from the Linac Coherent Light Source (LCLS) at SLAC National Lab [27]. We evaluate the robustness of the ensemble methods to provide accurate mean predictions as well as reliable uncertainty estimation.

The paper is organized as follows: in Sec. II we first present the VD architecture. Then we discuss metrics to evaluate the reliability of the prediction to be used online on the particle accelerator. In Sec. III we demonstrate the methods and compare them on two experimental datasets from LCLS; 1D current profiles and 2D longitudinal phase space images. The 1D current profile is a projection of the LPS image. Finally, in Sec. IV we discuss reweighing the datasets, and other neural network architectures to improve the results.

II. METHODS

In this section we first present the virtual diagnostic (VD) neural network architecture and the datasets. We then present various ensemble and quantile methods to quantify the uncertainty. Last, we present the metrics used to evaluate the accuracy of the VD's mean prediction and uncertainty.

A. VD architecture and dataset

High brightness beam linacs typically operate in single-pass, multistage configurations where a high-density electron beam formed in the rf gun is accelerated and manipulated prior to delivery to users in an experimental station. An example of such a facility is the LCLS XFEL at SLAC where the electron beam traverses through an undulator, and emits coherent x-ray pulses. Typically, longitudinal phase space (LPS) is destructively measured by an X-band transverse deflecting cavity (XTCAV) [1,28]—as shown in Fig. 1.

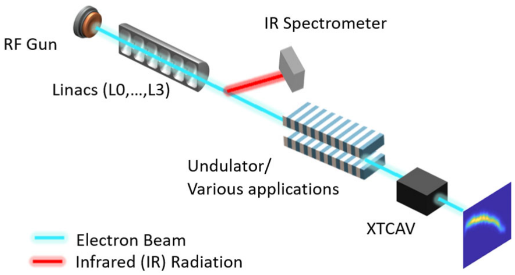
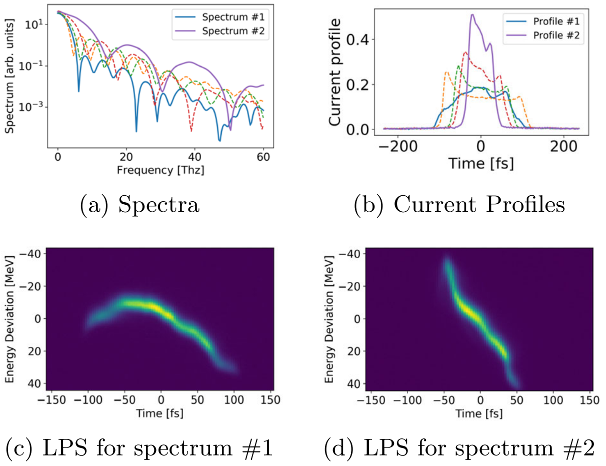


FIG. 1. Schematic of a generic, linac-driven light source. Reproduced from [9].

In this paper we used two experimental datasets: 2D LPS images or 1D current profiles, measured at the XTCAV, as the outputs, and the corresponding spectral information, as can be collected by IR spectrometer, as an input. The 1D current profile is a projection of an LPS image. The outputs have been centered and cropped about their region of interest as in Ref. [8]. Examples of the inputs and outputs are shown in Fig. 2. Both datasets contain 4,046 shots and were randomly shuffled and split 80% for training and validation, and 20% for testing. We then normalized all output profiles and images. The outputs shown in the paper are all normalized.

The neural network (NN) architecture we used is a fully connected feed-forward NN composed of three hidden layers (200, 100, 50) with rectified linear unit activation function. Other network architectures are discussed in Sec. IV. For training we used a batch size of 32, 500 epochs and the Adam optimizer with a fixed learning rate of 0.001 [9]. Training the NNs with a Gaussian likelihood,



(a) Spectra (b) Current Profiles

(c) LPS for spectrum #1 (d) LPS for spectrum #2

FIG. 2. Example of VD input of spectrum in (a) and corresponding outputs; either 1D current profiles in (b), or images of longitudinal phase space (LPS)—examples of which are shown in (c) and (d) for two different spectra input.

i.e., to minimize the standard mean squared error (MSE) loss function on a training set, yields symmetric uncertainty intervals. To introduce asymmetric uncertainty intervals relative to the mean value, we trained on quantiles of data as described in Sec. II B. The open source Keras and TensorFlow libraries were used to build and train the NN module [29,30].

B. Uncertainty quantification methods

Various ways exist to estimate an interval presenting the uncertainty in the neural network prediction. In what follows, we describe a few ensemble methods, wherein a collection of various neural networks is combined together. In this case, the variance of the ensemble is an estimate of the uncertainty. We then describe quantile regression to predict multiple quantiles of the data. We also applied the popular MC dropout technique [31]. However, we observed a degraded mean prediction and inaccurate uncertainty bounds compared to ensembling across the entire dataset. Further improvement could combine ensembles and MC dropout but this is beyond the scope of this paper.

1. Ensemble methods

Deep ensembles are ensembles of neural networks, each initialized differently and optimized to converge independently of the others. Reference [20] has shown empirically that explicit ensembling can result in improved uncertainty estimates when they used large neural networks with nonconvex loss surfaces. Deep ensembles are closely related to Bayesian neural networks and Gaussian processes. For example, Ref. [24] has recently shown that with a small adaptation to their training procedure, deep ensemble components can be interpreted as draws from a Gaussian process posterior with a certain kernel.

We investigated three ways of generating this ensemble: using random initialization of the NN parameters, using a random subset of the data (e.g., k-fold cross validation), and bagging (e.g., using both). The predicted current profile for a test shot $\vec{I}_{\text{predicted}} = M^{-1}\sum_{m=1}^{M}\vec{i}_{\text{predicted},m}$ is the mean prediction of M neural network predictions $\vec{i}_{\text{predicted},m}$. The uncertainty is manifested as the standard deviation of the neural network predictions $\vec{\sigma} = \sqrt{M^{-1}\sum_{m=1}^{M}(\vec{i}_{\text{predicted},m} - \vec{I}_{\text{predicted}})^2}$.

a.Random parameters initialization

Random initialization of the neural network parameters (e.g., weights) is a promising approach for improving prediction accuracy [32] and uncertainty [20]. It has been recently shown that random initializations explore entirely diverse solutions in function space [33] rather than collapse to the same solution [22]. Popular initializations used in many applications are the He and Glorot uniform distributions [34,35]. Those initializations often lead to quick and reliable convergence during training. However, to create a more diverse ensemble of neural networks, a random normal initialization has the potential to reduce the amount of near identical solutions. Here, we initialized all NN parameters as independent and identically distributed random variables from $\mathcal{N}(0, 0.05)$ when training on the same predetermined training/validation data split. Glorot uniform is used for initialization in all other parts of this paper.

b.K-fold cross validation

Rather than using the same training/validation split for each NN in the ensemble, we randomly select a subset of the data. We fixed the seed for a Glorot uniform distribution for all ensemble components. We split the training dataset into K partitions for K neural networks. Using a different partition for validating each model would yield a global model that can generalize better since it has been trained/validated using many subsets of the same data. This is a more responsible way to validate models for ensembles [36].

c.Bagging

Bootstrap AGGregatING (bagging) is an ensemble method that trains each NN on a different ‘bag’ of data. Each bag randomly (with replacement) contains n' out of n possible data points where n'/n is typically ~60%. All ensemble components are initialized with a different Glorot uniform seed, resulting in a NN model with both random subsampling of the data and random initialization. Bagging is a method commonly used to reduce variance and avoid overfitting [37]. Figure 3(a) shows an example of the measured current profile (normalized) $\vec{I}_{\text{measured}}$, its mean prediction $\vec{I}_{\text{predict}}$ and the uncertainty ($\pm 2\vec{\sigma}$) of bagging with 16 ensemble components.

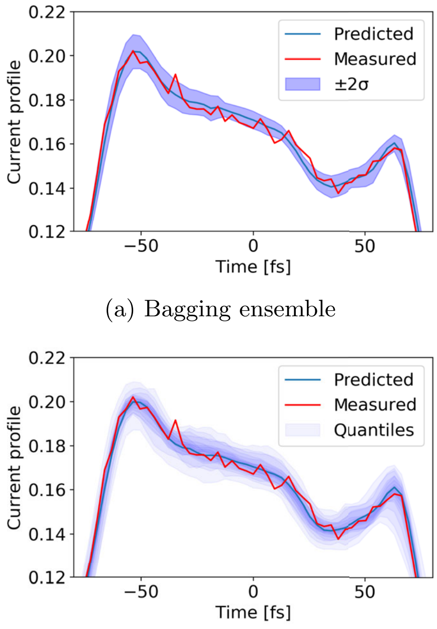
2. Quantile regression

Neural networks can be trained to predict multiple quantiles of the data [18] by using a tilted loss function

$$\mathcal{L}(\zeta_t|\tau) = \begin{cases} \tau\zeta_t & \text{if } \zeta_t \geq 0 \\ (\tau - 1)\zeta_t & \text{if } \zeta_t < 0 \end{cases} \tag{1}$$

where $\zeta_t = I_{\text{measured},t} - I_{\text{predicted},t}$ for each time index t. The average loss over the entire shot of length T is $T^{-1}\sum_{t=1}^{T}\mathcal{L}(\zeta_t|\tau)$. In practice, a separate NN is used to predict each quantile. Each is trained using the tilted loss function but with a different τ value corresponding to the desired quantile, 0.5 being the median prediction.

Figure 3(b) shows 19 quantiles between 0.05 and 0.95. When plotting multiple quantile lines at once, we can gain better insight into where the ground truth may lie. We used the median quantile ($\tau = 0.50$) for calculating the MSE of the prediction.



(a) Bagging ensemble

(b) Quantile regression

FIG. 3. Normalized measured current profile (red) of shot #729 and its prediction (blue) using (a) bagging with 16 ensembles and with $\pm 2\sigma$ uncertainty interval, and (b) quantile regression with 90% uncertainty interval.

C. Metrics for Evaluation

In order to compare different approaches to quantify uncertainty, we need to have a way to evaluate the desirable characteristics of the uncertainty. Those metrics of the predictive uncertainty quality will be used in real-time during the machine operations as well to indicate the validation of the VD's prediction. We used a custom accuracy metric to evaluate the model's performance in capturing the ground truth. At each time index (t) in the shot, we check whether the measured ground truth value ($I_{\text{measured},t}$) was within an upper $I_{\text{upper},t}$ and lower $I_{\text{lower},t}$ prediction bounds. We then weight it by the measured value to penalize the near-zero noise while focusing on important features in the signal. The prediction accuracy of a shot with a length T is defined as:

$$\text{Accuracy} = \frac{\sum_{t=1}^{T} \alpha_t \cdot I^2_{\text{measured},t}}{\sum_{t=1}^{T} I^2_{\text{measured},t}} \qquad (2)$$

where $\alpha_t = \mathbb{1}(I_{\text{lower},t} < I_{\text{measured},t} < I_{\text{upper},t})$. Unless otherwise stated, for a symmetric loss function we used bounds of $I_{\text{upper},t}, I_{\text{lower},t} = I_{\text{predicted},t} \pm 2\sigma_t$ where σ_t is predictive standard deviation at time t. For the tilted loss function, the NN predictions using $\tau = 0.05$ and $\tau = 0.95$ were used for I_{lower} and I_{upper} respectively. We chose to use $\pm 2\sigma$ of the ensembles and 0.90 (0.95-0.05) of the quantiles for these calculations because we wanted to capture ≥90% of the uncertainties correlated with each method.

Since the measured ground truth will not be available when operating the machine in real-time, in order to discern a good prediction from a poor prediction we correlate between the mean squared error between the measured and predicted values—$\text{MSE} = T^{-1} \sum_{t=1}^{T} (I_{\text{measured},t} - I_{\text{predicted},t})^2$—for a vector length of T and the maximum predicted uncertainty $\sigma_{\text{max}} = \max\{\vec{\sigma}\}$. A positive correlation indicates that the model can accurately inform which predictions have low vs high uncertainty before seeing the ground truth. If choosing to retain/reject predictions in practice, a retention curve can be used to set a proper threshold for deployment; we retain (and predict) shots with maximum standard deviation per shot smaller than a given threshold.

III. RESULTS

In this section, we evaluate the methods described in Sec. II B with the metrics discussed in Sec. II C on the 1D current profiles. Next, we apply those techniques to the 2D LPS image dataset. Finally, we discuss problems that are unique to the images and how they can be relieved.

A. 1D current profiles

Before exploring the quantified uncertainty using the 1D current profile dataset, we first found the ensemble method which yielded the best MSE on the mean prediction. Figure 4 shows the MSE vs ensemble size for the three ensemble methods. Bagging has a better MSE than random initialization and performs slightly worse than the K-fold ensemble for small ensemble size. Since bagging with 16 ensemble components yielded the smallest MSE, we evaluated this model's performance using the correlation plot and retention curve described in section II C.

Figure 5 shows retention curve, which can be used to set a rejection threshold for deployment of the VD. Here, MSE_θ is the average MSE for all test shots with σ_{max} lower than a given threshold θ displayed on the color bar. The Retention_θ on the x-axis describes the percentage of test shots we retain vs reject in the process of choosing a threshold θ. If a VD shot prediction yields a σ_{max} higher than the set θ value, we ignore the prediction.

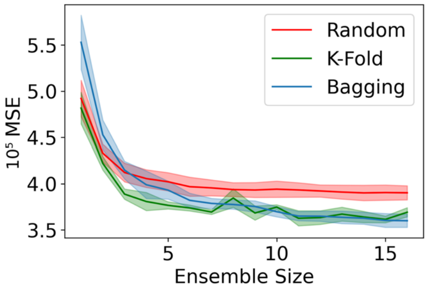


FIG. 4. MSE vs ensemble size for three ensemble methods: Random initializations (red), K-fold cross validation (green), bagging (blue) with standard deviation from five runs.

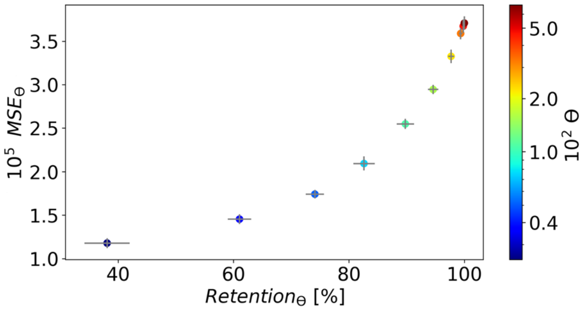


FIG. 5. Retention curve with the threshold (θ) set by $\sigma_{\max}$ to evaluate the bagging ensemble method. When a retention$_\theta$[%] of the predictions are accepted, MSE$_\theta$ indicates the VD performance on that specific prediction subset.

The retention curve is helpful for balancing the rejection/threshold trade-off for a specific use case and is comparable to a receiver operating characteristic (ROC) curve used for binary classification tasks [38]. In our case, we see a desirable positive correlation between MSE$_\theta$ and Retention$_\theta$. In deployment, the retention curve can be interpreted as follows: if the experimental conditions can tolerate shots with averaged MSE$_\theta \leq$ 2e-5, by choosing $\theta =$ 1.97e-2, we can retain 80% of the shots and yield a low MSE$_\theta$ of 1.85e-05.

Over the entire test set, the average MSE of bagging with 16 ensemble components was 3.65e-05 with an accuracy of 0.452 using $\pm 1\sigma$ and 0.706 using $\pm 2\sigma$ for computing α in Eq. (2). The average MSE for quantile regression method (using $\tau = 0.50$) was 4.05e-05. Accuracy was 0.973 using quantiles 0.05-0.95 and 0.648 using quantiles 0.25–0.75. Figure 6(a) shows a test shot and the predicted uncertainty from the bagging ensemble ($\pm 2\sigma$) and the 90% of the quantiles to represent epistemic and aleatoric uncertainty respectively [39].

In order to evaluate the robustness of the model we use it to predict an out-of-distribution (OOD) sample. This means that the test shot has a different probability distribution than the training dataset [40]. Therefore, it is expected that the model would manifest higher prediction uncertainty. In particle accelerators, shifts in the distribution could come, for example, from a different operation mode or slow drift over time. In those cases, it is critical to have reliable predictive uncertainty. As an example, we compare a prediction of a test shot with the same distribution as the training dataset [Fig. 6(a)], and a prediction for an out-of-distribution input from a different operation mode of the accelerator [Fig. 6(b)]. The spectrum inputs to the VD are shown in Fig. 6(c). It is evident that the $\pm 2\sigma$ interval from the ensembling method captures most of the uncertainty in the measured OOD shot.

This shows that the VD can identify out-of-distribution shots, wherein the quantiles and ensemble predict abnormally large uncertainty ranges, and these shots can be rejected. Previous works suggest that we can treat the predictive standard deviation from ensembles as epistemic uncertainty [20], and quantile regression as doing maximum likelihood with an asymmetric Laplace distribution, hence capturing aleatoric uncertainty [41]. Further development of this technique would allow users to know whether the predicted uncertainty comes from lack of useful data, or variation within that training data.

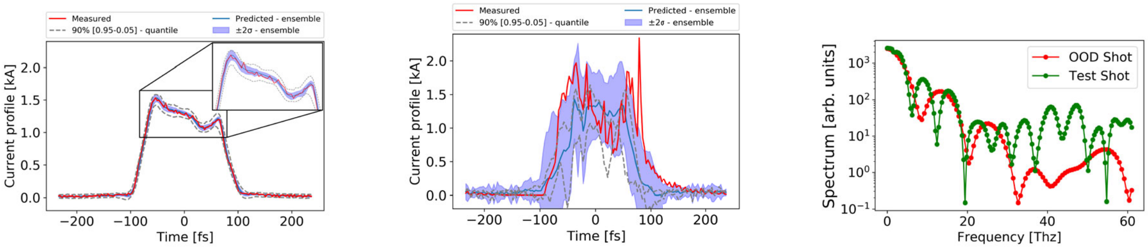
B. 2D LPS images

We used similar architecture to train ensembles with 2D longitudinal phase space images. The average MSE of the entire dataset is 6.714e-04, and the accuracy with $\pm 1\sigma$ is 0.291, and with $\pm 2\sigma$ is 0.538. The accuracy of the image predictions was calculated using Eq. (2) evaluated across an additional dimension:

$$\text{Accuracy} = \frac{\sum_{t,e=1}^{T,E} \alpha_{t,e} \cdot L^2_{\text{measured},t,e}}{\sum_{t,e=1}^{T,E} L^2_{\text{measured},t,e}} \qquad (3)$$

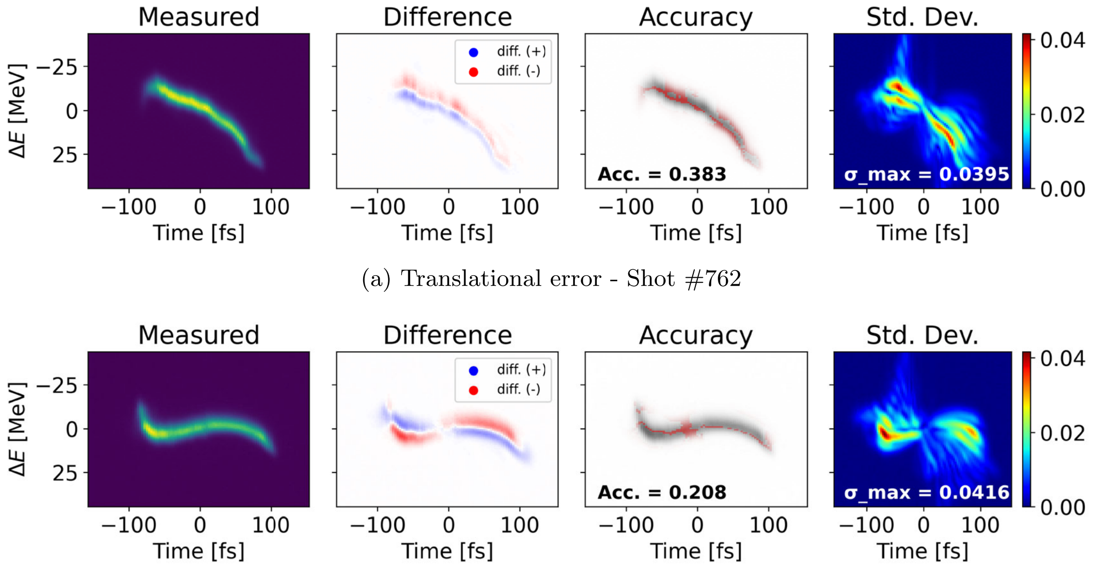
where $\alpha_{t,e} = \mathbb{1}(|L_{\text{measured},t,e}| < L_{\text{predicted},t,e} \pm 2\sigma_{t,e})s$.

Figure 7 shows two examples of shots in the test set. The panels from left to right show the measured LPS image, the difference between the measured and predicted, the accuracy metric [Eq. (2)], and the prediction uncertainty (σ). The blue and red colors on the difference panel indicates the positive and negative differences of the predicted LPS



(a) Test Shot (b) Out-of-distribution (c) Spectra

FIG. 6. Comparison of (a) test shot in the same distribution as the training set, and (b) out-of-distribution (OOD) test shot. The uncertainty intervals are predicted with an ensemble model of bagging with 16 components and 90% quantiles (dashed). The spectra of both shots are compared in (c).



(a) Translational error - Shot #762

(b) Shape error - Shot #789

FIG. 7. 2D longitudinal phase space (LPS) measurements with (a) translational error and (b) shape error examples.

respectively. The red mask on the accuracy panel indicates regions where the measured value falls within the predicted uncertainty.

The two examples show predictions with translational error (shot #762 in 7a) and shape error (shot #789 in 7b) with corresponding MSEs of 2.737e-4 and 1.096e-3. We classified a prediction as one with translational error when the positive and negative differences are shifted with respect to each other. For the translation error in shot #762, shifting the mean prediction up by a single pixel brings a 85.0% decrease in MSE and a 41.8% increase in accuracy. In order to determine how to shift the prediction, the "center of mass" of the prediction and the ground truth are calculated and then translated to match. In Sec. IV, we further discuss ways to leverage the spatial connectivity in these images to reduce such errors.

IV. DISCUSSION

In this section we discuss two techniques aiming to improve the LPS image neural network predictions. First we present the concept of a transposed convolution which can leverage the spatial connectivity in each data sample to produce better predictions. Then we present a specific "bottleneck" NN architecture that was applied and the insights derived from it.

A. Transposed convolutions

Transposed convolutions are a mathematical operation that allow for the upscaling of data. Filters are trained to learn features which are then projected on to a feature map larger than the input. Transposed convolutions can be especially useful in regression tasks where the output involves important features and connectivity [42]. The 2D outputs of the LPS image dataset are essentially images rather than vectors of unrelated measurements. Therefore, it makes sense to model architectures better fit for image generation which take advantage of the structured nature of the output. If we train filters to learn specific features in the dataset (tails, curves, bright spots, etc.), the NN model would be better equipped to handle the dataset at hand.

B. Bottleneck architecture

Bottleneck-shaped architecture is a NN design in which the number of neurons in the middle layer is smaller than in the other layers. The number of neurons per layer gradually decrease from the input to the middle layer, and then gradually increase from the middle layer to the output. The first part serves as an encoder or compressor which condenses essential information, and the second part serves as a decoder. The middle layer allows for the input to be recorded as an N dimensional vector. This embedding vector can be viewed as a representation of the instance in those N dimensions, with similar data points having a small distance between them, and distinct ones being very distant. Bottleneck architectures can reduce over-fitting by decreasing the system complexity [43].

The encoder was a dense NN with layers of size 256, 128, and an embedding dimension of $N = 64$. The decoder had transposed convolution layers of 8 2×2 filters, 8 4×4 filters, and 16 16×16 filters. Using this architecture, we trained a NN which achieved a negligibly improved average MSE of 8.833e-4. Nevertheless, a strength of the bottleneck architecture is its ability to condense information about the input and output relation into a single vector. In a high dimensional space, we expect that

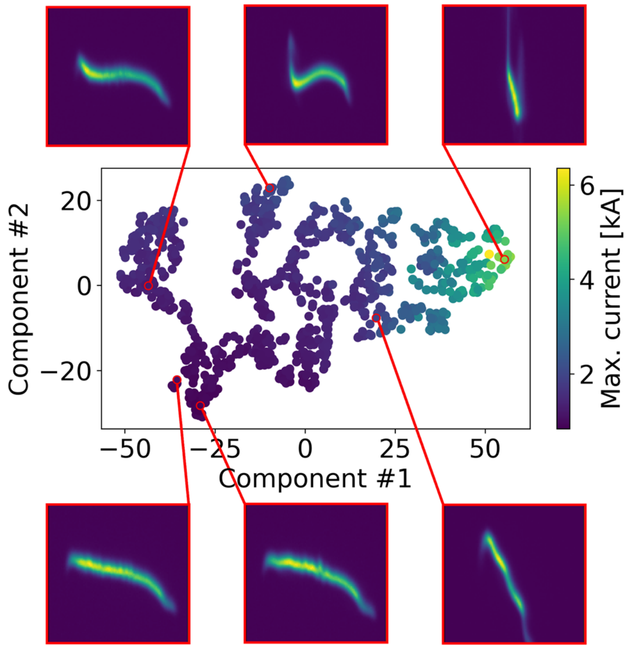


FIG. 8. t-SNE clustering of the compressed vectors of the bottleneck architecture for all test shots, with six LPS examples of the outputs shown on the top. Notably, shots are grouped by shape and maximum current in the latent space.

similar shots would aggregate close to each other, and distinctly different shots would be distant from one another.

In order to visualize this, we used the t-SNE clustering algorithm [44] on all the embedding vectors for shots in the test set. After investigating multiple parameter values, we used a perplexity of 30, a learning rate of 200, and observed that the algorithm converged after 1000 iterations. Figure 8 shows a visualization of the latent space of these vectors using two components. The maximum current for each shot is shown in color bar. The six examples shown illustrate that shots are grouped by shape and maximum current in the latent space.

V. CONCLUSIONS

In this paper we presented several deep learning approaches to incorporate uncertainty in the virtual diagnostic (VD) tool. Deep learning models such as neural networks have great predictive power, but they often suffer from over-fitting and provide over-confident predictions. Here, we compared three ensemble methods and quantile regression as a way to provide accurate mean predictions as well as correctly capturing confidence intervals. Specifically, we considered a VD trained on non-invasive spectral measurements of the electron beam to predict the 2D longitudinal phase space or a 1D current profile.

The UQ methods presented in the paper were shown to be robust against out-of-distribution inputs by providing un-confident predictions on data it was not trained with. In comparison to a simple dense neural network, it was shown that a more tailored architecture can be used to exploit information about the data and offer better explanation during inference. A principled approach to quantifying uncertainty is crucial for the deployment of the virtual diagnostic tool, especially for safety-critical systems as particle accelerators.

ACKNOWLEDGMENTS

This work was supported by the Department of Energy, Laboratory Directed Research and Development program at SLAC National Accelerator Laboratory, under Contract No. DE-AC02-76SF00515, the Alan Turing Institute AI Fellowship (UK), and the Engineering and Physical Sciences Research Council (UK) grant reference EP/L015897/1.

APPENDIX: REWEIGHTING THE CURRENT PROFILE DATASET

Upon further analysis of the dataset, it became clear that certain current profile features are under-represented. As a result, the prediction of profiles with higher peak current presents high MSE. Typically, using models trained on this unbalanced data would be reasonable assuming nominal usage of the machine. However, although we have fewer examples of high peak shots, we would like to train the model to give them equal importance as the lower peak shots. Therefore, we weighted the output (current profile) such that we gave instances with lower representation more weight during training, and instances with overrepresented samples less weight. Reweighting the data not only greatly reduced the MSE for many shots with high peak shots, but it also slightly reduced the overall MSE in other cases.

In order to determine the weight for each shot, we made a 100-bin histogram of $I_{\max}$ for all $S = \sum_{i=1}^{100} n_i$ shots in the training set. Denoting n_i as the number of shots in the ith bin, the weight of the shots in that bin is $w_i = c \cdot (n_i)^{-0.5}$ with a normalizing constant $c = S \cdot (\sum_{i=1}^{100} n_i^{0.5})^{-1}$. We tried other constants for the weighting power, but found that 0.5 as weighting constant performed the best. The overall MSE of the test set was 4.601e-5 for the unweighted NN, and 4.478e-5 for the reweighted NN (both averaged over 5 runs).

[1] P. Craievich *et al.*, Phys. Rev. Accel. Beams **23**, 112001 (2020).
[2] B. Marchetti *et al.*, Sci. Rep. **11**, 3560 (2021).
[3] S. C. Leemann, S. Liu, A. Hexemer, M. A. Marcus, C. N. Melton, H. Nishimura, and C. Sun, Phys. Rev. Lett. **123**, 194801 (2019).
[4] C. Tennant, A. Carpenter, T. Powers, A. S. Solopova, L. Vidyaratne, and K. Iftekharuddin, Phys. Rev. Accel. Beams **23**, 114601 (2020).

[5] E. Fol, R. Tomás, J. Coello de Portugal, and G. Franchetti, Phys. Rev. Accel. Beams **23**, 102805 (2020).

[6] P. Arpaia, G. Azzopardi, F. Blanc, G. Bregliozzi, X. Buffat, L. Coyle, E. Fol, F. Giordano, M. Giovannozzi, T. Pieloni, R. Prevete, S. Redaelli, B. Salvachua, B. Salvant, M. Schenk, M. S. Camillocci, R. Tomás, G. Valentino, F. Van der Veken, and J. Wenninger, Nucl. Instrum. Methods Phys., Res., Sect. A **985**, 164652 (2021).

[7] A. Scheinker, A. Edelen, D. Bohler, C. Emma, and A. Lutman, Phys. Rev. Lett. **121**, 044801 (2018).

[8] C. Emma, A. Edelen, M. J. Hogan, B. O'Shea, G. White, and V. Yakimenko, Phys. Rev. Accel. Beams **21**, 112802 (2018).

[9] A. Hanuka, C. Emma, T. Maxwell, A. Fisher, B. Jacobson, M. J. Hogan, and Z. Huang, Nat. Sci. Rep. Ochanomizu Univ. **11**, 2945 (2021).

[10] A. Sanchez-Gonzalez *et al.*, Nat. Commun. **8**, 15461 (2017).

[11] C. Emma, M. D. Alverson, A. Edelen, A. Hanuka, M. Hogan, B. O'Shea, D. Storey, G. White, and V. Yakimenko, in *Proceedings of the 8th International Beam Instrumentation Conference* (JACoW, Geneva, 2019), https://doi.org/10.18429/JACOW-IBIC2019-THBO01.

[12] D. Marx, R. Assmann, P. Craievich, U. Dorda, A. Grudiev, and B. Marchetti, Nucl. Instrum. Methods Phys. Res., Sect. A **909**, 374 (2018).

[13] A. Kendall and Y. Gal, What uncertainties do we need in bayesian deep learning for computer vision?, arXiv:1703.04977.

[14] C. E. Rasmussen, in *Summer School on Machine Learning* (Springer, New York, 2003), pp. 63–71.

[15] D. J. MacKay, Bayesian methods for adaptive models, Ph.D. thesis, California Institute of Technology, 1992.

[16] P. J. Bickel and D. A. Freedman, Ann. Statistics **9**, 1196 (1981).

[17] B. Clarke, J. Mach. Learn. Res. **4**, 683 (2003).

[18] L. Torossian, V. Picheny, R. Faivre, and A. Garivier, A review on quantile regression for stochastic computer experiments, arXiv:1901.07874.

[19] K. Sriram, R. Ramamoorthi, and P. Ghosh, Bayesian Analysis **8**, 479 (2013).

[20] B. Lakshminarayanan, A. Pritzel, and C. Blundell, in *Adv. Neural Inf. Process. Syst.* (NeurIPS, San Diego, CA, 2017), pp. 6402–6413.

[21] T. G. Dietterich, in *Multiple Classifier Systems* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2000) pp. 1–15.

[22] Y. Gal, Uncertainty in deep learning, Ph.D. thesis, University of Cambridge, 2016.

[23] L. Smith and Y. Gal, Conference on Uncertainty in Artificial Intelligence (2018).

[24] B. He, B. Lakshminarayanan, and Y. W. Teh, arXiv:2007.05864.

[25] B. Efron, *The Jackknife, the Bootstrap and other Resampling Plans* (SIAM, Philadelphia, PA, 1982).

[26] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap* (CRC Press, Boca Raton, FL, 1994).

[27] P. Emma *et al.*, Nat. Photonics **4**, 641 (2010).

[28] D. Marx, R. Assmann, P. Craievich, U. Dorda, A. Grudiev, and B. Marchetti, Nucl. Instrum. Methods Phys., Res., Sect. A **909**, 374 (2018).

[29] F. Chollet *et al.*, Keras, https://keras.io (2015).

[30] M. Abadi *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems, (2015), software available from tensorflow.org.

[31] Y. Gal and Z. Ghahramani, in *Proceedings of The 33rd International Conference on Machine Learning* (PMLR, New York City, NY, 2016), pp. 1050–1059.

[32] S. Lee, S. Purushwalkam, M. Cogswell, D. Crandall, and D. Batra, Why m heads are better than one: Training a diverse ensemble of deep networks, arXiv:1511.06314.

[33] S. Fort, H. Hu, and B. Lakshminarayanan, Deep ensembles: A loss landscape perspective, arXiv:1912.02757.

[34] K. He, X. Zhang, S. Ren, and J. Sun, in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15 (IEEE Computer Society, New York, 2015), p. 1026–1034.

[35] X. Glorot and Y. Bengio, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 9, edited by Y. W. Teh and M. Titterington (PMLR, Chia Laguna Resort, Sardinia, Italy, 2010), pp. 249–256.

[36] S. Raschka, Model evaluation, model selection, and algorithm selection in machine learning, arXiv:1811.12808.

[37] D. Opitz and R. Maclin, J. Artif. Intell. Res. **11**, 169–198 (1999).

[38] T. Gneiting and P. Vogel, Receiver operating characteristic (roc) curves, arXiv:1809.04808.

[39] N. Tagasovska and D. Lopez-Paz, in *Adv. Neural Inf. Process. Syst.* (NeurIPS, San Diego, CA, 2019).

[40] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek, Advances in neural information processing systems, arXiv:1906.02530.

[41] Tishby, E. Levin, and S. Solla, in *International 1989 Joint Conference on Neural Networks* (IEEE, Piscataway, NJ, 1989), pp. 403–409, Vol. 2.

[42] V. Dumoulin and F. Visin, A guide to convolution arithmetic for deep learning, arXiv:1603.07285.

[43] G. Perin, I. Buhan, and S. Picek, IACR Cryptol. ePrint Arch. **2020**, 58 (2020).

[44] S. Arora, W. Hu, and P. K. Kothari, in *Proceedings of the 31st Conference On Learning Theory*, Proceedings of Machine Learning Research, Vol. 75, edited by S. Bubeck, V. Perchet, and P. Rigollet (PMLR, Stockholm, Sweden, 2018), pp. 1455–1462.