

Full Shot Predictions for the DIII-D Tokamak via Deep Recurrent Networks

Ian Char

*Machine Learning Department
Carnegie Mellon University*

Youngseog Chung

*Machine Learning Department
Carnegie Mellon University*

Joseph Abbate

Princeton Plasma Physics Laboratory

Egemen Kolemen

*Department of Mechanical and Aerospace Engineering
Princeton Plasma Physics Laboratory
Princeton University*

Jeff Schneider

*Machine Learning Department
Robotics Institute
Carnegie Mellon University*

Abstract—Although tokamaks are one of the most promising devices for realizing nuclear fusion as an energy source, there are still key obstacles when it comes to understanding the dynamics of the plasma and controlling it. As such, it is crucial that high quality models are developed to assist in overcoming these obstacles. In this work, we take an entirely data driven approach to learn such a model. In particular, we use historical data from the DIII-D tokamak to train a deep recurrent network that is able to predict the full time evolution of plasma discharges (or “shots”). Following this, we investigate how different training and inference procedures affect the quality and calibration of the shot predictions.

I. INTRODUCTION

In a wide range of fields, dynamics modeling is a fundamental tool that can be used to gain better understanding of a given system. Dynamics models are especially useful in the context of control, as they allow for prediction of responses to system perturbations over time, which can then be used to design and implement control sequences that optimize for desirable behaviors.

Such benefits are especially apparent in tokamak systems. Tokamaks are toroidal devices which magnetically confine plasma at high temperatures and pressures for prolonged periods, during which nuclear fusion reactions occur within the plasma. The tokamak system is one of the most promising approach to realizing nuclear fusion as an energy source. While strides are being made to improve the efficiency, stability, and reliability of the system, there are crucial control challenges which remain [1].

Since running these devices is extremely expensive, domain experts rely on virtual representations of the system dynamics, such as simulators and dynamics models. Simulators typically rely on first principles and simulate the dynamics via known equations which describe the theoretical behavior of the plasma. However, simulators are prohibitively expensive in terms of time and computation, and despite these costs, they are often still unable to accurately describe the plasma’s dynamics.

Concurrently, massive strides have been made in machine learning (ML), where advances in algorithms and modeling

architectures paired with data and compute have allowed for a completely data-driven approach to learning highly accurate models. This approach is promising for the nuclear fusion setting, and indeed, numerous recent works have applied ML methods to tokamak modeling [2, 3, 4, 5, 6].

In this work, we focus on learning a dynamics model for the DIII-D tokamak, a tokamak in San Diego, California operated by General Atomics. Since the device has been in operation since 1986, we are able to draw from a wealth of previous plasma discharges (or “shots”) from the device to train a deep recurrent network. A typical shot on DIII-D lasts around 6-8 seconds, with a 1 second ramp up phase, several second flat top phase, and one second ramp down phase. DIII-D also has several real-time and post-shot diagnostics that measure the magnetic equilibrium and plasma parameters with high temporal resolution. We find that learned models are able to predict these measurements for entire shots remarkably well.

We further investigate the impacts of our modelling choices. Along with architecture and training choices, we highlight the importance of uncertainty quantification and explore which methods of forming predictive distributions results in the most calibrated models. As more interest accumulates in control of tokamaks via data-driven models [4, 5, 6, 7], we hope that this work provides valuable insights that accelerates prediction and control.

II. RELATED WORK

A. Simulators for Tokamaks

Predictive modeling of the plasma through first principle equations is difficult since different aspects of the plasma evolve at different time scales. State of the art simulators solve this problem by evolving these aspects independently [8]. While these simulators have been useful for exploring different regimes for the plasma [9] and making new controls [10], they are nevertheless limited in that they require additional external information, such as an estimate for the density at the edge of the plasma. Our learned models are unique from these in that the only information the models require are the settings for the different actuators throughout the shot.

There is a long lineage of methods for inferring behavior of a dynamic system from data. System identification is one broad categorization of such methods and can range from “white” to “black” based on how much prior domain knowledge is incorporated. Whereas white-box models rely strictly on prior knowledge of the relationships between variables to infer the system parameters, black-box methods rely on purely observed data to model their plausible relationships. We refer the reader to Ljung [11] and Schoukens and Ljung [12] for a more in-depth, comprehensive survey of existing methods in system identification.

Recently, neural networks (NN) have been widely used for modeling dynamics and have shown substantial success [13]. Many of these methods require at least some prior knowledge of the system (i.e. they are gray-box models). For example, one may be able to inject prior information into the model using a set of ODEs [14, 15] or PDEs [16, 17]. There has been recent exciting work applying these classes of models to tokamaks [18]; however, in our work, we explore black-box models in which there is no prior knowledge available.

C. Machine Learning in Nuclear Fusion

There has been a recent surge of interest in applying machine learning methods to predict the state of the plasma within tokamaks. One of the areas with the greatest interest is predicting whether the plasma is in (or is about to be in) an unstable state. Fu et al. [19], Parsons [20], Rea et al. [21], Boyer et al. [22], Seo et al. [7], Olofsson et al. [23], Keith et al. [24] learn predictive models of whether the plasma will become unstable and use these models to take preventative actions to stabilize the plasma. Char et al. [25] also use Bayesian optimization with a similar objective; however, they directly learn the actions to be applied rather than produce a prediction of whether the plasma is unstable or not.

In terms of learning the evolution of the plasma state, Abbate et al. [2] learn a deep neural network in order to predict the profiles of the plasma; however, they focus on one step predictions for their model. In contrast, Seo et al. [4, 5] learn a recurrent neural network to predict scalar states of the plasma. They can use this model to autoregressively predict these states into the future, and they leverage this to plan shots on the KSTAR tokamak. Recently, Char et al. [6] used a (non-recurrent) learned model that predicted both scalar and profile states in an autoregressive manner. They used this model as a simulator to train a reinforcement learning agent, which was then deployed on DIII-D. Whereas most of these works focus on the control aspect of dynamics modelling, we do a deeper investigation on the modeling task itself. We hope our exposition on the modeling choices, evaluation techniques, and ablation studies will provide useful insights for future research in dynamics modeling and model-based control for tokamaks.

A. Data

We begin this section by describing the data used to train our dynamics model. In total, we use 7,884 historical shots from the DIII-D tokamak. We include both the ramp up and flat top phases of each shot, and each shot is subdivided into a number of “time steps” 25ms apart from each other. For each time step, we average the measurements collected 25ms previous to that point in time.

We partition the input signals into two groups: state signals and actuators signals. All of these signals can be found in Table I. For the state signals, we use 17 different *scalar* states and 6 so-called *profile* states. While scalar states provide a summary statistic of one aspect of the current plasma state with a single scalar, profile states are 1D measurements of the plasma, and in our dataset, they consist of 33 discrete measurements along the minor radius of the cross-section of the tokamak. Following previous works in profile modeling [6, 3], we choose to lower the dimensionality of the (originally 33-dimensional) profile states via Principle Component Analysis (PCA). In particular, we use four principal components to represent all profiles states except for the pressure and q profile. For these two signals, we use the first two principal components to represent the profile. For actuator signals, we use 21 different scalar values that summarize neutral beam settings, current and density targets, gas settings, and plasma shape control.

We choose to separate these signals into two groups (states and actuators) since we assume that all of the actuators are known a priori (from the perspective of the experiment operator). As such, the model takes as input the current state measurements, the current actuators settings, and the actuator settings 25ms into the future. The model then predicts the change in the state variables for the next 25ms. Once trained, the model can predict many more steps into the future by autoregressively feeding in predicted next states back into the model as inputs.

B. Model Architecture and Training

In designing our model architecture, we use a recurrent neural network (RNN) with a gated recurrent unit (GRU) [26]. We use 6 hidden layers (including encoder and decoder), each with 512 hidden units and residual connections [27]. A visual diagram of the model architecture can be seen in Figure 1. We train our recurrent model with full length shots, the longest of which is 225 time steps. We use a learning rate of $3e-4$ and a weight decay of 0.001.

For the model output, rather than making point predictions, we have two output heads, where each head predicts the mean and log variance of a Gaussian distribution, respectively. The negative log likelihood (NLL) is computed with this Gaussian prediction, and the model is trained to optimize the NLL loss. This method of predicting the parameters of a Gaussian distribution via the outputs of a neural network is also known as a mean-variance network or a probabilistic neural network

Group	Representation	Type	Signal	Dimension
States	Scalar	Shape	κ , a_{minor} , Triangularity Top, Triangularity Bottom, R and Z Coordinates of Magnetic Axis	6
		Other	β_N , Line Averaged Density, Internal Inductance (li), q_0 , q_{95} , $n1_{\text{rms}}$, $n2_{\text{rms}}$, $n3_{\text{rms}}$, v_{loop} , w_{mhd} , Differential Rotation [6]	11
	Profile	Electron Temperature, Ion Temperature, Density, Rotation, Pressure, q		20
Actuators	Scalar	Beam	Power Injected, Torque Injected	2
		Gas	gasA, gasB, gasC, gasD	4
		Shape	12 Shape Controls	12
		Other	Current Target, Density Target, Toroidal Field	3
	Total Dimension: 58			

TABLE I
LIST OF ALL STATE AND ACTUATOR VARIABLES

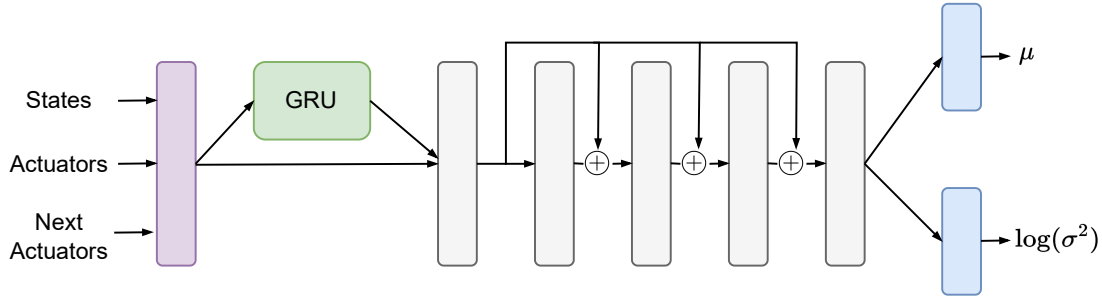


Fig. 1. **Architecture for the Recurrent Model.** The **encoder** is a single layer MLP which embeds the states, actuators, and next actuators into a 512 dimensional space. This is fed to the **GRU unit** which outputs a 128 dimensional embedding which is concatenated with the original embedding before being fed to the decoder. The **double headed outputs** are single linear layers outputting the mean and log variance of a Gaussian. Note the pluses with circles denote a residual connection.

(PNN) [28, 29], and is one of the most widely used methods of modeling predictive uncertainty. We extend our discussion on modeling uncertainty in Section IV-A.

Following Chua et al. [30], we found it essential to place a soft bound on the log variance using a learned lower and upper bound to ensure stability during training. The difference between the upper and lower bound is then added as an additional penalty term to the loss function, encouraging the width of the bounds to be as small as possible.

With the full dataset of shots available, we dedicate 90% of shots for training, 5% for validation, and 5% for testing. The shots are sorted chronologically before the splits are made, and we ensure that the testing shots consist of the 5% most recent shots. This is essential for testing since experiments (shots) run on the same day tend to be similar. Further, the tokamak is upgraded over time, which alter the dynamics of the plasma. Hence, enforcing the chronological order not only allows us to test the generalization of the learned dynamics model, but

also reflects the realistic test setting that a practitioner will be faced with.

C. Evaluation

To start, we visualize a “replay” of a shot from the test set. That is, we predict the full shot using only the initial state of the plasma and the sequence of actuators. We show the results of this in Figure 2, where we find that the model is able to predict the trend across time for the majority of the plasma’s states remarkably well.

To quantitatively evaluate the model’s accuracy, we use *Explained Variance* (EV) as an interpretable metric. This metric is used for the 1D regression setting where, given ground-truth label $y \in \mathbb{R}$ and prediction $\hat{y} \in \mathbb{R}$, the metric is defined as

$$\text{EV} := 1 - \frac{\text{Var}(y - \hat{y})}{\text{Var}(y)} \quad (1)$$

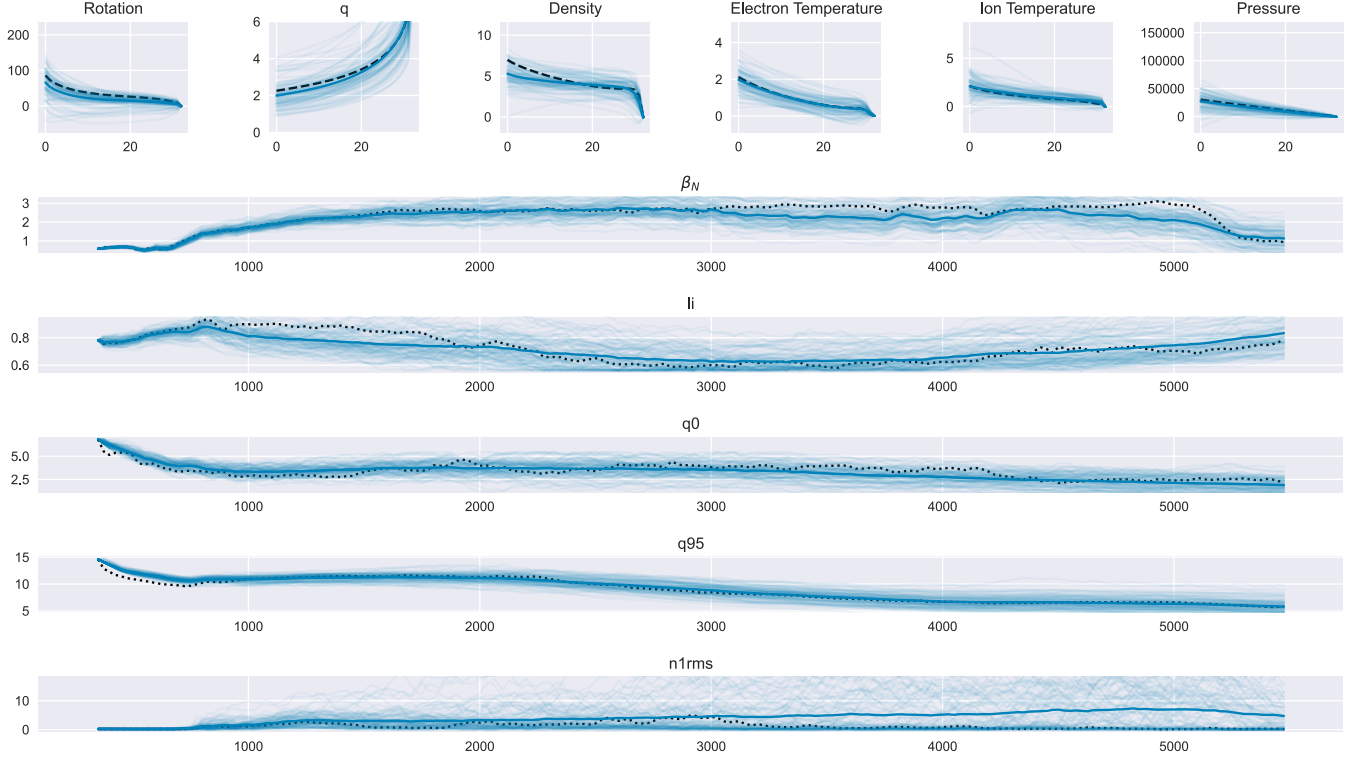


Fig. 2. **Replay of a Test Set Shot** The replay was generated with an ensemble of models which sample from their respective Gaussian distribution at each step. While the model has access to the true actuators throughout the entire shot, it only takes in the first true state and autoregressively predicts the rest. The faded blue lines show one sampled trajectory, while the darker blue line shows the average over the trajectories. The black lines show the true values for the experiment. The top row shows the reconstructed profiles at the last time step. Here, the x-axis is over the minor radius of the tokamak, where 0 is the closest to the magnetic axis and 33 is closest to the wall. The other plots show the scalar values over time. The x-axis shows the time into the shot in ms.

Here, $\text{Var}(y - \hat{y})$ and $\text{Var}(y)$ are the empirical variance of the residuals and labels, respectively. Intuitively, this metric shows how much of the variability in the dataset the model can explain, with the maximum (best) score being 1. Since the plasma's state is multi-dimensional, we compute EV for each of the dimensions and for each time step into the future.

We choose to compute the EV for the difference in the plasma's state at some time step t and the plasma's initial state. With respect to Equation 1, we set $y = s_t - s_0$ and $\hat{y} = \hat{s}_t - s_0$, where s_t is a single dimension of interest in the plasma state at time step t , \hat{s}_t is the model's prediction at time step t , and s_0 is the initial state. We believe that this choice in label better aligns with the task of predicting the evolution of the plasma. Indeed, we found if we measure EV of the plasma state (i.e. we do not subtract s_0), EV is nearly perfect for small t since the plasma does not evolve drastically from time step to time step.

For each of the test shots, we evaluate the model by starting prediction at different time steps during the shot (but always seeing the history up to that point), and then autoregressively predicting the remainder of the shot. We assemble the dataset to compute EV using every possible starting time step in all testing shots. We also compare the EV when sampling from the learned distribution versus using only the mean of the Gaussian. For this sampling method, we sample 30 different

trajectories and take the mean over them before computing the EV.

Figure 3 displays the EV over time for a select set of scalar quantities, the first component of the profiles, and averaged across all input dimensions. We chose these scalar signals because of their significance. In particular, β_N is the normalized ratio between plasma and magnetic pressure and can be used as an indicator of economic performance; q_0 and q_{95} are two points along the q (or safety factor) profile, which is an important indicator of stability of the plasma; and n1rms is the root mean squared of magnetic fluctuations for toroidal mode number $n = 1$, which can signify an event such as a tearing mode in the plasma. We note that we expect n1rms in particular to be hard to predict.

For each of these plots, EV starts low and grows over time, which can be explained by a number of factors. First, there is noise in the system which makes it difficult to predict on the 25ms time scale. However, we hypothesize that the signal starts dominating over the noise after a handful of time steps. In addition, as the number of time steps starts to grow the variance of $s_t - s_0$ continues to grow (i.e. the denominator of Equation 1), making it easier to achieve higher EV. In terms of using the mean of the Gaussian versus sampling from it, it seems sampling helps with short term predictions; however, for long term predictions, it appears that using the mean can

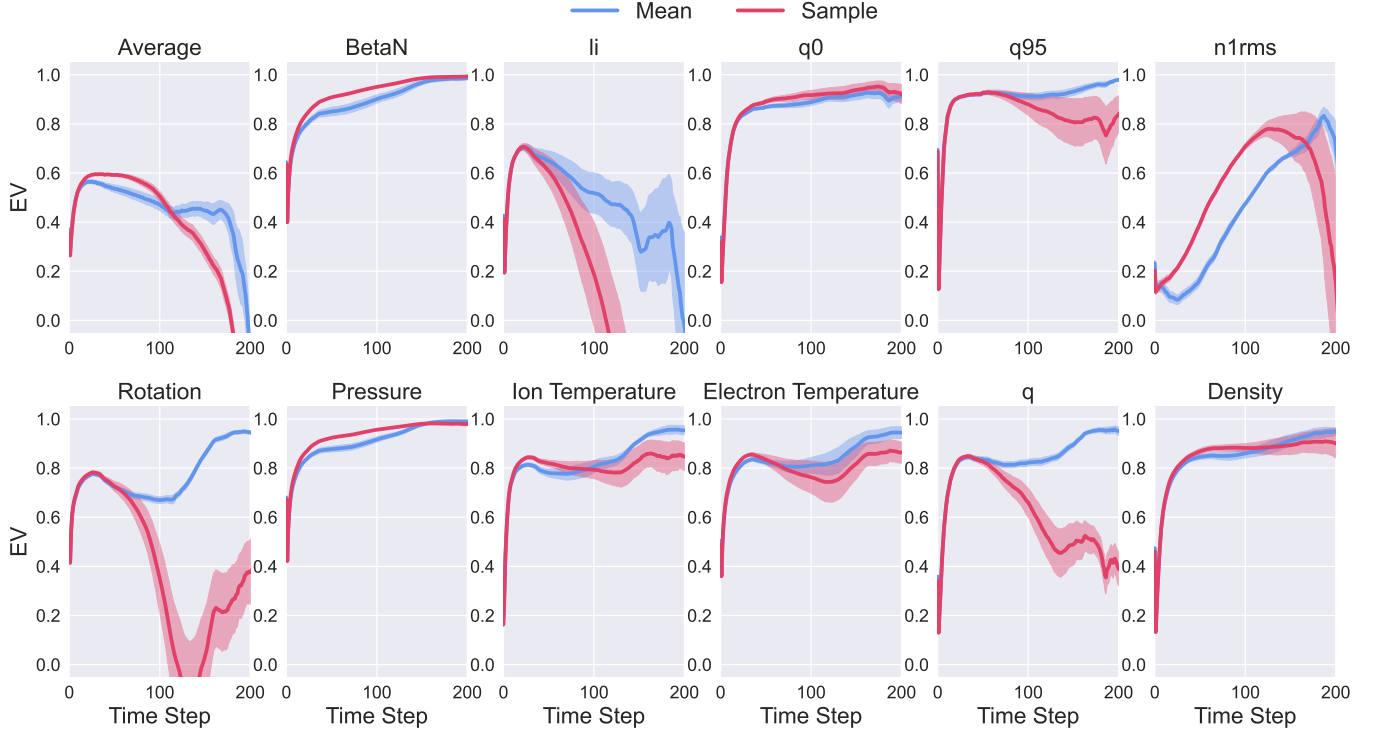


Fig. 3. **Explained Variance per Time Step.** Each of the colored lines show a different way of generating trajectories with the same models. The blue lines simply take the mean of the Gaussian distribution while the red line samples from the Gaussian distribution at every step. Each curve shows the mean over four different models with different random seeds. The shaded area shows the standard error. In the bottom row, we show the EV for the first principle component of the corresponding profiles.

be more reliable.

IV. ABLATIONS

In this section, we examine a number of choices that we have made for modeling and inference procedures. Our objective is to provide valuable insights that can assist other practitioners in the field when developing dynamic models of plasma evolution in tokamak devices.

A. Uncertainty Quantification

Uncertainty quantification is a crucial aspect in any modeling or prediction task, especially in the face of system stochasticity or insufficient data. Adequately modeling uncertainty has been shown to be especially critical for dynamics models when they are leveraged for control [30].

In our modeling efforts, we account for uncertainty by producing predictive distributions instead of point predictions, and we do so by relying on two different methods: by predicting a Gaussian distribution and by ensembling predictions. These two methods were utilized by Chua et al. [30] to capture the aleatoric uncertainty (the uncertainty inherent in the system) and the epistemic uncertainty (the uncertainty stemming from insufficient data), respectively. The ensemble consists of 4 models, each of which have the same model architecture as described in Section III-B, but each model was randomly initialized and trained independently [29] on the same dataset.

Inspired by Chua et al. [30], we test three methods of generating predictive distributions from our ensemble of networks, each of which predict Gaussian distributions. In the first method, which we denote as “Mean-TS1”, we take the mean prediction of the Gaussian distribution, but sample a new model from the ensemble to generate the next state every step. In the other two methods, we sample from the Gaussian distribution, and we either choose to sample a model from the ensemble every step or every shot. We denote these two methods as “Sample-TS1” and “Sample-TSInf”, respectively. Because of ensembling and the auto-regressive nature of the model, we do not have a closed form predicted distribution for the plasma’s state. Instead, we approximate this distribution with independent Gaussians for each dimension of the plasma’s state. The mean and standard deviations are estimated from 30 samples from the dynamics model.

To evaluate the predictive uncertainties, we measure the *Coverage* of a 90% prediction interval (PI) and the so-called *Miscalibration Area*. At a high level, Coverage is the empirical frequency of observations that fall within a constructed PI. Ideally, if the PI is constructed to capture $1 - \alpha$ probability mass, $(1 - \alpha) \times 100\%$ of the data should fall within this interval. Concretely, given data points $y_n \in \mathbb{R}$ and $(1 - \alpha)$ intervals $PI_{n,(1-\alpha)}$ for $n = 1, \dots, N$, the $(1 - \alpha)$ coverage is defined

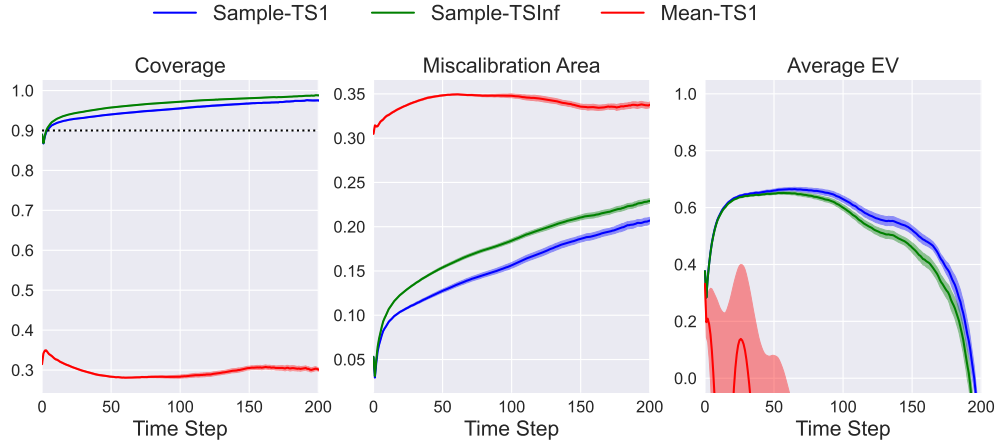


Fig. 4. **Uncertainty Metrics over Time.** The leftmost plot shows coverage of the 90% prediction interval. Models with good predictive uncertainties should therefore match this 90% (shown as dotted black line). For the miscalibration area plot, the lower the score, the better calibrated the model is. Each of the metrics is averaged over all output dimensions. Moreover the curves show the mean over four models, and the shaded region shows the standard error.

by

$$\text{Coverage}_{(1-\alpha)} = \frac{1}{N} \sum_{n=1}^N \mathbb{1}\{y_n \in \text{PI}_{n,(1-\alpha)}\}.$$

Building on this, the deviation from the expected probability is regarded as miscalibration, and *Miscalibration Area* takes the average deviation over a set of expected probabilities. Given a set of M expected probabilities drawn uniformly from $[0, 1]$: $p_i \sim U[0, 1], i \in [M]$, and the observed Coverage_{p_i} , *Miscalibration Area* is calculated as

$$\frac{1}{M} \sum_{i \in [M]} |p_i - \text{Coverage}_{p_i}|.$$

Much like EV, these metrics are for single dimensional spaces. As such, we compute these metrics for each of the dimensions of the state space and average the results together to produce a single metric. We also compute both metrics for each time step into the future.

Figure 4 shows the Coverage, Miscalibration Area (computed with the ‘‘Uncertainty Toolbox’’ [31]), and EV for the three methods of generating predictive distributions. We see that in the ensemble regime, purely taking the mean of the distribution is detrimental. Not only does the EV suffer, but we also observe extreme overconfidence as shown by the low Coverage (‘‘Mean-TS1’’ method in Figure 4).

Looking at the other two methods, we see that both methods are very well-calibrated at the beginning in their short-term predictions. Moreover, there is an improvement in average EV over the single model case (displayed in the top left plot of Figure 3), indicating the significance of utilizing an ensemble approach beyond simply quantifying uncertainty. We find that in all aspects, Sample-TS1 dominates over every other method. This aligns with the suggestion given by Chua et al. [30].

As the prediction horizon increases, Miscalibration Area steadily increases, and as evidenced by the Coverage plot, the predictive distributions become increasingly under-confident

Modelling Choice	Scaled One-Step MSE
MLP + Gaussian	1.20
LSTM + Gaussian	1.05
GRU + Point Prediction	1.06
GRU + Gaussian	1.0

TABLE II
ONE-STEP MSE FOR MODEL CHOICES WE SCALE THE MSE BY DIVIDING EACH SCORE BY THE BEST MSE THAT APPEARING IN THE TABLE. THAT IS, 1.0 IS THE BEST SCORE, WHILE A SCORE OF 1.20 MEANS THAT THE MSE ACHIEVED WAS 20% WORSE THAN THE BEST MSE. EACH SCORE IS THE MEAN OVER FOUR DIFFERENT SEEDS.

(i.e. higher than expected coverage). In many cases, one can apply recalibration [32] methods to adjust the calibration; however, we are unaware of any such method for the autoregressive setting.

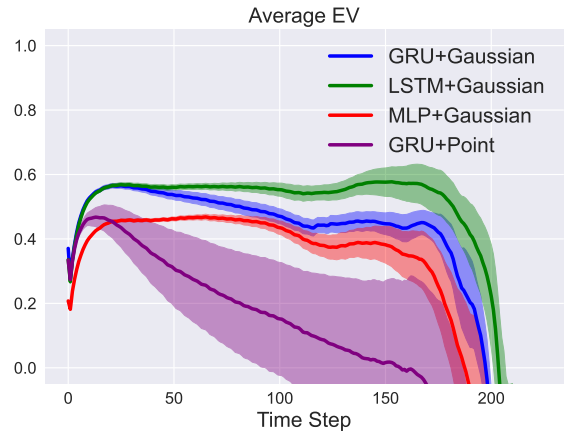


Fig. 5. **Explained Variance Averaged over All Output Dimensions.** Each curve was generated by taking the average over four different trained models. The shaded area shows the standard error. All curves were generated by taking the mean output of the predicted Gaussian distributions (where applicable).

B. Recurrent Unit

Next, we consider the impact of the recurrent unit chosen. We consider two alternatives besides the GRU component discussed in Section III: a model with no recurrent unit at all (e.g. an MLP) and a model with an LSTM [33] unit. From Table II, one can see that the GRU is superior in terms of single step MSE. Indeed, overall we see that GRU seems to be a good choice when looking at shorter horizon predictions. However, from Figure 5 it appears that LSTMs are better when considering a longer time horizon. Therefore, one may want to decide on the best recurrent unit based on the downstream application of the dynamics model. In either case, we see that recurrent units are essential since a standard MLP struggles both with one-step MSE and EV.

C. Point vs Distributional Estimate

Lastly, we look at the effect of having the model learn a distribution, as proposed in Section III, as opposed to having the model output a point prediction and training with MSE loss. Interestingly, even though the models that output a point prediction are trained on MSE, they achieve worse MSE on the test set according to Table II. We hypothesize this is because learning a Gaussian distribution prevents the network from overfitting on the training data. Indeed, we observe that on the training set, models with point predictions achieve roughly 20% lower MSE when compared with those that output Gaussian distributions. On top of this, Figure 5 shows that while models with point predictions have decent EV at first, as one predicts further into the future they achieve worse performance than even models with no recurrent units.

V. DISCUSSION

In this work, we show that deep recurrent models are a powerful tool that can be used for full shot predictions in tokamak devices. We emphasize that these models were simply given the initial state and the actuators to be applied during the duration of the shot. We encourage the fusion community to leverage data driven models when designing controllers and exploring actuator choices, and we hope that insights shown in this work will prove useful in those pursuits.

ACKNOWLEDGEMENT

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, using the DIII-D National Fusion Facility, a DOE Office of Science user facility, under Awards DE-AC02-09CH1146 and DE-FC02-04ER54698. This work was also supported by DE-SC0021414 and DE- SC0021275 (Machine Learning for Real-time Fusion Plasma Behavior Prediction and Manipulation). Additionally, this work is supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE1745016 and DGE2140739. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science

Foundation. Lastly, Youngseog Chung is supported by the Kwanjeong Educational Foundation.

Disclaimer This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

REFERENCES

- [1] D. Humphreys, G. Ambrosino, P. de Vries, F. Felici, S. H. Kim, G. Jackson, A. Kallenbach, E. Kolemen, J. Lister, D. Moreau *et al.*, “Novel aspects of plasma control in iter,” *Physics of Plasmas*, vol. 22, no. 2, p. 021806, 2015.
- [2] J. Abbate, R. Conlin, and E. Kolemen, “Data-driven profile prediction for diii-d,” *Nuclear Fusion*, vol. 61, no. 4, p. 046027, 2021.
- [3] M. Boyer, J. Wai, M. Clement, E. Kolemen, I. Char, Y. Chung, W. Neiswanger, and J. Schneider, “Machine learning for tokamak scenario optimization: combining accelerating physics models and empirical models,” in *APS Division of Plasma Physics Meeting Abstracts*, vol. 2021, 2021, pp. PP11–164.
- [4] J. Seo, Y.-S. Na, B. Kim, C. Lee, M. Park, S. Park, and Y. Lee, “Feedforward beta control in the kstar tokamak by deep reinforcement learning,” *Nuclear Fusion*, vol. 61, no. 10, p. 106010, 2021.
- [5] —, “Development of an operation trajectory design algorithm for control of multiple 0d parameters using deep reinforcement learning in kstar,” *Nuclear Fusion*, vol. 62, no. 8, p. 086049, 2022.
- [6] I. Char, J. Abbate, L. Bardóczi, M. Boyer, Y. Chung, R. Conlin, K. Erickson, V. Mehta, N. Richner, E. Kolemen *et al.*, “Offline model-based reinforcement learning for tokamak control,” in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 1357–1372.
- [7] J. Seo, S. Kim, A. Jalalvand, R. Conlin, A. Rothstein, J. Abbate, K. Erickson, J. Wai, R. Shousha, and E. Kolemen, “Avoiding fusion plasma tearing instability with deep reinforcement learning,” *Nature*, vol. 626, no. 8000, pp. 746–751, 2024.
- [8] F. Felici, O. Sauter, S. Coda, B. Duval, T. Goodman, J. Moret, J. Paley, T. Team *et al.*, “Real-time physics-model-based simulation of the current density profile in tokamak plasmas,” *Nuclear Fusion*, vol. 51, no. 8, p. 083052, 2011.

- [9] P. Rodriguez-Fernandez, A. Creely, M. Greenwald, D. Brunner, S. Ballinger, C. Chrobak, D. Garnier, R. Granetz, Z. Hartwig, N. Howard *et al.*, “Overview of the sparc physics basis towards the exploration of burning-plasma regimes in high-field, compact tokamaks,” *Nuclear Fusion*, vol. 62, no. 4, p. 042003, 2022.
- [10] F. Felici and O. Sauter, “Non-linear model-based optimization of actuator trajectories for tokamak plasma profile control,” *Plasma Physics and Controlled Fusion*, vol. 54, no. 2, p. 025002, 2012.
- [11] L. Ljung, “Perspectives on system identification,” *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, 2010.
- [12] J. Schoukens and L. Ljung, “Nonlinear system identification: A user-oriented road map,” *IEEE Control Systems Magazine*, vol. 39, no. 6, pp. 28–99, 2019.
- [13] R. Wang and R. Yu, “Physics-guided deep learning for dynamical systems: A survey,” *arXiv preprint arXiv:2107.01272*, 2021.
- [14] V. Mehta, I. Char, W. Neiswanger, Y. Chung, A. Nelson, M. Boyer, E. Kolemen, and J. Schneider, “Neural dynamical systems: Balancing structure and flexibility in physical prediction,” in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 3735–3742.
- [15] Y. Yin, V. Le Guen, J. Dona, E. de Bézenac, I. Ayed, N. Thome, and P. Gallinari, “Augmenting physical models with deep networks for complex dynamics forecasting,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2021, no. 12, p. 124012, 2021.
- [16] M. Raissi and G. E. Karniadakis, “Hidden physics models: Machine learning of nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 357, pp. 125–141, 2018.
- [17] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.
- [18] A. M. Wang, D. T. Garnier, and C. Rea, “Hybridizing physics and neural odes for predicting plasma inductance dynamics in tokamak fusion reactors,” *arXiv preprint arXiv:2310.20079*, 2023.
- [19] Y. Fu, D. Eldon, K. Erickson, K. Kleijwegt, L. Lupin-Jimenez, M. D. Boyer, N. Eidietis, N. Barbour, O. Izacard, and E. Kolemen, “Machine learning control for disruption and tearing mode avoidance,” *Physics of Plasmas*, vol. 27, no. 2, p. 022501, 2020.
- [20] M. S. Parsons, “Interpretation of machine-learning-based disruption models for plasma control,” *Plasma Physics and Controlled Fusion*, vol. 59, no. 8, p. 085001, 2017.
- [21] C. Rea, K. Montes, K. Erickson, R. Granetz, and R. Tinguely, “A real-time machine learning-based disruption predictor in diii-d,” *Nuclear Fusion*, vol. 59, no. 9, p. 096016, 2019.
- [22] M. Boyer, C. Rea, and M. Clement, “Toward active disruption avoidance via real-time estimation of the safe operating region and disruption proximity in tokamaks,” *Nuclear Fusion*, vol. 62, no. 2, p. 026005, 2021.
- [23] K. Olofsson, D. Humphreys, and R. La Haye, “Event hazard function learning and survival analysis for tearing mode onset characterization,” *Plasma Physics and Controlled Fusion*, vol. 60, no. 8, p. 084002, 2018.
- [24] Z. Keith, C. Nagpal, C. Rea, and R. A. Tinguely, “Risk-aware framework development for disruption prediction: Alcator c-mod and diii-d survival analysis,” 2024.
- [25] I. Char, Y. Chung, W. Neiswanger, K. Kandasamy, A. O. Nelson, M. Boyer, E. Kolemen, and J. Schneider, “Offline contextual bayesian optimization,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [26] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.
- [27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [28] D. A. Nix and A. S. Weigend, “Estimating the mean and variance of the target probability distribution,” in *Proceedings of 1994 ieee international conference on neural networks (ICNN’94)*, vol. 1. IEEE, 1994, pp. 55–60.
- [29] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” *Advances in neural information processing systems*, vol. 30, 2017.
- [30] K. Chua, R. Calandra, R. McAllister, and S. Levine, “Deep reinforcement learning in a handful of trials using probabilistic dynamics models,” *Advances in neural information processing systems*, vol. 31, 2018.
- [31] Y. Chung, I. Char, H. Guo, J. Schneider, and W. Neiswanger, “Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification,” *arXiv preprint arXiv:2109.10254*, 2021.
- [32] V. Kuleshov, N. Fenner, and S. Ermon, “Accurate uncertainties for deep learning using calibrated regression,” in *International conference on machine learning*. PMLR, 2018, pp. 2796–2804.
- [33] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.