Predicting the Temporal Evolution of Tokamak Plasma Profiles with Deep Learning

Jalal Butt[1] and Egemen Kolemen[2,3]

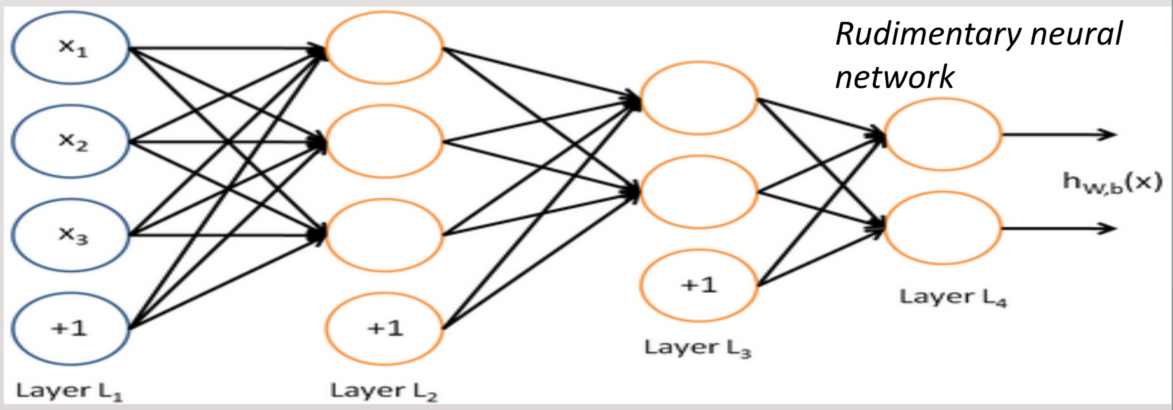
[1]Central Connecticut State University, [2]Princeton University, [3]Princeton Plasma Physics Laboratory

Motivation

- *Fusion goal*: Make nuclear fusion a commercially feasible energy source
- Major challenge: plasma stability
  - if we can predict plasma stability → manipulate plasma to avoid instabilities/dangerous modes
  - Tokamak plasma profiles are inputs to plasma stability codes
  - **Can AI help us predict plasma [profile] evolution?**
- Plasma evolution knowledge → predict potential instability onsets

Deep Learning with Neural Networks

- Inspired by biological neural networks
- Interdisciplinary applications (including fusion, particularly disruption predictions)
- Supervised learning problem (target profiles known)



- 1.a.) Weights (*W*) and biases (*b*) for the model's raw neuron activations (A^{i+1}) are defined as:

$$\begin{bmatrix} A_1^{i+1} \\ A_2^{i+1} \\ \vdots \\ A_n^{i+1}\end{bmatrix} = \begin{bmatrix} W_1^1 & W_2^1 & \cdots & W_{np}^1 \\ W_1^2 & \ddots & & \vdots \\ \vdots & & & \\ W_1^n & \cdots & & W_{np}^n \end{bmatrix} \cdot \begin{bmatrix} A_1^i \\ A_2^i \\ \vdots \\ A_{np}^i \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{np}\end{bmatrix}$$

Tensors are extended by one rank for multi-layer NN

- 1.b.) Raw neuron acts. are linearly activated:

$$ReLU(A_j^{i+1}) = max(0, A_j^{i+1}) = x^+$$

- 2.) A mean-absolute difference loss [error] function is defined for loss minimization:

$$loss(\hat{y}_i) = \sum_{1}^{batchsize} \left| \frac{(y_i - \hat{y}_i)}{batchsize} \right|$$

$\hat{y}_i$: parameterized by $\vec{W}$ and $\vec{b}$

y_i is the sum of the target values vector and $\hat{y}_i$ is the sum of the predicted values vector

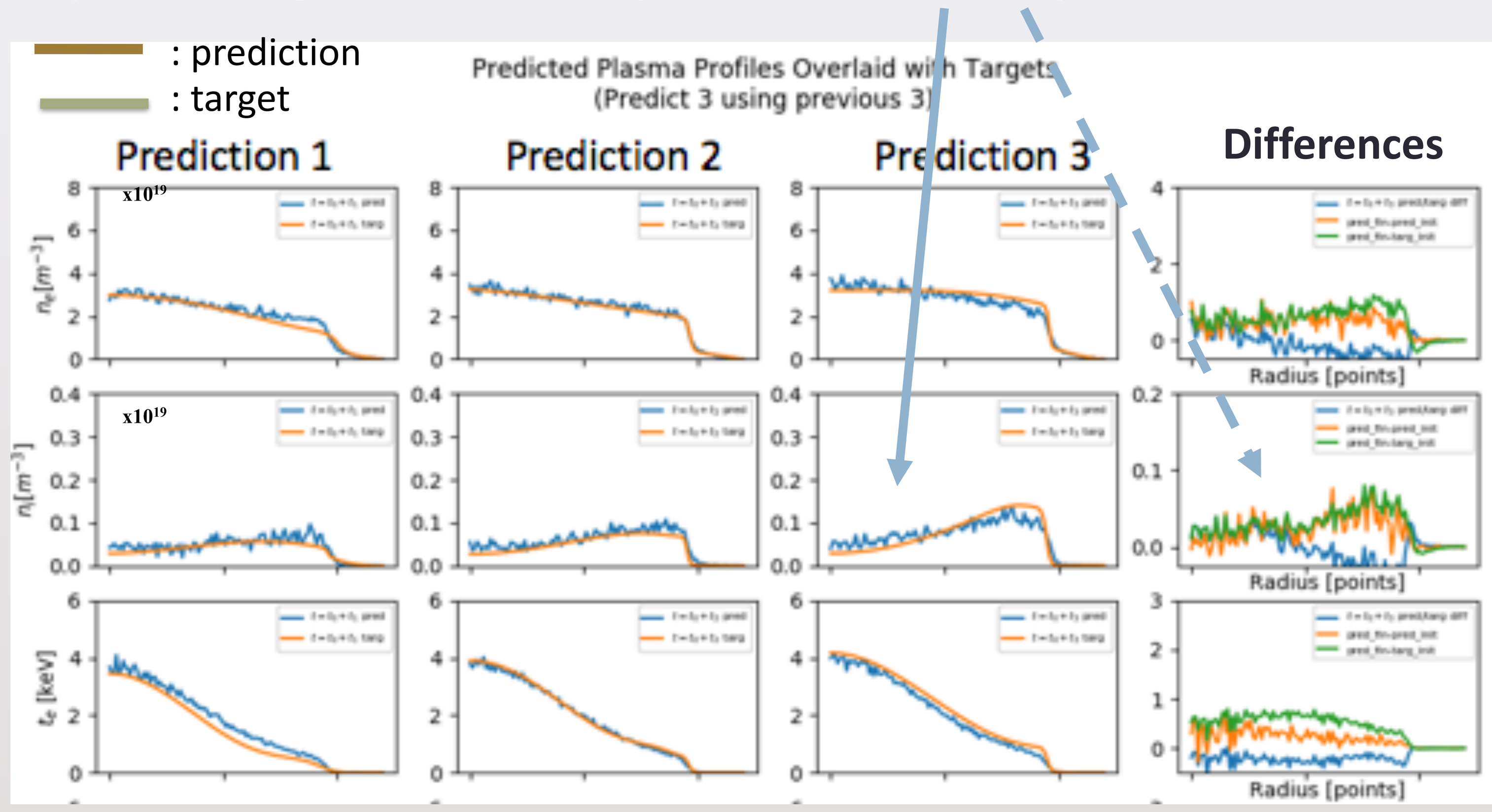
- 3.) Learning Process: Minimize *loss*($\hat{y}_i$) by updating weights and biases (θ) using SGD algorithm; 4.) minimization optimized by Adam optimizer

$$\theta^i \leftarrow \theta^{i-1} + \frac{\gamma^i}{batch} \sum_{n \in batch} (\nabla_\theta f_n(\theta)|_{\theta=\theta^{(i-1)}})$$

θ^i: updated weights and biases γ^i: learning-rate schedule;

Preliminary Plasma Profile Predictions via RNN

- Model goal: predict physically significant spatio-temporal changes in profiles with knowledge of previous plasma state (via profiles) and current 0D inputs
- **Examples from preliminary results show model predicting onset of ion profile "hollowing"**



Relevant information

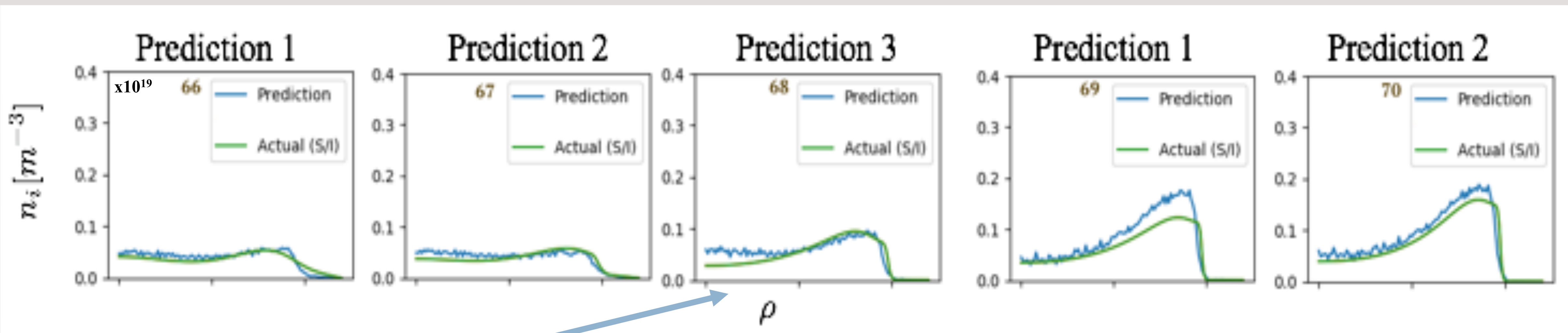
- **R**ecurrent **N**eural **N**etwork Model:
  - LSTM Units: 200; LSTM Cells [layers]: 4
    - *Limited* by comp. power
- 588 [DIII-D] shots
- 50-ms time-slices
- Built using Google's TensorFlow API

Model Inputs:

- Previous 150-ms of n_e, n_i, T_e, T_i, Rot. profiles (zipfit - OMFITmdsplus)
- 0D time-traces (previous and concurrent)
  - NBI power/torque, gas A/B flowrates, ECH power

Model Outputs:

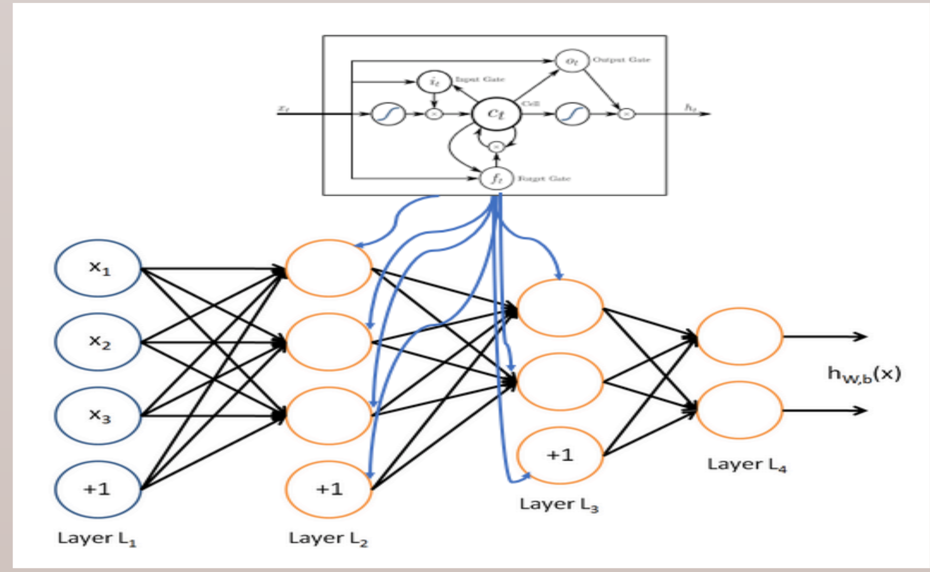
- Future (unseen) 150-ms of plasma profiles



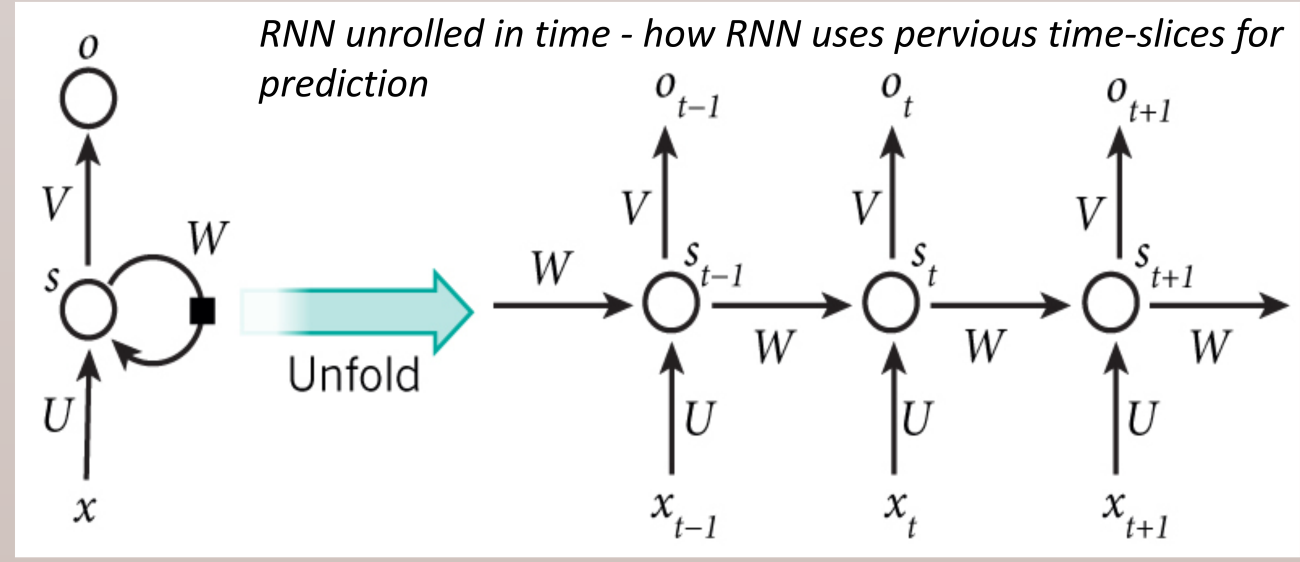
Another example of ion-hollowing onset prediction

Background - Deep Learning via *Recurrent* Neural Networks

- Why?: Rudimentary NN previously developed doesn't explicitly use previous time-steps and predictions as model inputs (doesn't have "memory") – but plasma evolution has spatial + *temporal* dependence



=



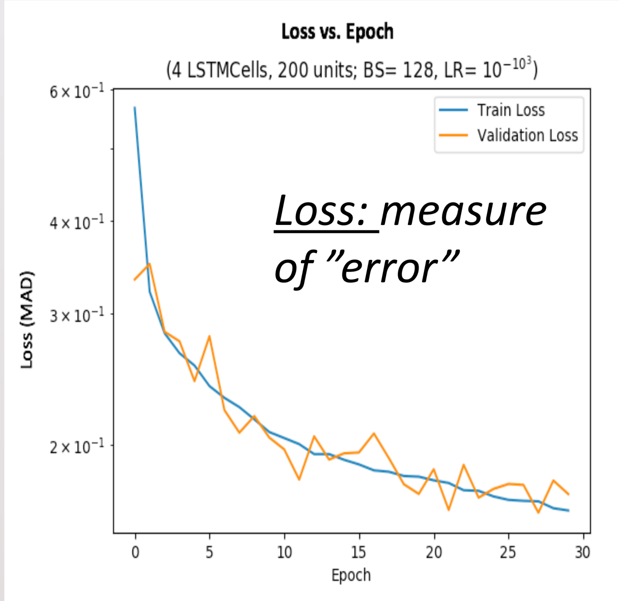
- Each node (LSTM unit) described by:

$$f_t = ReLU(W_f \cdot [C_{t-1}, h_{t-1}, x_t] + b_f) \rightarrow \text{Forget gate}$$
$$i_t = ReLU(W_i \cdot [C_{t-1}, h_{t-1}, x_t] + b_i) \rightarrow \text{Input gate}$$
$$o_t = ReLU(W_o \cdot [C_t, h_{t-1}, x_t] + b_o) \rightarrow \text{Output gate}$$

- Each "neuron" is its own neural network – node learns non-linear combination of values to determine gates' operation for ML task

Implementing RNN in NERSC Supercomputers

- Awarded NERSC exploratory resources to upscale project
- Implementing RNN model in Cori and Edison Supercomputers
  - Necessary: increase accuracy → increase computational power



Future Work

Model in early stages (lots still to do):

- Larger training shot list (model currently trains on ~600 shots– lots more data available)
- More 0D data (power and torque from each beam – potentially valuable influence on local spatial scales)
- Comparing profile evolution predictions for data-driven and physics-based models
- CNN-RNN hybrid - learn more abstract spatial patterns
- Parameterize input data – reduce computational expense
- Kinetically-constrained plasma profiles
- Train model on plasma transport codes (e.g. TRANSP)

References and Acknowledgements

- [1] Long Short-Term Memory, Hochreiter et al, Neural Computation 1997
- [2] Adam: A Method for Stochastic Optimization, Kingma et al, ICLR 2015
- [3] A Brief Introduction to Machine Learning for Engineers, Simeone, KCL

- This work is supported by US DOE grants DE-FC02-04ER54698, DE-AC02-09CH11466, and DE-SC0015878.
- NERSC, U.S. D.O.E. Office of Science User Facility operated under Contract No. DE-AC02-05CH11231.
- Princeton Plasma Physics Laboratory, DOE for funding the Summer Undergraduate Laboratory Internship (SULI)
- Patrick Vail, Florian Laggner, and Yichen Fu for their insight to fusion plasmas.