Disruption Avoidance using Dynamic Deep Latent Variable Model-based Plasma State Monitoring

Y. Poels[1,2], A. Pau[1], C. Venturini[1], A. Bürli[3], C. Donner[4], G. Romanelli[4], O. Sauter[1], V. Menkovski[2], the TCV team[a] and the EUROfusion Tokamak Exploitation Team[b]

[1] *École Polytechnique Fédérale de Lausanne, Swiss Plasma Center, Lausanne, Switzerland*

[2] *Eindhoven University of Technology, Eindhoven, The Netherlands*

[3] *Centre Suisse d'Electronique et de Microtechnique, Predictive Analytics Group, Alpnach, Switzerland*

[4] *Swiss Data Science Center, Zürich & Lausanne, Switzerland*

Introduction

Disruptions form one of the main threats to the reliable and continuous operation of tokamaks. Avoiding and mitigating disruptions is of key importance, and significant progress has been made using both physics-based and data-driven methods [1]. This work explores disruption avoidance through data-driven methods, leveraging the vast amount of data from past experiments. Models such as deep neural networks have shown much success in disruption prediction [2], but their black-box nature makes them difficult to use for active avoidance. Conversely, latent variable-based models aim to represent the plasma state by mapping high-dimensional observables (measurements) to a low-dimensional compressed state. This state can then be used to map and monitor the plasma disruptivity, by comparing the projection of the current measurements to statistically extracted disruption boundaries [3, 4]. While this approach provides much more interpretability, there is still room for extension. In this report, we conduct an initial investigation into extending latent variable-based models to be more informative with respect to identifying (disruptive) plasma regimes and to model dynamics in the full discharge. Specifically, we extend the Variational Autoencoder (VAE) framework [5]. A VAE is a generative model that aims to approximate the data distribution $p(\boldsymbol{x})$ of variable $\boldsymbol{x}$. It is assumed that this high dimensional variable $\boldsymbol{x}$ can be represented using a lower-dimensional latent variable, referred to as $\boldsymbol{z}$. This can be expressed as $p(x) = \int_z p_\theta(x|z)p(z)dz$; recovering the data distribution by marginalizing over $\boldsymbol{z}$. A distribution is chosen for prior $p(z)$, $p_\theta(x|z)$ (the 'decoder') is approximated by a neural network, and additionally an approximate posterior distribution $q_\phi(z|x)$ (the 'encoder') is learned by a neural network (due to the difficulty in directly computing $p_\theta(z|x)$). In this context, $\boldsymbol{x}$ covers diagnostic data from fusion experiments, and $\boldsymbol{z}$ an abstract compressed representation of this data; by virtue of describing the observations of experiments we consider $\boldsymbol{z}$ to be an approximation of the plasma state.

[a] *See the author list of H. Reimerdes et al 2022 Nucl. Fusion 62 042018*

[b] *See the author list of E. Joffrin et al 2024 Nucl. Fusion in press*

The formulation is extended with the main objective of tracking the plasma state. Specifically, we implement the following three points: (1) an extension of prior formulation $p(z)$ to gaussian mixtures in order to better model different plasma regimes; (2) a residual formulation for the approximate posterior as $q_\phi(z_i|x_i, z_{<i})$ (time index i) to model time in a continuous fashion; and (3) an estimate for the proximity to disruption as function of the plasma state: $f_\varphi(z_i)$. For the latter, this proximity is defined as a value in the range [0, 1], with 0 indicating no disruptivity and 1 the time of disruption, taken from the EUROfusion Disruption Database [6]. To define this value everywhere, a linear ramp is added for time points just before the disruption, i.e. the disruptivity at index i in a discharge is defined using time t_i and time of disruption t_D:

$$disr_i = \begin{cases} 0, & \text{if } (t_D - t_i) > \Delta t_D \text{ or } t_D = \emptyset \\ 1 - (t_D - t_i)/\Delta t_D, & \text{otherwise} \end{cases}$$

for a chosen window $\Delta t_D = 0.25\text{s}$. Ideally this window would correspond to the relevant disruptive chain of events for each discharge, although gathering this information on a large scale is challenging—identifying this window from data also presents an interesting future direction. Nevertheless, here we make a simple approximation to serve as a useful basis allowing the exploitation of thousands of shots.

Dataset

We fit the model using data from the Tokamak à Configuration Variable (TCV). We use a dataset of 2586 discharges, using 2504 for the train set and 82 as the test set on which we report results. Disruption times are taken from the EUROfusion Disruption Database [6], with 1728 disrupted shots and 755 regular terminations (split 41:41 for the test set). Each shot is represented as the full timetrace at 10KHz (interpolated using the last known measurement) for 21 signals that cover information about the plasma regime and/or disruptivity, which are:

I_p $[A]$, q_{95}, β_p, l_i, P_OH $[W]$, W_tot $[J]$, A_p $[m^{-2}]$, κ_{edge}, δ_{edge}, γ_VGR $[Hz]$, $n_\text{e,core}$ [**FIR**, m^{-2}], $\langle n_\text{e} \rangle_{\rho=0.85-0.95}$ [**TS**, m^{-3}], $\langle T_\text{e} \rangle_{\rho=0.85-0.95}$ [**TS**, eV], $n_{\text{e},peak,\text{N}}$ [**TS**], $T_{\text{e},peak,\text{N}}$ [**TS**], LM $[T]$, $RMS_{(n=0)}$ $[T]$, $RMS_{(n=1)}$ $[T]$, $RMS_{(n=2)}$ $[T]$, $P_\text{rad,bulk}$ $[W]$, $P_\text{rad,tot}$ $[W]$.

Model setup

We fit the model using a prior distribution with 5 modes equally distant from each other, i.e. a regular pentagon, see Figure 1a. While this does not impose a hard constraint, since the approximate posterior can deviate from the prior, it favors a specific partitioning of different regimes in the latent space. Currently this choice is somewhat arbitrary—a systematic study of appropriate choices for the prior is planned for future development. Alternatively, methods could be explored to learn this shape from the dataset itself. In Figure 1b, the approximate posterior distribution (i.e. the neural network encoder after training) is plotted. It follows the