WPJET1-CPR(18) 20131

GA Ratta et al.

Multidimensional linear model for disruption prediction in JET

Preprint of Paper to be submitted for publication in Proceeding of
30th Symposium on Fusion Technology (SOFT)

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Multidimensional linear model for disruption prediction in JET

G.A.Rattá[1], J.Vega[1], A. Murari[2] and JET Contributors*

[1]Laboratorio Nacional de Fusión. CIEMAT, Madrid, Spain.
[2]Consorzio RFX (CNR, ENEA, INFN, Universitá di Padova, Acciaierie Venete Sp A) Corso Stati Uniti 4, 35127 Padova. Italy
*See author list in the paper, X. Litaudon et al., Nucl. Fusion, 57 (2017) 102001

Corresponding author's e-mail: giuseppe.ratta@ciemat.es

The implementation of Machine Learning (ML) techniques has considerably improved the prediction of disruptions. However, they usually provide outcomes difficult to understand from a physics point of view due to their mathematical formulation. In this work an interpretable linear equation has been derived from an accurate ML disruption predictor. It can be used for real-time forecasting and the off-line analysis of the variables that contribute to the alarm triggering. To create the linear model, in addition to physic quantities, Time Increments (TIs) have been considered. TIs represent the variation of two amplitude values of a signal X at two different times divided by their temporal difference (i.e. $\Delta X/\Delta t$). To select the best subset of quantities for training purposes among the wide possible combinations of signals and Tis, Genetic Algorithms have been applied. The results, obtained over an independent testing database of 131 unintentional disruptive and 1310 non-disruptive shots, are 99,24% of success rate (94,66% of them with at least 10 ms of warning time) and 3,51% of false alarms.

Keywords: disruption prediction, linear equation, machine learning, JET.

1. Introduction

During disruptions [1] the plasma confinement is lost in the order of milliseconds. As result, extreme forces and heat loads able to provoke severe damages to the device are induced and released. In case their occurrence unavoidable, the last resort measure to mitigate their effects is the fast injection of pellets [2] or gas [3] into the vessel. A fundamental pre-requisite to start these palliative actions is to anticipate, in real-time, the incoming disruption, in this way opening a time window to intervene. The prediction of disruptions has improved with the application of Machine Learning (ML) techniques [4-6]. However, the data-driven models created with these methods supply as output a very complex mathematical formulation. For a more efficient prediction it is required to develop models understandable enough to unravel their meaning and to apply the most appropriate rescue or mitigation action.

The aim of this work is to take advantage of ML methods to create an accurate predictor of disruptions and, at the same time, to force the ML engine to deliver an interpretable solution. This implies a trade-off between precision (normally reached in ML algorithms with highly non-linear formulations) and simplicity (a linear equation).

Raw signals and processed ones (here called Time Increments (TIs)), have been computed to provide to the ML system (described in Section 3) with a wide variety of possible inputs. To select, from this large set of inputs (10 signal values plus 70 TIs) an appropriate subset to train the predictor, Genetic Algorithms were used (see Section 4). The obtained results are detailed in Section 4 and discussed in Section 5.

2. The database

A large database was collected from JET for this work. It contains a total of 2541 discharges (231 of them are unintentional disruptions and 2310 are non-disruptive) corresponding to the period between the January 2012 and November 2016.

The database has been divided in two main groups. The chronologically first 100 disruptive shots and 1000 non-disruptive shots have been used to train and validate the predictor (T/V database). The last 131 disruptive and 1310 non-disruptive shots have been saved for testing the final model (Testing database).

The fast terminations of some pulses after the use of the Disruption Mitigation Valves (DMVs) could lead the discharge towards a disruption. To avoid any bias, these pulses have been not included in the database since the ML system can learn to detect the firing of the DMVs as a disruption precursor and falsely improve its rates.

For each discharge, the 10 signals shown in Table 1 were considered for their possible use. Each parameter was resampled to 1 ksample/s following the methodology detailed in [7]. In general, to achieve a predictor with high accuracy using ML techniques, it is necessary to generate highly non-linear models which are in direct conflict with the main objective of this work: linear relationships.

This problem could be avoided by choosing a proper set of parameters (raw signals and processed values from those signals). A significant innovation introduced here are the Time Increments (TIs). TIs represent the local increment of the amplitude values of a signal X (ΔX)

author's email: giuseppe.ratta@ciemat.es

Table 1. List of the considered signals.

Signal	Acronym	Unit
1- Line integrated density	Dens	m-2
2- Plasma elongation	ELONG	
3- Plasma current	IPLA	A
4- Plasma internal inductance	LI	
5- Plasma vertical centroid position	PVP	m
6- Toroidal magnetic field	Bt	T
7- Mode lock amplitude	Mlock	T
8- Total radiated power	Rad	W
9- Time increment of the stored diamagnetic energy	Wdia	W
10- Poloidal Beta	BP	

over a predefined temporal difference (i.e. $\Delta X/\Delta t$). In this article, TIs use **7** values of Δt: {1; 2; 16; 32; 64; 128; 256} (ms).

TIs provide 2 important advantages for the real-time application of the model. First, they get rid of possible off-sets in the signals: their computation is performed between closely separated samples and any inherited DC component is avoided. Second, they supply past information of the ongoing discharge (past samples are involved in their calculation).

The 10 signals plus the 70 TIs (7 TIs from 10 signals) are processed, adding up a total 80 parameters. These parameters are computed every 1 ms and condensed in a *80x1* dimensional vector, called Feature Vector (FV). For its computational treatment, each pulse is processed as a concatenation of 1 ms FVs, each one of them with 80 parameters.

3. The applied machine learning techniques

3.1 Support Vector Machines

Support Vector Machines (SVM) [8] is ML technique effectively applied in previous works to predict disruptions [9].

During the training phase, SVM computes a hyper-plane able to split a set of examples with the maximum possible margin. The examples can be *pre*-disruptive (extracted **10 ms before the disruption**) or *non*-disruptive (extracted at a random time from non-disruptive discharges). Therefore, SVM generates a function able to classify whether each example is pre-disruptive or not. This training process is performed off-line but once the splitting hyper-plane has been calculated, it can be directly applied to new samples under analysis from FV of running discharges. In case the new sample is placed in the pre-disruptive side of the already computed hyper-plane an alarm is triggered.

In most of the cases the training set is non-linearly separable. To solve this problem by applying the mathematical principles described in [8], SVM uses non-linear Kernel functions. Kernels transform the space of inputs into a higher dimensional feature space where the margin maximization performed by SVM can be applied. However, the resulting models are non-linear and of a complex formulations.

To avoid the use of non-linear Kernels and, at the same time, to reach a space where a linear separation is possible, the inclusion of more parameters (in this case, the TIs) is necessary. The goal is to select, among the *80x1* dimensional FVs, a subset of linearly separable variables. A straightforward way to find this arrange of parameters would be through a brute force methodology, training and testing a different SVM predictor for each possible combination of the 80 signals and TIs. However, this procedure would carry unaffordable computational times due to the immense number of possible combinations to be assessed. It has been estimated that an exhaustive analysis could take up to **hundreds of thousands of years** [9]. To solve this problem in **~4 days** Genetic Algorithms (GAs) have been applied.

3.2 Genetic Algorithms

GAs [10] are optimization methods suitable for solving the combinational problem stated at the end of the previous Section. Instead of using brute force, the heuristic of GAs is based on combining promising solutions to create newer and better ones in an iterative process. Since the essence of the procedure here applied has been described in [9], this Section will be devoted only to the main concepts of the methodology.

GAs optimization is performed by the following set of iterative steps:

Step 1 consists of creating the first population of individuals. The size of the population has been pre-defined as 120 based on the criterion proposed in [9]. Each individual of the population is a *codified* set of *instructions*. The codification consists of summarizing, in a *80x1* dimensional string, which signals and TIs must be included. Each position/box of the 80x1 string is associated to a specific parameter (e.g. box 10 is linked to the $IPLA_8$ TI). Only in this first iteration, random values of '1' or '0' are placed into each box of the 120 strings/individuals. Consequently, each individual is filled with ones and zeros. A '1' means that the signal or TI linked to that specific box must be used to train a predictor candidate. The ones tagged with a '0' must not.

In Step 2 the training of the candidate predictors is carried out. Each individual/string (created in Step 1) has the set of instructions (ones and zeros) to pick the inputs for SVM. The signals and TIs with an associated '1' are extracted from randomly selected half of the T/V dataset discharges (50 pre-disruptive samples and 500 non-disruptive samples). Using these samples, SVM creates a predictor candidate (hyper-plane). In this step, the 120 individuals become 120 disruption predictor candidates.

Step 3 is devoted to evaluate the candidate predictors using a Fitness Function (FF). The idea behind GAs is to drive the population to evolve towards the pursuit objectives defined by this FF. The FF quantifies the 'goodness' of each candidate and may include several objectives to be optimized, each one of them with a different priority. The scores have been given as follows: 8 points are added when the candidate predictor correctly triggers an alarm at least 10 ms before the disruption (minimum required time to develop mitigation action in JET). 7 points are assigned each time an alarm is NOT activated in a NON-disruptive pulse. Finally, since one of the main objectives of this work is to derive an interpretable and concise equation, the final score of the candidate (determined by the sum of all points) is increased by 5% in case it uses fewer variables than the mean of its generation.

In Step 4, parents are selected to interchange their instructions (ones and zeros). The candidates with higher FF values have more chances to be selected as parents. This selection was performed using the *roulette* method, explained in [9]. As consequence, 60 pairs of parents are chosen.

The instructions enclosed in the strings of these 60 parents (ones and zeros) are interchanged in Step 5 to create children. To do that computationally, the crossover operator [9] is applied. It consists of randomly interchanging boxes of parents' strings to create 2 children. Then, the children are a mixture of parent's strings. The process is repeated for the 60 pair of parents. As result, a new population of 120 children is generated. Also, the mutation operation has been also implemented. It is useful to increase the diversity of the gene pool. It consists of flipping a randomly selected box value in the creation of children. A standard value of 0.05% of mutation probability was set for this operation.

Finally, in Step 6 the new population of 120 children is used to replace the previous one. The GA iterates again with this new population from Step 2 until an ending condition is reached. The ending condition is to develop 50 generations. 50 iterations were sufficient to achieve the expected results in a total time of ~4 days. Once that ending condition is fulfilled, the **final predictor** is selected as the one with the highest FF score from all 50 generations.

4. Results

The **final predictor**, selected from all the iterations of the GAs evolution, is the linear inequality expressed in equation 1. This clear formulation makes simple its real-time application. It is just necessary to evaluate the new ongoing pulses every 1 ms replacing the corresponding values in equation 1. In the case that the total value is lower than 0, no alarm needs to be triggered and the next time slice of 1 ms (next FV of the running discharge) is analyzed. Otherwise (equation 1 > 0) a disruption alarm is activated.

$$0{,}04.BP + 3768.MLock - 342{,}79.PVP256 + 260{,}30.TB128 + 5{,}87.106.Wdia128 - 2{,}16 > 0 \quad (1)$$

Subscripts in this equation represent the Δt of the TI used. In this form the equation is not easy to interpret due to the high variability (of several orders of magnitude) of the involved variables. For instance, it is not evident whether the $Wdia_{128}$ TI or the ML signal is a more important factor (in the sense of relative weight) to be taken into account.

For its better interpretation, this equation has been normalized (see equation 2). The normalization was performed by extracting the average maximum and minimum values (for every parameter of equation 1) of the non-disruptive discharges. A standard rescaling for each signal and TIs to ~[0, 1] using these averaged maximum and minimum values was performed.

$$0{,}07 \,.\, BP^{norm} + \mathbf{0{,}95}.MLock^{norm} - 0{,}18.PVP_{256}^{norm} + 0{,}06.TB_{128}^{norm} + \mathbf{0{,}23}.Wdia_{128}^{norm} - 1{,}96 > 0 \quad (2)$$

It is important to remark that equation 1 and equation 2 (with the normalized parameters) are equivalent. Since equation 2 is normalized, the contribution of each parameter can be directly appreciated as the weight multiplying each term.

According to the formula, the most relevant factor to predict disruptions is the amplitude of the raw ML signal, with a weight of **0.95**. This was an expected outcome since this magnitude has been historically used in JET as the main parameter to be considered to predict disruptions. The accompanying TIs in the formula provide smaller contributions (with the exception of the Wdia TI) but they are fundamental to reduce the false alarms and to achieve the accurate prediction rates shown in figure 1. This figure represents the summary of the predictor performance over all the disruptive discharges of the independent Testing dataset. Each one of the pulses was evaluated from the beginning of the current flat-top simulating a real-time scenario. In this figure warning time represent the anticipation (in s) of the alarm with respect to the disruption times. Notice that, for 10 ms of warning time (minimum anticipation in JET to develop mitigation actions) ~95% of the disruptions are predicted. The total rate reaches 99,24%, meaning that only 1 disruption was not detected in advance by the model (shot #92312, that suffered a minor disruption during the termination of the pulse).

The false alarm rate was computed as the percentage of alarms triggered (incorrectly) in discharges that ended without a disruption. This error was made in 46 shots(from the total 1310 of the non-disruptive Testing database), representing 3,51%.

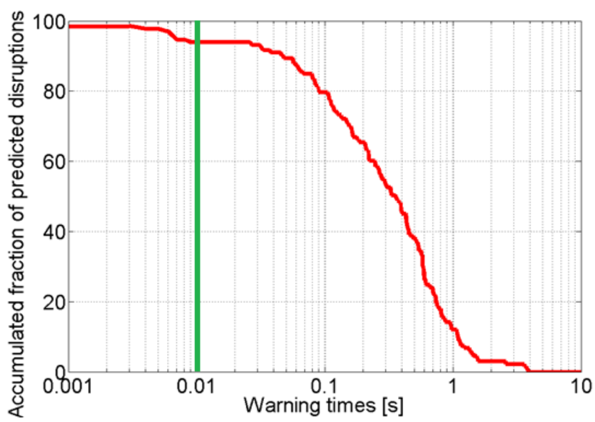

Fig. 1. Summary of the predictors' performance over the independent Testing dataset. Only 1 disruption is not anticipated (total detection rate of 99,24%). The ~95% of the alarms are triggered at least 10 ms before the disruption.

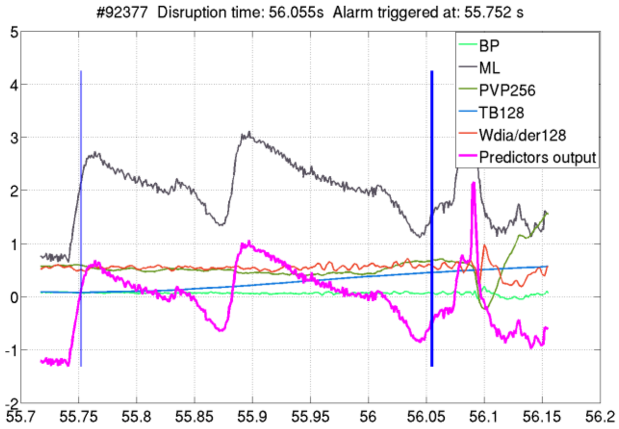

Fig. 2. Evolution of the predictor's output and the normalized parameters of equation 2 for shot #92377. The alarm is triggered ~300 ms before the disruption.

An example of a correctly activated alarm is provided in figure 2. There, the evolution of the normalized quantities in equation 2 and the predictor's output (sum of the weighted normalized variables) are shown. The main shape of the predictors' output follows the one of the MLock signal. The TIs contribute to automatically adjust a threshold (their changes are less sudden and they supply an optimized off-set level) to trigger the alarms. In this case the output value of the predictor crosses the zero value (thin vertical line) activating an alarm ~300 ms before the disruption (thicker vertical line).

5. Summary and conclusions

In this article a disruption predictor able to catch 130 of the 131 (99,24%) disruptions was developed. The results have been obtained over a wide and independent Testing dataset simulating real-time conditions. The 94,66% of the disruptions were predicted with an anticipation of at least 10 ms (minimum time required to start mitigation actions in JET) and triggering only a 3,51% of false alarms.

Beyond the obtained rates (comparable to the ones reached by the best ML predictors developed for JET) the main contribution of this work is the clear formulation of the final model.

The inclusion of an innovative signal processing (the TIs) and the application of GAs to select the appropriate combination of parameters, made it possible to take advantage of the precision of ML systems to derive a linear equation. The final predictor relays strongly in the ML signal but the use of the TIs is fundamental to attain higher accuracies with a reduced amount of false alarms. In figure 2 it can be observed that the time evolution of the TIs is fairly constant. Therefore, they could be replaced as constants (they evolution is not rapid) in equation 2. This suggests that they are supplying an automatic threshold value for the MLock signal (the only variable that shows sudden changes in the equation).

A deeper study of the equation can help to understand the reason why almost all the alarms are triggered and to develop accurate and robust predictors. In sights of ITER, a promising solution is to take advantage of an already existing tokamak database, to train with it a ML predictor and to extrapolate the resulting model to newer devices. This methodology has been already suggested in [11] but to reach a precise fit of the predictor into the new device it is fundamental to find reliable scaling factors, as it is concluded in [12]. The analysis of the TIs may provide hints to understand these possible scaling factors for the predictors, key to develop a general solution able to be applied in ITER.

Acknowledgments

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission. It was partially funded by the Spanish Ministry of Economy and Competitiveness under the Project ENE2015-64914-C3-1-R and ENE2014-56517-R..

References

[1] A.H. Boozer, Theory of tokamak disruptions, Physics of Plasmas.19, 058101. 2012.

[2] N. Commaux et al., First demonstration of rapid shutdown using neon shattered pellet injection for thermal quench mitigation on DIII-D, Nuclear Fusion, 2016, vol. 56, no 4, p. 046007.

[3] U. Kruezi et al., A new disruption mitigation system for deuterium–tritium operation at JET, Fusion engineering and design (2015) 286-289.

[4] G.A. Rattá et al., An advanced disruption predictor for JET tested in a simulated real-time environment, Nucl. Fusion 50 025005 (2010).

[5] G. Pautasso et al., An adaptive real-time disruption predictor for ASDEX Upgrade, Nucl. Fusion 50 075004. 2010.

[6] J. Vega et al., Results of the JET real-time disruption predictor in the ITER-like wall campaigns, Fusion Engineering and Design. 6–8. (2013) pp 1228–1231.

[7] J.M. Lopez et al., Integration and Validation of a Disruption Predictor Simulator in JET, Fusion Science and Technology 63.1 (2013): 26-33.

[8] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, 1995.

[9] G.A. Rattá et al., Global optimization driven by genetic algorithms for disruption predictors based on APODIS architecture, Fusion Engineering and Design 112, 1014-1018 (2016).

[10] K.A. De Jong et al., Parallel Problem Solving from Nature. LNCS, vol. 496, pp. 38-47. Springer, Heidelberg. 1991.

[11] C.G. Windsor et al., A cross-tokamak neural network disruption predictor for the JET and ASDEX Upgrade tokamaks, Nucl. Fusion 45 337. 2005.

[12] G.A. Rattá et al, Viability Assessment of a Cross-Tokamak AUG-JET Disruption Predictor, Fusion Science and Technology, 1-10, (2018).