Deep convolutional neural networks for multi-scale time-series classification and application to tokamak disruption prediction using raw, high temporal resolution diagnostic data

R.M. Churchill[a], B. Tobias[b], Y. Zhu[c], and the DIII-D team[1]

[a] *Princeton Plasma Physics Laboratory, 100 Stellarator Road, Princeton, NJ 08540, USA*
[b] *Los Alamos National Laboratory, Los Alamos, NM 87545, USA*
[c] *University of California at Davis, Davis, CA 95616, USA*

Abstract

In this paper we discuss recent advances in deep convolutional neural networks (CNN) for sequence learning, which allow identifying long-range, multi-scale phenomena in long sequences, such as those found in fusion plasmas. We point out several benefits of these deep CNN architectures, such as not requiring experts such as physicists to hand-craft input data features, the ability to capture longer range dependencies compared to the more common sequence neural networks (recurrent neural networks like long short-term memory (LSTM) networks), and the comparative computational efficiency. We apply this neural network architecture to the popular problem of disruption prediction in fusion energy tokamaks, utilizing raw data from a single diagnostic, the Electron Cyclotron Emission imaging (ECEi) diagnostic from the DIII-D tokamak. Initial results trained on a large ECEi dataset show promise, achieving an F_1-score of ∼91% on individual time-slices using only the ECEi data. This indicates the ECEi diagnostic by itself can be sensitive to a number of pre-disruption markers useful for predicting disruptions on timescales not only for mitigation but also avoidance. Future opportunities for utilizing these deep CNN architectures with fusion data are outlined, including impact of recent upgrades to the ECEi diagnostic.

1. Introduction

Plasma phenomena contain a wide range of temporal and spatial scales, often exhibiting multi-scale characteristics (see Figure 1). In fusion energy plasmas, many disparate diagnostic instruments are simultaneously used in order to capture these various spatiotemporal scales, and to cover the multiple physics present in these plasmas. In addition, fusion experiments are increasingly built to run longer pulses, with a goal of eventually running a reactor continuously. The confluence of these facts leads to large, complex datasets with phenomena manifest over long sequences. A key challenge is enabling scientists/engineers to utilize these long sequence datasets to, for example, automatically catalog events of interest or predict the onset of phenomena.

Machine learning, and specifically the variant deep learning, has been proven to be highly successful in automating a number of tasks, such as identifying objects in images, language translation, and even the playing of strategic games such as Go [1]. Many deep learning architectures have been created and successfully applied to sequence learning [1, 2, 3] problems, in areas of time-series analysis or natural language processing. However, many of the typical architectures used for learning from sequences (e.g. recurrent neural networks (RNN) and its most popular variant long short-term memory networks (LSTM)) suffer from memory loss; long-range dependencies in sequences are difficult for these architectures to track.

In this paper we discuss recent advances in neural networks, specifically an architecture that uses dilated convolutions in a deep convolutional neural network (CNN), which was designed to overcome these problems of learning on long sequences. We use this architecture to predict oncoming disruptions in fusion plasma discharges of the DIII-D tokamak utilizing only raw data from a single, high temporal resolution imaging diagnostic (the Electron Cyclotron Emission imaging diagnostic, or ECEi). Because the ECEi diagnostic is sensitive to a range of multi-scale dynamics in the plasma related to disruptions [4], it offers the potential to more accurately predict them. Avoiding disruptions is a grand challenge for tokamak fusion devices on the road to fusion energy [5]. While much research has gone into utilizing machine learning for disruption prediction [6, 7, 8], often global, reduced 0-D features are used in shallow machine learning methods. Recently, work utilizing deep LSTM networks also added the use of low temporal resolution 1-D plasma profiles [8], and another work used a combination CNN/LSTM on resampled, low temporal resolution bolometer data [9]. The work we present here takes inspiration from these works in utilizing higher

Email address: rchurchi@pppl.gov (R.M. Churchill)
[1] C.C. Petty, the DIII-D Team, Nucl. Fusion 59 (2019) 112002.

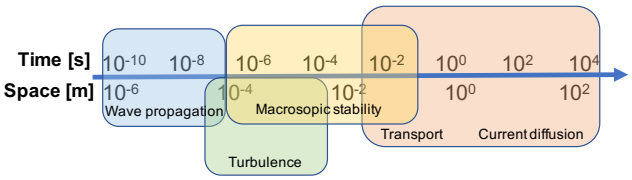


Figure 1: Example temporal and spatial scales of different broad physics phenomena in fusion plasmas, based on Ref. [10]

dimensional signals, and shows how to use newer deep learning architectures to learn on high-temporal resolution data with long-range dependencies due to multi-scale physics.

The outline for the rest of the paper is as follows: Section 2 discusses a paradigm for understanding deep learning and its usefulness in fusion plasma physics, Section 3 discuss typical sequence learning architectures, and the need for newer neural network architectures to capture long-range dependencies in sequences, Section 4 applies deep convolutional neural networks with dilated convolutions to ECEi data for disruption prediction, and Section 5 discusses future applications and directions for these networks in fusion.

2. A paradigm for deep learning

Deep learning has been tremendously successful in recent years in achieving state-of-the-art results for many machine learning tasks, including image classification, language translation, and speech recognition. Reference [1] provides a good review of the underlying principles of deep learning. Deep learning refers to neural networks with many hidden layers. Each hidden layer provides a linear transform (with a number of weights, W_j, along with a bias term b_j), followed by a non-linear transform (called the activation). By stacking up several layers, a deep neural network can potentially learn complicated non-linear functions. A task specific loss function is defined, and produces a measure of the error between predictions of a network, and the user-labelled targeted outcomes. Using this measure of error from the loss function, neural networks use a method called backpropagation [11] to update the weights of the neural network, with the goal of making the predictions match the targets. In this sense, a paradigm for deep learning is that deep neural networks are a series of filters whose coefficients (or weights) are "learned" instead of user-prescribed.

One of the reasons for deep learning's great success is the ability to learn multiple filters for high-dimensional data, avoiding the need for humans to do feature extraction [1]. This allows the deep learning algorithms to learn directly from raw data, for example using directly the pixels from a camera image to predict whether a cat is in the picture, instead of having humans to specify filters which can find features such as ovals (for eyes) and triangles (for ears).

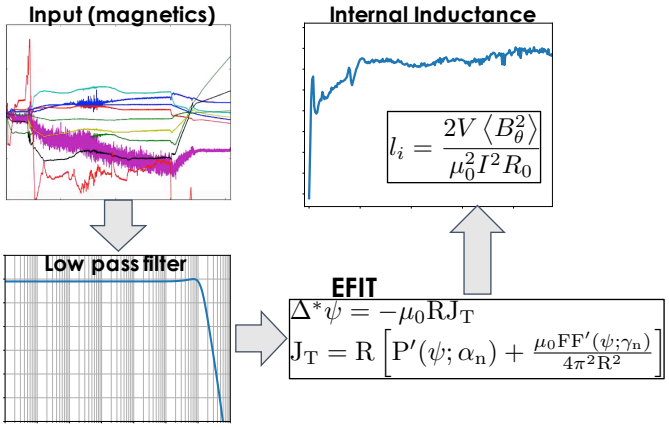


Figure 2: Schematic example of a typical fusion plasma analysis pipeline: raw data from the magnetics data is processed to extract the plasma internal inductance. A number of transforms or "filters" are applied to achieve this, including low-pass filter on the data, solving a PDE set of equations (idealistically matrix inversion multiplying magnetics input), and finally normalizing and averaging on surfaces of constant magnetic flux.

An example from fusion can aid in understanding how feature extraction is applied everyday by fusion physicists, and the potential for deep learning to learn the filters necessary to perform tasks. In fusion plasmas, often a variety of filters, transforms, and models are applied to measured data to extract physically relevant quantities. Figure 2 shows the example of extracting the normalized internal inductance, ℓ_i, from the raw magnetic diagnostic measurements. A series of transforms are applied: low pass filter, Grad-Shafranov equation using the EFIT code [12], and finally volume averages and normalizations to calculate the normalized internal inductance. ℓ_i is often a parameter passed into shallow machine learning architectures (e.g. Random Forests[7], SVM[6], etc.) for use in disruption prediction. Deep learning offers the potential to bypass this process and allow the algorithm to learn filters for identifying particular phenomena such as oncoming disruptions, neoclassical tearing modes, Alfven eigenmodes, etc.

3. Sequence learning architectures

For learning on sequences, such as time series or sentences in a document, several types of neural network architectures can be used. In this section we point out difficulties with traditional architectures used for sequence learning, and discuss how deep convolutional neural networks with dilated convolutions overcome these difficulties.

3.1. Traditional architectures

Traditionally, special neural network architectures called Recurrent Neural Networks (RNN) have been employed for sequence learning problems. These architectures have feedback connections among layers, allowing the use of information from previous parts in the sequence [2]. For

years the dominant flavor of RNN has been the long short-term memory (LSTM) network [13], which overcame difficulties with training RNN's on sequences with long-range dependencies. LSTM's use memory gates to enable them to effectively remember important pieces of past sequences, in theory making them able to remember dependencies in infinitely long sequences.

However, in practice LSTM's (and even more so RNN's) tend to suffer from memory loss, forgetting sequence events that occurred much earlier [14]. The length of sequence that LSTM's can remember will be highly dependent on the dataset, but a general empirical observation is sequence lengths in the thousands are the limit, i.e. a general rule of thumb for LSTM's to be useful is $T_{long}/T_{short} \lesssim 1000$ (where T_{short} is a fast timescale of interest in a sequence, and T_{long} is a longer timescale that needs to be captured for a particular prediction).

CNN's have also been used for sequence learning [15, 16], though traditionally are not as common as RNN/LSTM. One difficulty in using CNN's for sequence learning is for scenarios that need to respect causality (i.e. do not use future time points to make predictions), to be sensitive to long sequences you must increase the convolutional filter size and/or the number of layers in the network, both of which significantly increase the number of parameters for the network to learn [17].

3.2. Deep convolutional neural networks with dilated convolutions

Recently there has been much research into deep learning architectures which can overcome the deficiencies of RNN/LSTM's, and handle long, multi-scale sequences. A seminal paper presented one such architecture, WaveNET [17], which is a convolutional neural network (CNN) focused on generating realistic audio. One of the key insights of this paper was to use dilated convolutions to increase the receptive field of the network (i.e. the number of sequence points used by a neural network to make a prediction at a single time point). This overcomes the dilemma faced with using normal convolutions in causal networks, where to be sensitive to long sequences you must increase the convolutional filter size and/or the number of layers in the network. Dilated convolutions have a dilation factor (d) which represents the number of input points skipped between filter parameters, e.g. the sequence output $y[n]$ from a dilated convolution with dilation d is:

$$y[n] = \sum_{i=0}^{k-1} w[i]\, x[n - d \cdot i]$$

where w represents the weights of the 1D dilated convolution filter of length k, and $x[n]$ is the input sequence. A normal convolution results by setting $d = 1$. By stacking layers of dilated convolutions, and increasing the dilation factor in each layer, the receptive field of the network can be increased while maintaining a tractable number of model parameters. The difference between normal and dilated convolution architectures is shown in Figure 3.

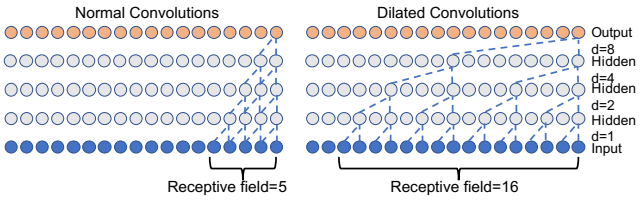


Figure 3: Normal vs dilated convolutions. The CNN with dilated convolutions has a much larger receptive field, with the same number of model parameters as the normal CNN. Figure modified from WaveNET paper [17]

Dilated convolutions impose an inductive bias or specific structure to the architecture which guide the transformations learned by the neural network. Specifically, dilated convolutions have a natural connection with wavelet structures, which have been used for separating out structure in multi-scale data, including turbulent flows [18]. In a loose sense, these neural networks allow us to learn the wavelet coefficients needed to accomplish our classification task.

A simplified yet powerful architecture named temporal convolutional network (TCN) [14] built upon this WaveNET work, utilizing dilated convolutions and many modern neural network techniques, such as weight normalization (normalizes layer weights to speed up training) and residual connections (connections which skip layers, found to stabilize deep neural networks). Bai *et. al.* [14] showed the TCN could outperform LSTM and a similar architecture, Gated Recurrent Unit (GRU) [19], on many common sequence learning tasks, especially for long sequences with long-range dependencies. It is this TCN architecture that we will now apply to the problem of disruption prediction using ECEi data.

Before continuing, we mention that besides deep CNN's with dilated convolutions, there are other neural network architectures being researched and developed to handle long range dependencies, most focusing on Natural Language Processing (NLP) applications. Many of these use attention mechanisms, which learn which parts of sequences are most important for certain predictions, to better focus the neural network on relevant parts of the sequence. The most popular of these are transformer networks [20, 21, 22]. Non-local networks [23] also use a mechanism similar to attention. In addition, there are CNN architectures which have been enhanced with attention mechanisms [15]. While attention can be computationally expensive, research is ongoing into making them more efficient [24]. It is left for a future study to explore applying these architectures to plasma time-series predictions.

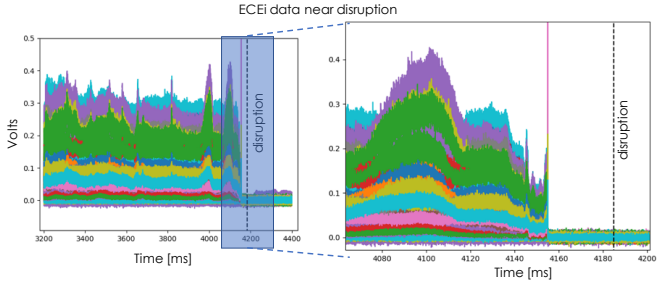


Figure 4: DIII-D ECEi diagnostic time-series data from each of the 160 channels of the LFS diagnostic, near a disruption event. The sudden drop in ECEi signal a few milliseconds before the disruption time (which is set at the current quench time) is due to the drop in temperature at the thermal quench. Shot #154056.

4. Application to Disruption Prediction using Raw ECEi Imaging Data

Disruptions in tokamak plasmas are a sudden loss of control which cause a termination of the plasma and potentially large destructive forces and/or heating on the containment vessel and protective wall materials. Next-step devices such as ITER and beyond will have a low tolerance for disruptions [25]. We need to ensure disruptions can be avoided by accurate prediction of oncoming disruptions and mitigation techniques if necessary.

Here we apply the TCN architecture to high-temporal resolution, raw ECEi imaging data from the DIII-D tokamak for the purpose of predicting oncoming disruptions[2].

4.1. DIII-D Electron Cyclotron Emission Imaging (ECEi) diagnostic

The ECEi diagnostic [26] records RF emission intensity at 16 tunable frequencies in the range of 2nd harmonic electron gyromotion and along 20 lines of sight. An example time series of the DIII-D ECEi data near a disruption is shown in Figure 4. Under conditions that are common in the core of many mid-sized tokamaks (optically thick, Maxwellian plasma having modest gradients in pressure and magnetic field, etc.) this signal is well-correlated to a local electron temperature [27, 28]. The diagnostician locates the spatial region of interest by tuning the heterodyne receiver to a harmonic of the local cyclotron frequency ($\omega_c = eB/m_e$) and adjusting the quasi-optical lenses that couple radiation to the two ECEi antenna arrays. However, for the analysis presented in the next sections, none of the metadata describing those operations is utilized; the neural net knows nothing of the instrument's design, operation, or potential correlation of the signal with physical quantities. The filtering and interpretation of raw ECEi data is spawned entirely from the machine learning procedure.

There are a number of different root causes for disruptions, including edge radiation, too high density, and MHD instabilities [29]. ECEi systems acquire data at such a high sampling rate and spatial resolution under ideal conditions that well-behaved signals span spatiotemporal scales to reflect the dynamics of turbulent fluctuations, Alfvén eigenmodes, tearing modes, sawteeth, ELMs, and other potential pre-disruption markers. Non-ideal conditions also impact the signal in ways that can be difficult for a human diagnostician to interpret, but are rich in information. For example, a sudden loss of signal can be the result of density cutoff. Alternatively, a sudden spike in signal can be the result of a non-thermal electron distribution. A wide range of other conditions can impact the signal, producing fluctuations or other features that machine learning techniques might become sensitive to, even when the human data analyst finds them to be troublesome or ambiguous. By using raw, un-processed ECEi data, we make full use of all these signal features.

4.2. Data, Model, and Training Setup

A dataset of 2,747 DIII-D shots (∼42% disruptive, ∼58% non-disruptive, ranging from shot 144199 to 167542) were selected from a shot database in the DISRUPTIONS module [30] of the modeling framework OMFIT [31]. Of the disruptive shots, about 45% were in the flattop phase of the discharge (full plasma current disruption). Shots were selected specifically where good ECEi data was available, defined as shots where each ECEi channel has a signal-to-noise ratio SNR > 3 (as a result, this excludes shots where ECEi was in cutoff). Even though the spatial location of the diagnostic can change between shots depending on plasma magnetic field and ECEi user settings, we purposely made no effort to down select to only aligned shots or do any kind of spatial interpolation. Instead, we chose to rely on the neural network to learn pre-disruption markers from a variety of locations. Time length of each shot varies, typically between 5 to 10 seconds. The entire dataset (approximately 10 TB) was transferred to the Princeton Research Compute center, to enable use of the Princeton Tiger GPU cluster, which has 320 Nvidia P100 GPU's, with 4 GPU's per node.

The ECEi data was taken in its raw, digitizer voltage output, i.e. without calibration to the actual electron temperature. The only corrections made were removing digitizer offsets by subtracting the average value before the shot begins ($t < 0$). As is common for neural networks[32], the input ECEi data was then normalized before input into the neural network, using the z-normalization. This is done by creating a per-ECEi channel mean ($\bar{x}$) and standard deviation (σ_x) over the entire dataset, then normalizing the input as $x_{norm} = (x - \bar{x})/\sigma_x$. Since $\bar{x}$ and σ_x are constant over the dataset, it should still allow meaningful correlations to be learned by the network, including relative changes in absolute temperature between channels.

For ease of training the neural network, we decided in this initial paper to temporally downsample the ECEi data to 100 kHz (i.e. factor of 10x less data). We also pass the ECEi channels into the TCN without applying any initial

[2]Code available at `https://github.com/rmchurch/disruptcnn`

2D convolutions in R, Z space. This means that the TCN will learn the temporal correlations between channels, but that we don't explicitly tell the network about the spatial relationship between channels (the spatial relationship between channels is known implicitly by the network, by passing the channels in the same order every time).

We treat the problem of disruption prediction as a binary classification problem, where we predict whether each time slice corresponds to a "non-disruptive" or "disruptive" class. Rea, et. al. [7] showed that data from time slices in a disruptive shot but far from the disruption time, $t_{disrupt}$, had similar data distributions as time slices from non-disruptive shots. Therefore, we label all time slices within 300ms of a disruption as "disruptive" ($t_{disrupt} - t < 300$ms), and all other time slices as "non-disruptive" [7] (sequences from shots without disruptions are taken during established times of the discharge, i.e. during the plasma current flattop). Note that we could have instead selected by hand a number of events important for predicting disruptions (e.g. tearing modes, temperature drop from radiation, etc.), and had the network instead predict these events (i.e. multi-class classification), which may have some advantages in identifying important characteristics of disruptions. Choosing a binary target (is this timeslice close to a disruption or not?) allows the network to learn implicitly a variety of these events through experience. Also note that we will aggregate timeslice prediction into shot predictions, and there will relax the requirement that only timeslices 300 ms before a disruption are considered "disruptive". The intuition here is that we expect disruptive precursors to be present in a majority of disruptive shots *at least* 300 ms before a disruption. Further work could shorten this time, attempting to guarantee disruptive precursors during the "disruptive" time slices, or lengthen this time, to better capture pre-disruption markers that may occur much earlier.

Typical binary cross-entropy loss was used for the loss function during the neural network training:

$$\mathcal{L} = -\frac{1}{n}\sum_{n} w_n \left[y_n \log \hat{y}_n + (1 - y_n) \log(1 - \hat{y}_n) \right] \quad (1)$$

where n is the number of timeslices in a batch, y_n is the target (binary, i.e. discrete, either 0 or 1), $\hat{y}_n$ the network prediction (continuous, ranging from 0 to 1), and w_n a constant class weight applied to help balance between disruptive and non-disruptive timeslices.

We define our TCN model to have a receptive field of N_{recept}~$30,000$. This is an order of magnitude larger than receptive fields in the original TCN [14] or WaveNET [17] papers. With the 100 kHz sampling rate, this means that each time slice prediction uses the previous $\sim$ 300ms in order to make the prediction. With our definition of disruptive time slices as within 300ms of the disruption, this implicitly assumes that 600ms before a disruption is sufficient to predict oncoming disruptions. We use a 4 hidden layer TCN with dilations $[1, 10, 100, 961]$ (i.e. increasing by a factor of about 10 each layer), with a filter kernel size of 15. The number of filters per hidden layer was held constant at 80 (varying number of filters per hidden layer was not attempted).

The TCN architecture allows parallelization of the sequence prediction by inputting sequences of length N_{seq}, which are longer than N_{recept}, resulting in $N_{seq} - N_{recept} + 1$ predictions per sequence. We are limited in the length possible due to memory constraints of the GPU, and the need to process a number of sequences from a variety of shots for best learning (the group of sequences processed for a single update of network weights is called a "batch"). Empirically it was found that sequence lengths of $N_{seq} = 78,125$ allowed model computations that fit inside the GPU memory constraints, while allowing a batch size of $N_{batch} = 12$ (per GPU) to ensure sufficient variety within each batch for training with stochastic gradient descent (i.e. each GPU processes $N_{batch} \times (N_{seq} - N_{recept} + 1)$ number of sequences for each weight update).

Multi-node, multi-GPU setup was used to parallelize the training. The Pytorch built-in synchronous data parallel training routine `DistributedDataParallel` was used [33], training on 16 GPUs over 2 days. This makes the total effective batch size with data parallelism $N_{batch} \cdot N_{gpu} = 192$. Larger batch size can be achieved reducing the sequence length, though at an increased computational cost due to more data reads.

Stochastic Gradient Descent (SGD) with Nesterov momentum 0.9 was used to train the model, with an initial learning rate of 0.5 that was decreased automatically upon plateau of the loss (`ReduceLROnPlateau`). A warmup period was used for the first 5 epochs (1 epoch is a single pass through the complete training dataset), increasing the learning rate from 0.0625 to 0.5 to enable larger batch training [34].

The set of sequences with timeslices consisting of only the majority class ("non-disruptive") was undersampled such that there were balanced disruptive and non-disruptive sequences[35], leading to a total data size of ~66 GB for training. 10% of the data was set apart as a hold-out validation set to determine during training how well the neural network was generalizing. Caution was taken to ensure that the training and validation datasets had no shots in common. Two validation metrics were tracked: accuracy (how many time slices were predicted correctly as disruptive or non-disruptive), and F1-score (a harmonic mean between precision and recall). Because the time slice classes are imbalanced (even though the sequence sets are balanced), the F1-score gives a better indication of how well our classifier does on the minority class (disruptive), as can be seen with its equation:

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2}{\frac{TP+FP}{TP} + \frac{TP+FN}{TP}}$$

where T stands for true (correctly predicted), F for false (incorrectly predicted), P for positive (disruptive), and N

for negative (non-disruptive). Since the output of the TCN network $\hat{y}_n$ is continuous, we need to determine a threshold to map the continuous output to a discrete prediction. We use the common technique of calculating the accuracy and F1-score using a range of thresholds (for our case from 0.05 to 0.95 in 0.05 increments), then select the threshold that maximizes the F1-score.

4.3. Results

The results of training this TCN model on ECEi data for disruption prediction on DIII-D are shown in Figure 5. Results are plotted over 1000 training epochs. The training binary cross-entropy loss continually decreases over the training, showing our model has the capacity to learn the task from this dataset. The validation loss also continually decreases, slightly flattening towards the end, indicating the model is reaching the limit of its generalizability after 1000 epochs. The two validation metrics are also shown in Figure 5, the metric of accuracy reaches ∼94%, but more importantly the metric of F1-score reaches ∼91%, showing the neural network has learned to predict individual time slices of both disruptive and non-disruptive time slices very well.

We use a hysteresis threshold method[36] to consolidate the time slice predictions to shot predictions. This method triggers a disruptive alarm if the neural network output rises above a high threshold, σ_{high}, and afterwards stays higher than a lower threshold σ_{low} for a time window τ_{alarm}. In this manner it avoids spurious spikes in the predictions. A Bayesian optimization scheme using the `optuna` package[37] was used to find the optimal thresholds and time window parameters to produce the highest performance. Unlike for the time-slice prediction, we do not only consider 300 ms before the disruption as "disruptive", but rather accept alarm triggers anytime during the shots (i.e. we allow an arbitrary class parameter from Ref. [36], $\tau_{class} \to \infty$), up to 30 ms before the disruption (a common quoted minimum time needed for disruption mitigation). This means that alarms anytime during a disruption shot will be considered a success (true-positive). The Bayesian optimization routine was set to minimize the distance to a perfect true-positive rate $TPR = TP/(TP + FN) = 1$ and false-postive rate $FPR = FP/(FP + TN) = 0$, i.e. the optimization objective was:

$$\mathbf{x}_* = \underset{\mathbf{x}}{\operatorname{argmin}} \sqrt{[TPR(\mathbf{y};\mathbf{x}) - 1]^2 + [FPR(\mathbf{y};\mathbf{x}) - 0]^2} \quad (2)$$

where $\mathbf{x}_* = (\sigma_{low}, \sigma_{high}, \tau_{alarm})$ is the optimal set of parameters, and $\mathbf{y}$ the neural network predictions. This produced parameters $\sigma_{high} = 0.96$, $\sigma_{low} = 0.96$, and $\tau_{alarm} =$ 1ms, with an shot F1-score of 0.944. A full grid search of the parameters was done to verify, and the resulting receiver-operator curve showing FPR versus TPR for each parameter combination on the validation dataset is shown in Figure 6, along with the convex hull showing the best points. Using the convex hull, this gives an AUC of 0.963.

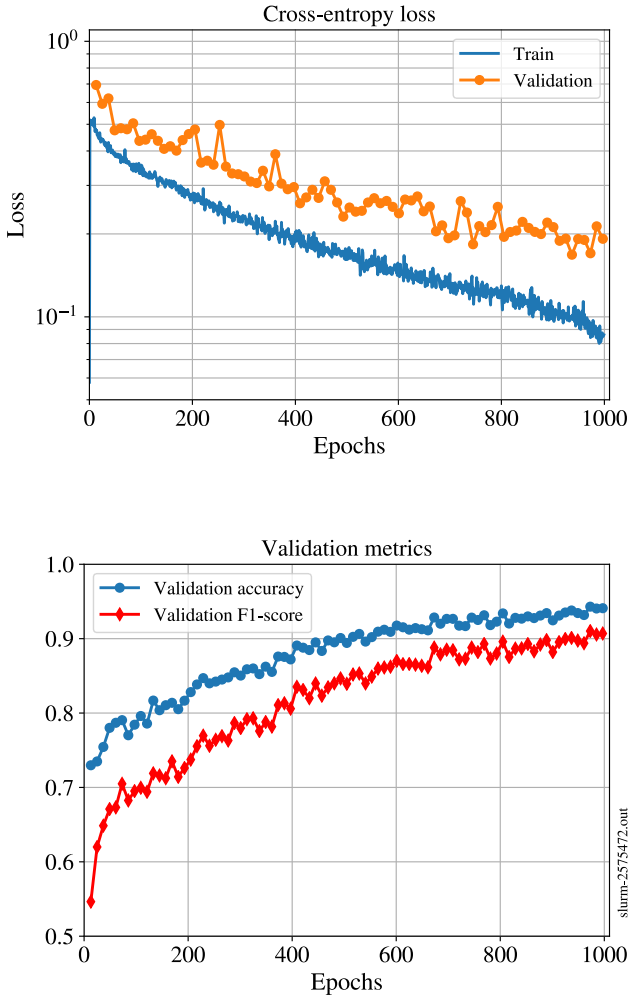


Figure 5: Results from training the TCN on ECEi data. The top plot shows the binary cross-entropy loss, both for the training dataset (the data used to train the model) and the validation set (used to monitor how the network generalizes on unseen data). The loss continually decreases as the neural network learns from the data over each training pass ("epoch"). The accuracy and F1-score are shown in the bottom plot. These metrics are for timeslice predictions.

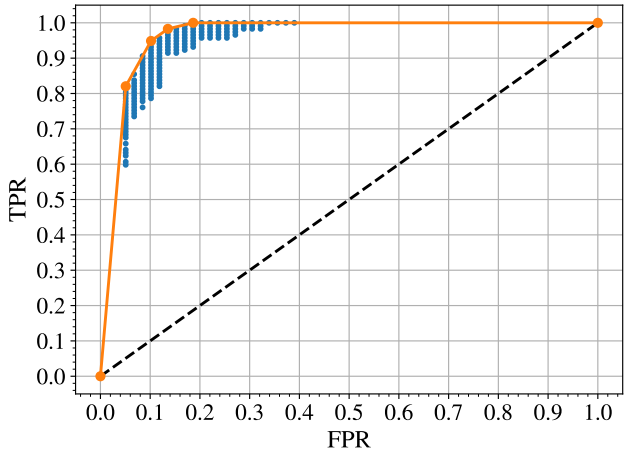


Figure 6: Receiver-operator curve for the validation set. Blue dots are for varying alarm parameters, and the orange points the convex hull of this set.

		Predicted	
		Disruptive	**Not-disruptive**
Actual	**Disruptive**	TP = 112	FN = 6
	Not-disruptive	FP = 28	TN = 129

Table 1: Confusion matrix for the shot predictions on the holdout test set.

These optimal parameters were then used with the holdout test dataset predictions, in order to test how this predictor and set of alarm parameters would perform in the future. This produced a shot F1-score of 0.868, with the number of shots correctly and incorrectly predicted (i.e. confusion matrix) shown in Table 1. As seen the true positive rate is encouraging, at ∼94.9%, however the false positive rate is too high, at ∼17.8%. We hypothesize the high false positive rate is due to the undersampling of non-disruptive shots, and that with further training, and instead oversampling this could be improved. Current machine learning disruption predictors typically achieve a true-positive rate in the low 90% on shots [6, 7, 8, 9], but with more reasonable false-positive rates (< 10%). The rough goal is a true-positive rate of >95% with a false-positive rate of <5%[25]. The results presented here show further work needs to be done with this machine learning technique, but, considering this utilizes a single diagnostic, shows promise in being able to contribute to a machine learning disruption prediction solution.

The warning times given by these disruption predictions are significantly before the minimum time needed for disruption mitigation (30 ms), as shown in Figure 7. The majority arrive at least 200 ms before the disruption, giving sufficient warning time to in principle use control algorithms for disruption avoidance[30]. As seen, approximately 15% of the detected disruptions were detected more than 1 second before the disruption, though more detailed

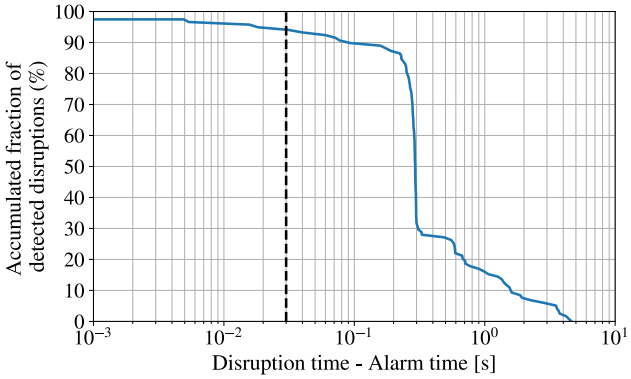


Figure 7: Warning time (i.e. time before a disruption when the trained predictor raises the alarm) for the detected disruptions. All disruptions are detected at least 500 ms before the disruption occurs.

analysis is needed to determine whether these were based on true disruption precursors, or should be counted as false-positives.

Examining the output of the TCN disruption predictor can give insight into what the neural network has learned from the ECEi data. We first show in Figure 8 an example shot with a flattop disruption, which the TCN predictor correctly identified roughly 260 ms before the disruption. Significant $n = 1$ mode activity is present, and appears to lock near 3.0 seconds. The plasma survives another 500 ms, with the TCN disruption predictor alarm raised 220 ms after the mode locks (within the receptive field of the TCN). Notice also, though, that the TCN disruption predictor has high output during sudden drops in electron temperature near 2.0 seconds, though never sustained over 1 ms to raise the alarm.

There is ongoing work identifying the failure mechanisms of the TCN disruption predictor, particularly the false-positives (i.e. alarm raised during non-disruptive shots). Figure 9 shows such a shot, where the TCN disruption predictor appears very certain a disruption is imminent. Examining the plasma time traces, this shot appears to have a minor disruption[38], where there is a strong drop in core electron temperature at 2.0 seconds, but the plasma recovers. There is MHD activity leading up to and during the drop, as seen in Figure 9. The TCN disruption predictor appears to trigger during the locking of the $n = 2$ mode, but closer analysis of magnetics also shows a $m/n = 2/1$ mode present just previous to when the TCN disruption predictor triggers. Further techniques to identify the salient features that cause the neural network to trigger will be discussed in the following section. As the sawtooth amplitude and period are often modified near disruptions (as in Figure 8), due to MHD mode coupling, the fact that the sawteeth do not substantially change could have been an indicator that this shot would not completely disrupt (though there is some evidence of sawteeth modification near minor disruptions also)[39]. The ECEi diagnostic does capture sawteeth behavior, nevertheless, this

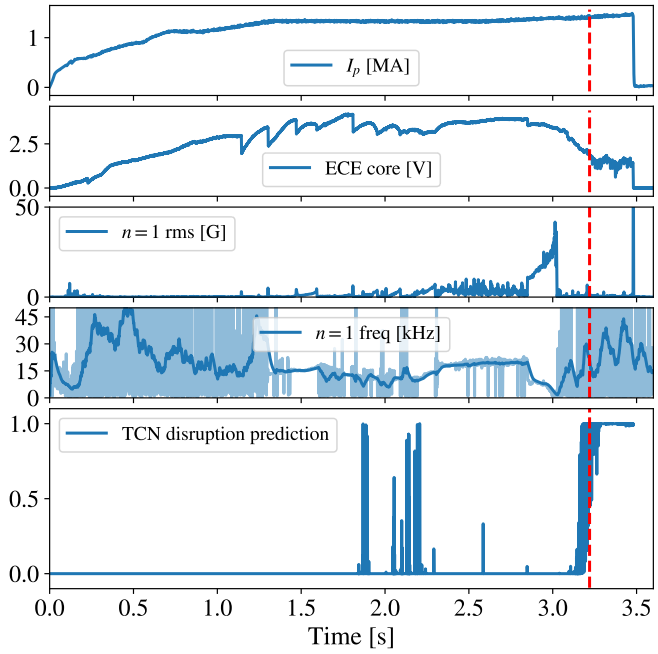


Figure 8: Time traces of plasma quantities (plasma current, I_p, an ECE core channel, magnetics measurements of the $n = 1$ mode amplitude and frequency) and the output of the TCN disruption predictor during a disruptive shot, correctly identified. Red dashed lines marks the predictor alarm time, roughly 260 ms before the disruption occurs. Shot 145018.

does suggest that additional diagnostics, or more reduced physics-based extracted features could augment the ECEi raw data in this disruption predictor, along with further training looking at more shots. This point will be further discussed in the next section.

5. Discussion and Future Work

These results show the usefulness of deep convolutional neural networks with dilated convolutions for fusion problems where the multi-scale, multi-physics nature mandates capturing long-range dependencies in time-series. The results also demonstrate that the ECEi diagnostic captures rich pre-disruption dynamics sufficient enough to by itself be useful for disruption predictions. The techniques applied here show deep learning can apply directly to raw data from diagnostics with high temporal resolution in order to make useful disruption predictions, a topic critical to the success of magnetic confinement fusion. These techniques can also be applied to various time-series sequence analysis problems, many which apply to data gathered by scientific instruments. They also show that training TCN networks with large receptive fields on the order of $\sim 30k$ is possible, allowing learning on long sequences with long-range dependencies. This initial paper presented the proof-of-principle of the technique. Comparisons using other machine learning techniques are needed to definitively show the benefit of these TCN architectures for

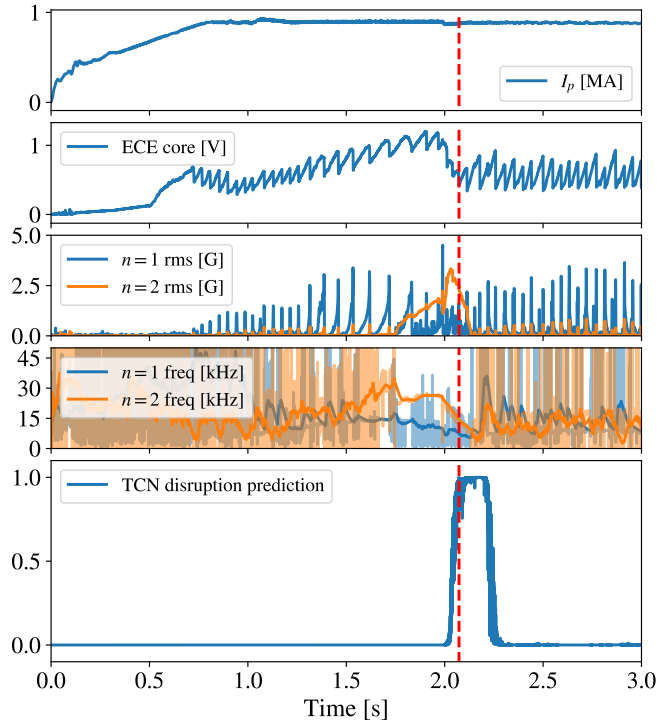


Figure 9: Time traces of plasma quantities (plasma current, I_p, an ECE core channel, magnetics measurements of the $n = 1$ mode amplitude and frequency) and the output of the TCN disruption predictor during a non-disruptive shot, incorrectly identified. This plasma appears to self-heal after a minor disruption. Red dashed lines marks the predictor alarm time. Shot 149485.

this particular application of disruption prediction, and the sole use of ECEi.

Looking forward to disruption prediction on machines such as ITER and beyond to fusion reactors, this work can help inform how to best utilize the diagnostic sets for data-driven disruption predictions, and what diagnostic coverage would be needed for accurate predictions. The results presented here motivate the installation and use of high-resolution spatiotemporal diagnostics such as ECEi in future fusion reactors, which are feasible to install but may be in different forms compared to diagnostics in current devices [40]. The TCN algorithm is quite general and can be applied to other diagnostics and/or physics-based features, expanding the temporal information used in disruption prediction. As diagnostic coverage in expected to be reduced in fusion reactors due a number of nuclear environment issues, this work could be expanded to help make the most of available diagnostics, and inform a minimal set of diagnostics needed for accurate disruption prediction. Additional lines of research related to utilization of the work presented here in future devices such as ITER are detailed below.

For future work, there are many areas which can extend and enhance the work presented here.

First, pertaining to using deep learning with the ECEi diagnostic for disruption prediction, there is a large potential to further increase the disruption prediction performance. Using the full dataset (no temporal downsampling, no undersampling of non-disruptive sequences) could further improve the results shown here. With the current computational setup, this may require a hybrid model/data parallelism in the parallelization of the training routine, where the model will be decomposed to on-node GPU's, and data parallelism will be used across nodes [41]. Varying model parameters such as the dilation factor may also be a viable solution. Also, although the TCN architectures allow training on raw data, it may be that certain pretraining transforms (e.g. short-time Fourier transforms) may be sufficient for the task at hand and can simplify the neural network architecture needed (this would be an example of human crafted features, see Section 2).

Second, combining multiple diagnostics (in machine learning parlance often referred to as multiple modalities) will most likely be mandatory for multi-physics problems such as disruption predictions. Disruptions have a number of different physics root causes, which may be more accurately predicted using diagnostics more sensitive to those physics, for example a bolometry diagnostic for impurity radiation induced disruptions. Separate neural networks could be trained on each diagnostic and combined at the end to give a prediction, but newer techniques such as feature-wise transformations offer the potential to integrate multiple diagnostics into a single network[42], allowing correlations between diagnostics to be utilized in the disruption prediction. We can also explore inputting into the neural network more physics-based features, extracted from the raw diagnostic signals, like those that have been traditionally used in disruption prediction such as the $n = 1$ mode amplitude and frequency. Care must always be taken when using processed signals that causality is not violated by filters or transforms used to extract the features.

Third, understanding how to transfer these trained models to newer machines such as ITER is critical, where not many disruptions can be tolerated [25]. Purely data-driven techniques utilizing the trained neural networks can be used, namely a technique known as transfer learning which enables learning on a small number of examples by retraining a neural network that has already been trained on a different, larger dataset from similar but not necessarily completely overlapping data distributions [43, 44]. For similar diagnostics, it may be possible to apply transfer learning for use on different machines, which can be tested now, for example by attempting to retrain the disruption predictor in this paper using the ECEi diagnostic on the KSTAR tokamak [45]. However, there are difficulties with this purely data-driven approach, as mismatches between spatial sizes, temporal sampling rates, and device physical timescales may make scaling or interpolation of data between devices impossible. Techniques which can be viewed as more hybrid, including physics information with the data-driven approach, may be necessary. One possibility for including physics constraints can come from including simulation data in retraining models, making use of synthetic diagnostics for the new machine [46]. Another possibility is to create models for plasma behavior, including disruption, and extract model parameters from existing machines and their diagnostics[47].

Fourth, interpretability of the neural network predictions is greatly desired. While these algorithms can be treated as black-boxes for certain engineering uses (one example may be disruption prediction for triggering mitigation), when extrapolating to new machines, understanding *why* the algorithm is giving a particular prediction is very beneficial. There is various research into neural network interpretability [48], including saliency methods which have been applied to, for example, neural networks on self-driving cars, to identify which pixels of an image most informed the neural network in the prediction on direction to go [49]. Caution must be used to ensure the inductive bias (i.e. selected structure) of the networks do not dominate the outputs of these methods [50]. Recent work using context activation vectors [51], a small set of samples with a domain-expert determined salient feature, have been used to identify when a neural network utilizes such features for a particular prediction. This could allow fusion physicists to isolate pre-disruption markers of interest (e.g. locked modes), and allow the neural network to output how important these markers were in its disruption prediction.

Fifth, recent upgrades to the DIII-D ECEi system give more accurate absolute electron temperature measurements, providing sharper details of modes in the plasma. System-on-chip (SoC) technology[52, 53] provides super electronics

noise suppression and outstanding shielding performance against out-of-band interference. Also, the working frequency has been upgraded into W-band (75-110 GHz), which is able to set the ECEI observation window on physically interesting regions of the pedestal or core. The new W-band SoC ECEI system has been calibrated with standard Electron Cyclotron Emission Radiometer and Thomson Scattering to provide real-time electron absolute temperature profiles. These upgrades give the promise of allowing neural networks to even more accurately capture and learn the pre-disruption dynamics.

Finally, this technique can of course be used to detect various fusion plasma phenomena, besides disruptions, which are of interest to operators and physicists. Especially in high temporal resolution diagnostics, the data is often cumbersome to manually review. Creating an "automated logbook" using a neural network trained to identify phenomena of interest could create tremendous value in helping physicists sift through the data intelligently. For situations where labelling the data may be cumbersome, self-supervised techniques can be used [54], where the network trains on a large diagnostic dataset, and attempts to predict next time points for the diagnostic. This trained model can then be retrained on a small labelled dataset of a particular phenomena in order to accurately identify other instances of the phenomena.

6. Acknowledgments

The main author would like to thank Dave Schissel, C.S. Chang, Bill Tang, Julien Kates-Harbeck, Raffi Nazikian, Cristina Rea, Bob Granetz, Neville Luhmann, Sean Flanagan, Ahmed Diallo, Brian Grierson, Nikolas Logan, and Ken Silber for various contributions to this work. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, under AC02-09CH11466, DE-FC02-04ER54698, and FG02-99ER54531. We also recognize the Princeton Research Computing center for the computational resources used in this paper. Part of the data analysis was performed using the OMFIT integrated modeling framework [31].

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.

7. Data Availability Statement

Data used to create these figure can be found in the open repository `https://dataspace.princeton.edu/jspui/handle/88435/dsp015425kd34n`. Please contact the authors for access to the complete curated ECEi dataset.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, Nature **521**, 436 (2015).
[2] Z. C. Lipton, J. Berkowitz, and C. Elkan, arXiv e-prints arXiv:1506.00019 (2015), arXiv:1506.00019 .
[3] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, arXiv e-prints arXiv:1809.04356 (2018), arXiv:1809.04356 .
[4] M. Choi, H. Park, G. Yun, W. Lee, N. Luhmann, K. Lee, W.-H. Ko, Y.-S. Park, B. Park, and Y. In, Nucl. Fusion **56**, 066013 (2016).
[5] T. Hender, J. Wesley, J. Bialek, A. Bondeson, A. Boozer, R. Buttery, A. Garofalo, T. Goodman, R. Granetz, Y. Gribov, O. Gruber, M. Gryaznevich, G. Giruzzi, S. Günter, N. Hayashi, P. Helander, C. Hegna, D. Howell, D. Humphreys, G. Huysmans, A. Hyatt, A. Isayama, S. Jardin, Y. Kawano, A. Kellman, C. Kessel, H. Koslowski, R. L. Haye, E. Lazzaro, Y. Liu, V. Lukash, J. Manickam, S. Medvedev, V. Mertens, S. Mirnov, Y. Nakamura, G. Navratil, M. Okabayashi, T. Ozeki, R. Paccagnella, G. Pautasso, F. Porcelli, V. Pustovitov, V. Riccardo, M. Sato, O. Sauter, M. Schaffer, M. Shimada, P. Sonato, E. Strait, M. Sugihara, M. Takechi, A. Turnbull, E. Westerhof, D. Whyte, R. Yoshino, H. Zohm, D. Group, the ITPA MHD, and Magnet, Nucl. Fusion **47**, S128 (2007).
[6] J. Vega, S. Dormido-Canto, J. M. López, A. Murari, J. M. Ramírez, R. Moreno, M. Ruiz, D. Alves, and R. Felton, Fusion Eng. Des. **88**, 1228 (2013).
[7] C. Rea and R. S. Granetz, Fusion Sci. Technol. , 1 (2018).
[8] J. Kates-Harbeck, A. Svyatkovskiy, and W. Tang, Nature **568**, 526 (2019).
[9] D. R. Ferreira, arXiv e-prints arXiv:1811.00333 (2018), arXiv:1811.00333 .
[10] F. I. Subcommittee, "FESAC ISOFS Subcommittee Final Report," Tech. Rep. (FES, 2002).
[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[12] L. L. Lao, H. St. John, R. D. Stambaugh, A. G. Kellman, and W. Pfeiffer, Nucl. Fusion **25**, 1611 (1985).
[13] S. Hochreiter and J. Schmidhuber, Neural Comput. **9**, 1735 (1997).
[14] S. Bai, J. Z. Kolter, and V. Koltun, arXiv e-prints arXiv:1803.01271 (2018), arXiv:1803.01271 .
[15] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, arXiv e-prints arXiv:1705.03122 (2017), arXiv:1705.03122 .
[16] Z. Wood-Doughty, N. Andrews, and M. Dredze, "Convolutions Are All You Need (For Classifying Character Sequences)," Tech. Rep. (Stroudsburg, PA, USA, 2018).
[17] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, arXiv e-prints arXiv:1609.03499 (2016).
[18] M. Farge, Annu. Rev. Fluid Mech. **24**, 395 (1992).
[19] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, in *EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.* (Association for Computational Linguistics (ACL), 2014) pp. 1724–1734, arXiv:1406.1078 .
[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, arXiv e-prints arXiv:1706.03762 (2017), arXiv:1706.03762 .
[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, arXiv e-prints arXiv:1810.04805 (2018), arXiv:1810.04805 .

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," Tech. Rep. (OpenAI, 2019).

[23] X. Wang, R. Girshick, A. Gupta, and K. He, arXiv e-prints arXiv:1711.07971 (2017), arXiv:1711.07971 .

[24] R. Child, S. Gray, A. Radford, and I. Sutskever, arXiv e-prints arXiv:1904.10509 (2019), arXiv:1904.10509 .

[25] P. C. de Vries, G. Pautasso, D. Humphreys, M. Lehnen, S. Maruyama, J. A. Snipes, A. Vergara, and L. Zabeo, Fusion Sci. Technol. **69**, 471 (2016).

[26] B. Tobias, C. W. Domier, T. Liang, X. Kong, L. Yu, G. S. Yun, H. K. Park, I. G. J. Classen, J. E. Boom, A. J. H. Donné, T. Munsat, R. Nazikian, M. Van Zeeland, R. L. Boivin, and N. C. Luhmann, Rev. Sci. Instrum. **81**, 10D928 (2010).

[27] I. H. Hutchinson, *Principles of Plasma Diagnostics* (Cambridge University press, 2002).

[28] S. K. Rathgeber, L. Barrera, T. Eich, R. Fischer, B. Nold, W. Suttrop, M. Willensdorfer, E. Wolfrum, and t. A. U. Team, Plasma Phys. Control. Fusion **55**, 025004 (2013).

[29] A. H. Boozer, Phys. Plasmas **19**, 058101 (2012).

[30] Y. Fu, D. Eldon, K. Erickson, K. Kleijwegt, L. Lupin-Jimenez, M. D. Boyer, N. Eidietis, N. Barbour, O. Izacard, and E. Kolemen, Phys. Plasmas **27**, 022501 (2020).

[31] O. Meneghini, S. Smith, L. Lao, O. Izacard, Q. Ren, J. Park, J. Candy, Z. Wang, C. Luna, V. Izzo, B. Grierson, P. Snyder, C. Holland, J. Penna, G. Lu, P. Raum, A. McCubbin, D. Orlov, E. Belli, N. Ferraro, R. Prater, T. Osborne, A. Turnbull, and G. Staebler, Nucl. Fusion **55**, 083008 (2015).

[32] A. Ng, "Normalizing inputs - Practical aspects of Deep Learning — Coursera," (2017).

[33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, in *NIPS-W* (2017).

[34] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, arXiv e-prints arXiv:1706.02677 (2017), arXiv:1706.02677 .

[35] M. Buda, A. Maki, and M. A. Mazurowski, Neural Networks **106**, 249 (2018).

[36] K. Montes, C. Rea, R. Granetz, R. Tinguely, N. Eidietis, O. Meneghini, D. Chen, B. Shen, B. Xiao, K. Erickson, and M. Boyer, Nucl. Fusion **59**, 096015 (2019).

[37] (ACM, 2019).

[38] R. Sweeney, W. Choi, M. Austin, M. Brookman, V. Izzo, M. Knolker, R. L. Haye, A. Leonard, E. Strait, F. Volpe, and T. D.-D. Team, Nucl. Fusion **58**, 056022 (2018).

[39] G. Kim, G. S. Yun, M. Woo, and t. K. Team, Plasma Phys. Control. Fusion **61**, 055001 (2019).

[40] B. Tobias, A. Donné, H. Park, J. Boom, M. Choi, I. Classen, C. Domier, X. Kong, W. Lee, T. Liang, N. Luhmann, T. Munsat, L. Yu, and G. Yun, Contrib. to Plasma Phys. **51**, 111 (2011).

[41] T. Ben-Nun and T. Hoefler, arXiv e-prints arXiv:1802.09941 (2018), arXiv:1802.09941 .

[42] V. Dumoulin, E. Perez, N. Schucher, F. Strub, H. Vries, A. Courville, and Y. Bengio, Distill **3**, e11 (2018).

[43] S. J. Pan and Q. Yang, IEEE Trans. Knowl. Data Eng. **22**, 1345 (2010).

[44] S. Ruder, "Transfer Learning - Machine Learning's Next Frontier," (2017).

[45] G. S. Yun, W. Lee, M. J. Choi, J. B. Kim, H. K. Park, C. W. Domier, B. Tobias, T. Liang, X. Kong, N. C. Luhmann, and A. J. H. Donné, Rev. Sci. Instrum. **81**, 10D930 (2010).

[46] K. D. Humbird, J. L. Peterson, and R. G. McClarren, arXiv:1812.06055 (2018), arXiv:1812.06055 .

[47] V. Mehta, I. Char, W. Neiswanger, Y. Chung, J. Schneider, A. O. Nelson, M. D. Boyer, and E. Kolemen, in *Work. Integr. Deep Neural Model. Differ. Equations (ICLR 2020)* (2020).

[48] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, arXiv e-prints arXiv:1806.00069 (2018), arXiv:1806.00069 .

[49] M. Bojarski, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, U. Muller, and K. Zieba, arXiv e-prints arXiv:1611.05418 (2016), arXiv:1611.05418 .

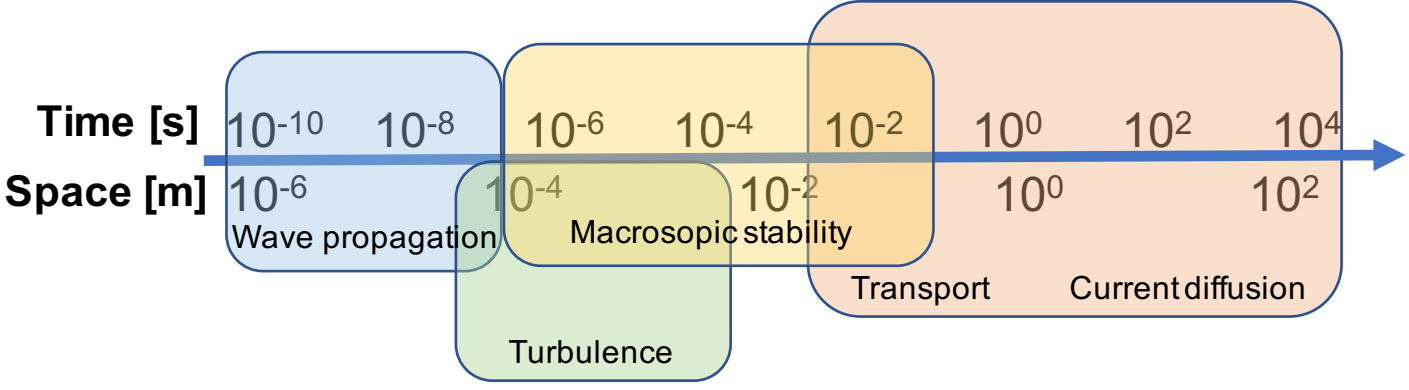
[50] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, arXiv e-prints arXiv:1810.03292 (2018), arXiv:1810.03292 .

[51] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, arXiv e-prints arXiv:1711.11279 (2017), arXiv:1711.11279 .

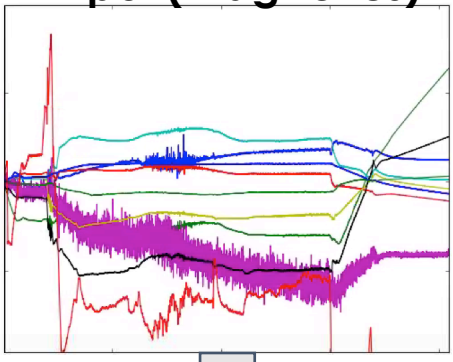
[52] Y. Zhu, Y. Ye, J. H. Yu, B. Tobias, A. V. Pham, Y. Wang, C. Luo, C. W. Domier, G. Kramer, Y. Ren, A. Diallo, R. Nazikian, M. Chen, G. Yu, and N. C. Luhmann, Rev. Sci. Instrum. **89** (2018), 10.1063/1.5035373.

[53] Y. Zhu, J. H. Yu, M. Chen, B. Tobias, and N. C. Luhmann, IEEE Trans. Plasma Sci. **47**, 2110 (2019).

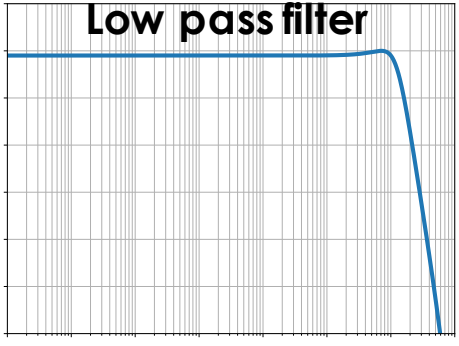
[54] S. Schneider, A. Baevski, R. Collobert, and M. Auli, arXiv e-prints arXiv:1904.05862 (2019), arXiv:1904.05862 .


Time [s]
Space [m]
10-10
10-8
10-6
10-6
10-4
10-4
10-2
10-2
100
100
102
102
104
Wave propagation
Turbulence
Macrosopic stability
Transport
Current diffusion

Input (magnetics)



Low pass filter

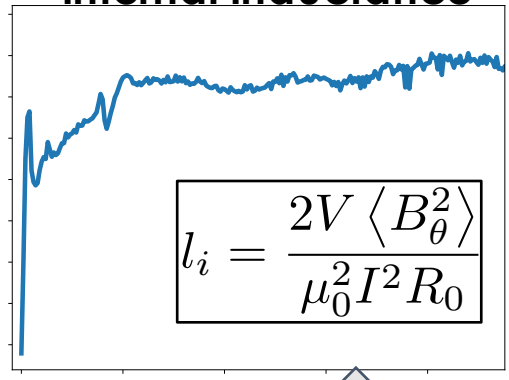


EFIT

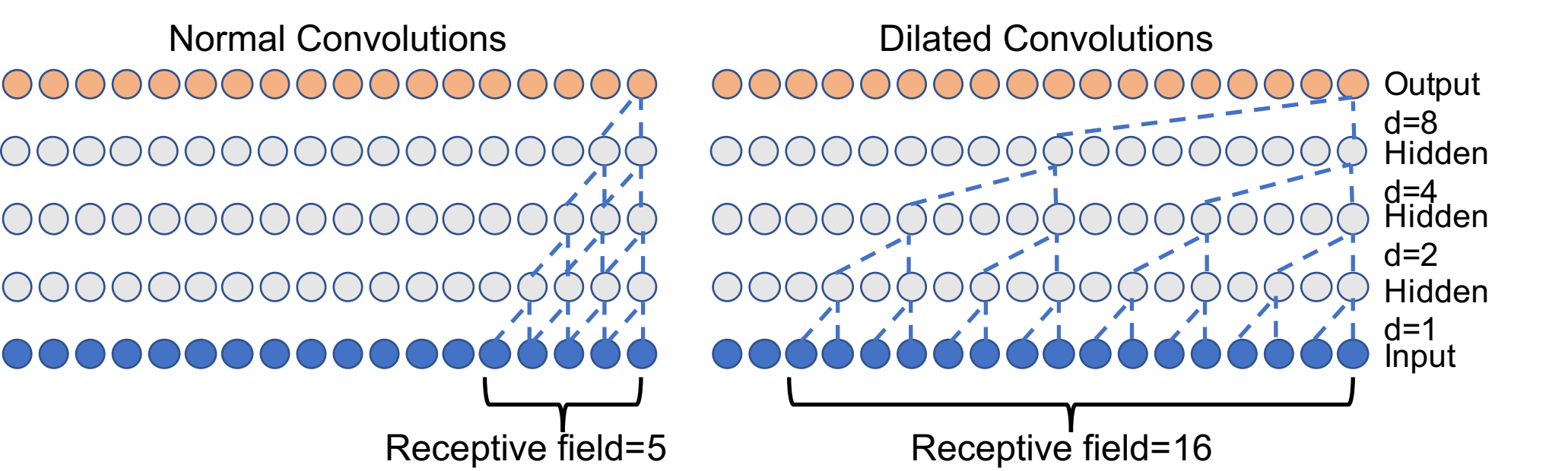
$$\Delta^* \psi = -\mu_0 \mathrm{R} \mathrm{J}_\mathrm{T}$$

$$\mathrm{J}_\mathrm{T} = \mathrm{R}\left[\mathrm{P}'(\psi;\alpha_\mathrm{n}) + \frac{\mu_0 \mathrm{F}\mathrm{F}'(\psi;\gamma_\mathrm{n})}{4\pi^2 \mathrm{R}^2}\right]$$

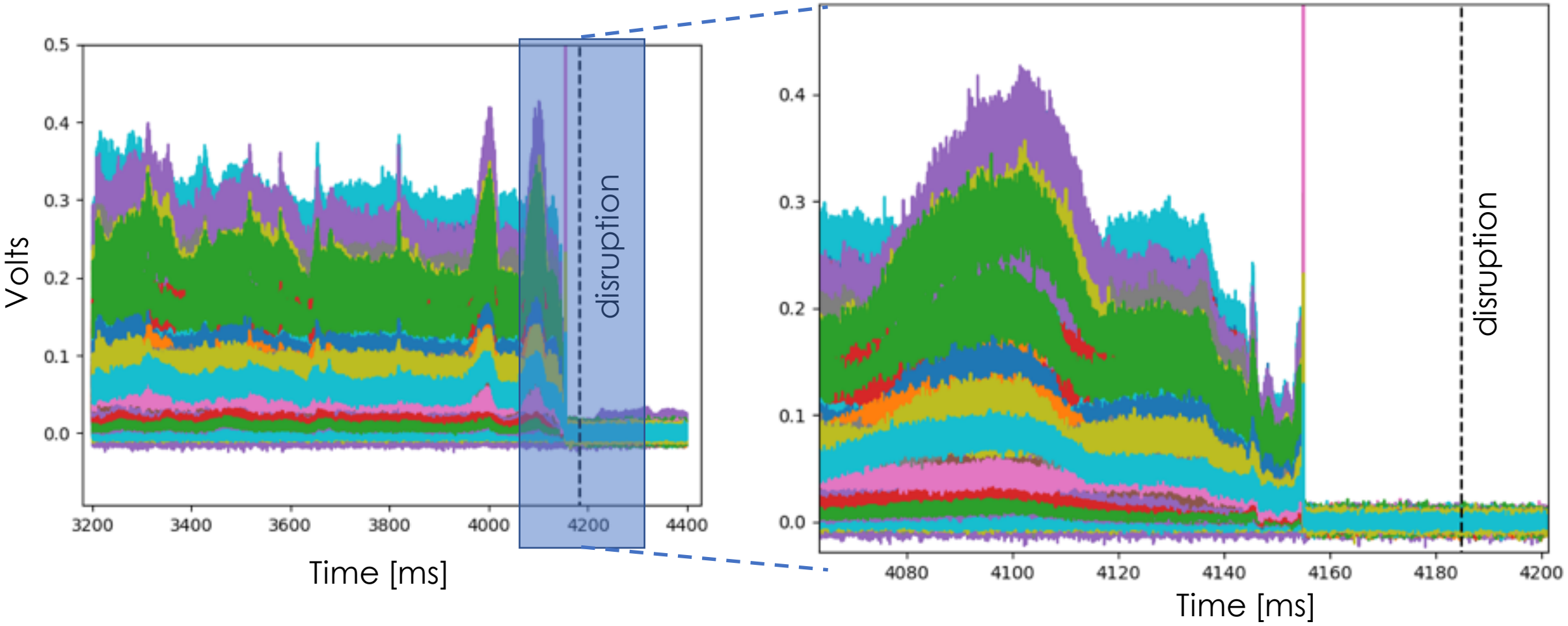
Internal Inductance

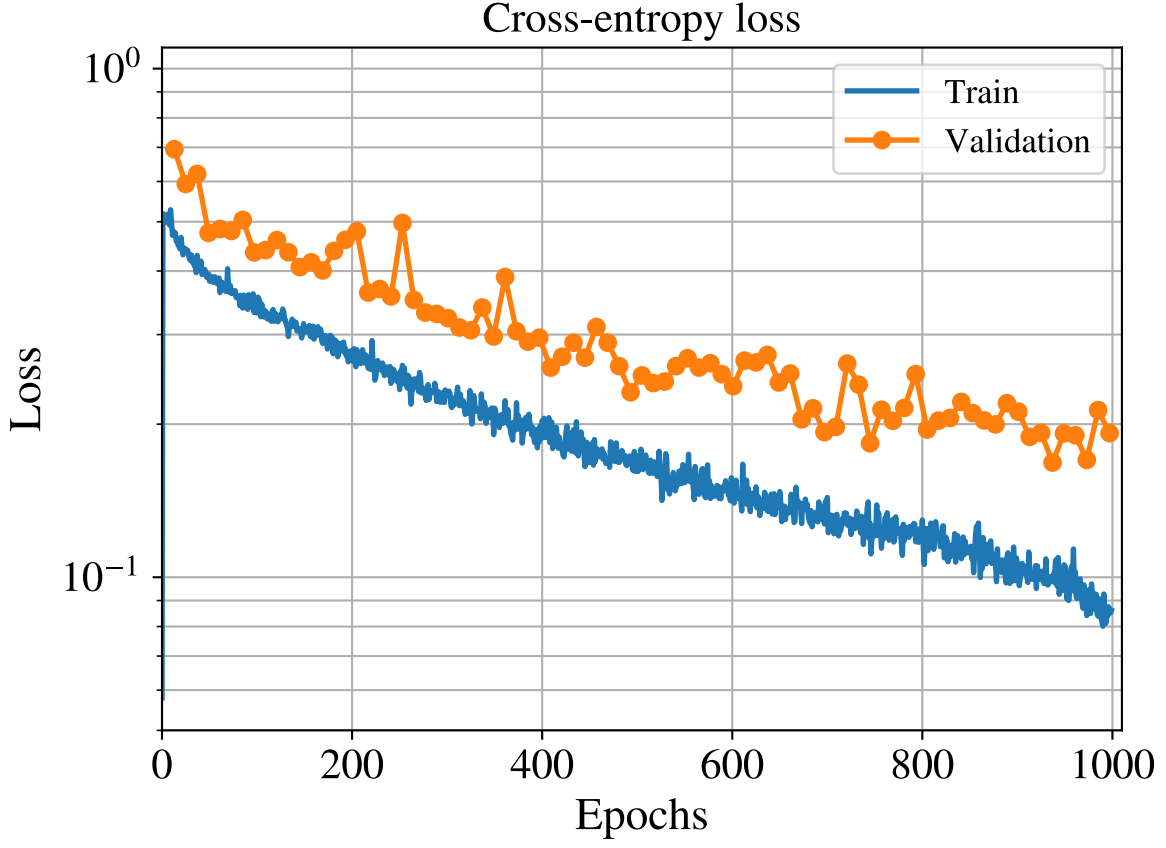


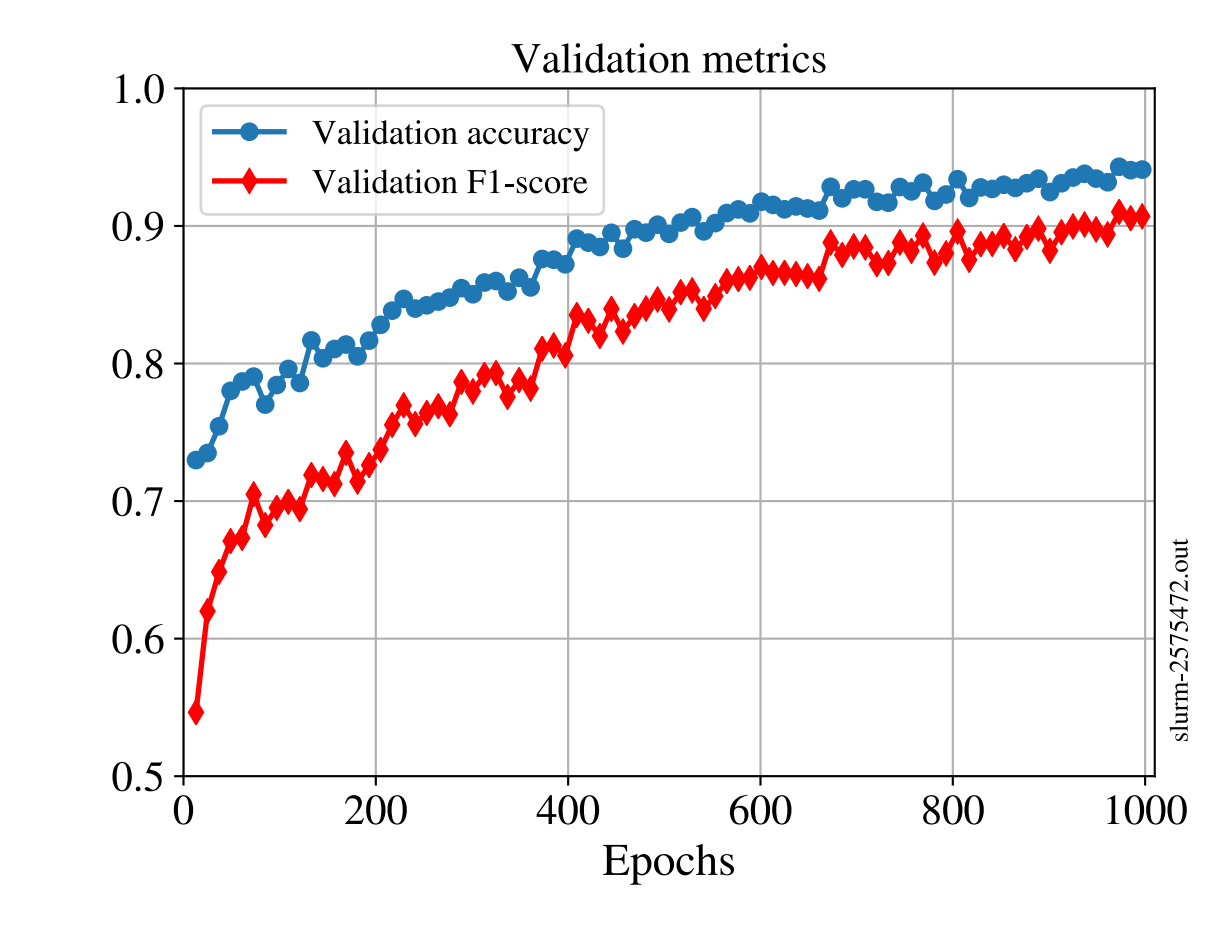
$$l_i = \frac{2V\left\langle B_\theta^2 \right\rangle}{\mu_0^2 I^2 R_0}$$

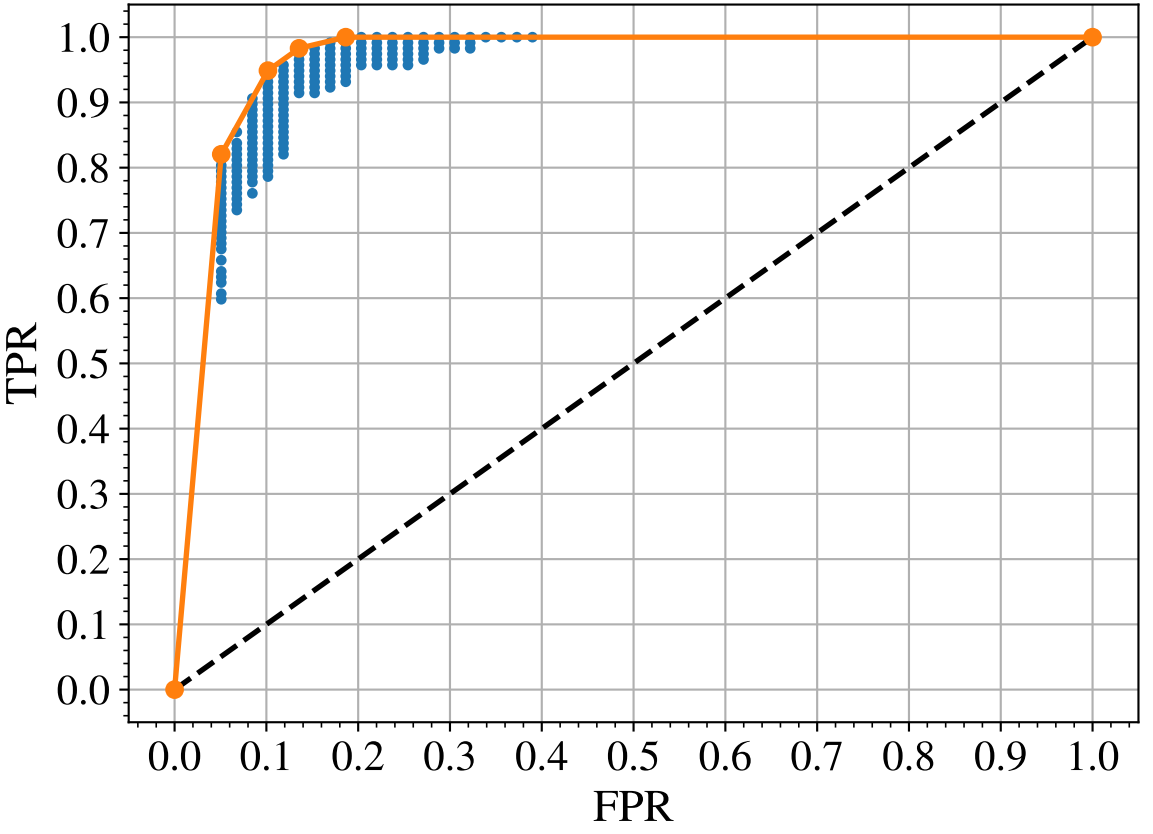

Normal Convolutions
Dilated Convolutions
Output
d=8
Hidden
d=4
Hidden
d=2
Hidden
d=1
Input
Receptive field=5
Receptive field=16

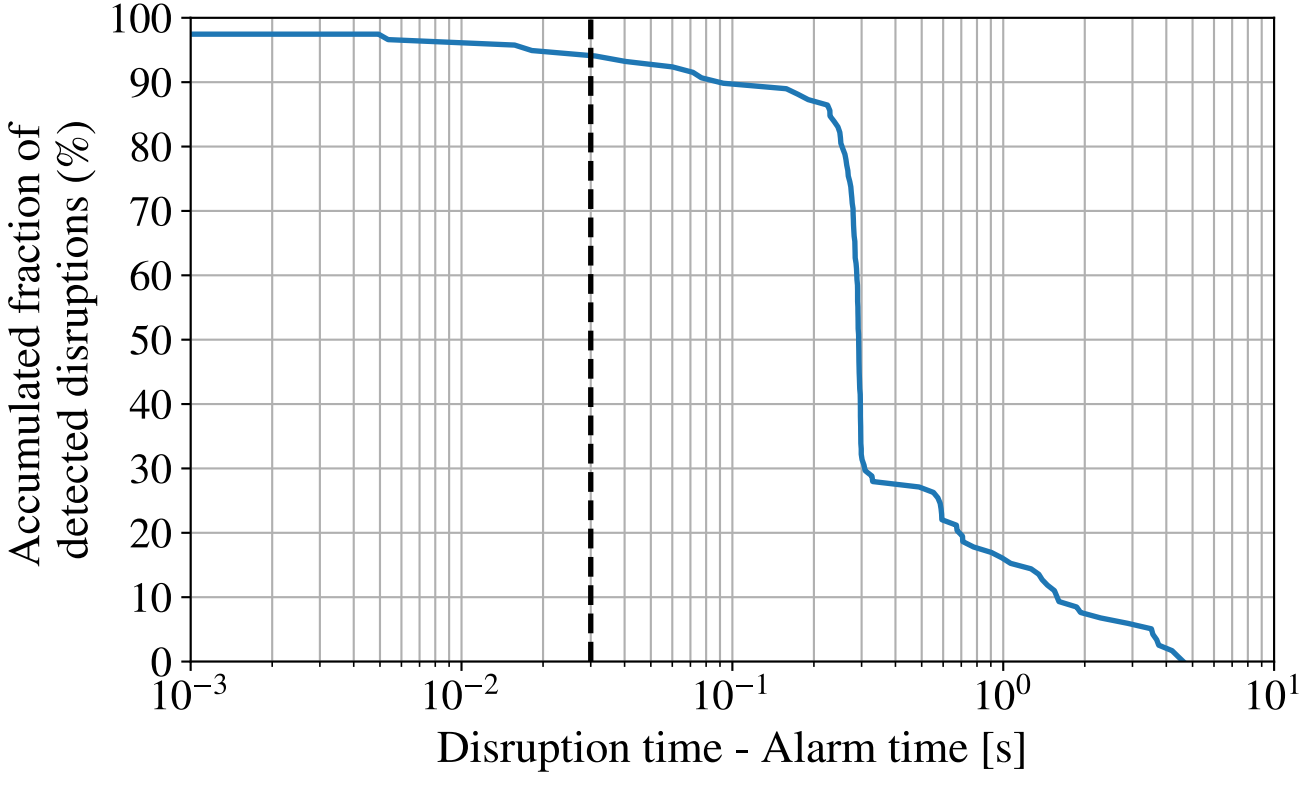
ECEi data near disruption

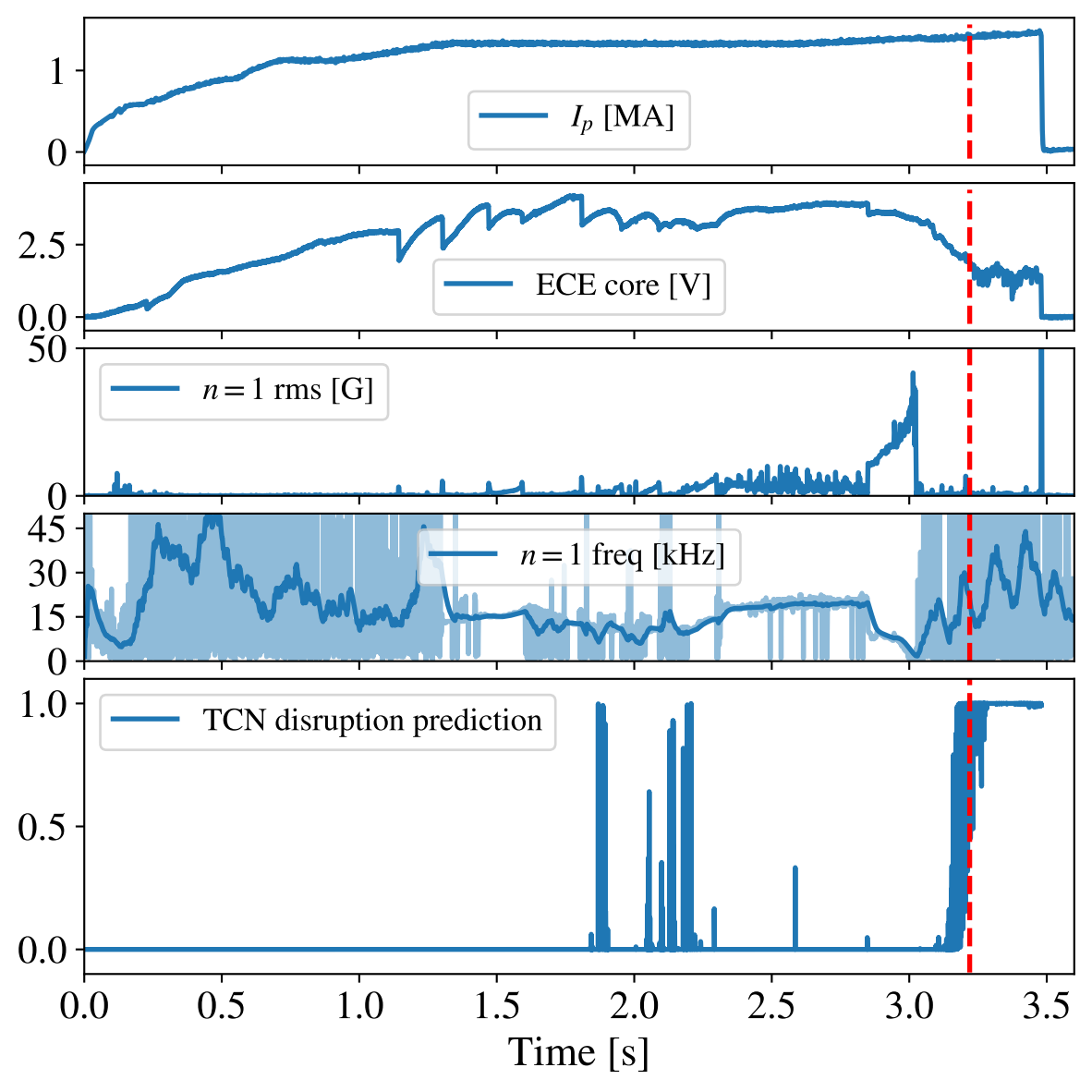


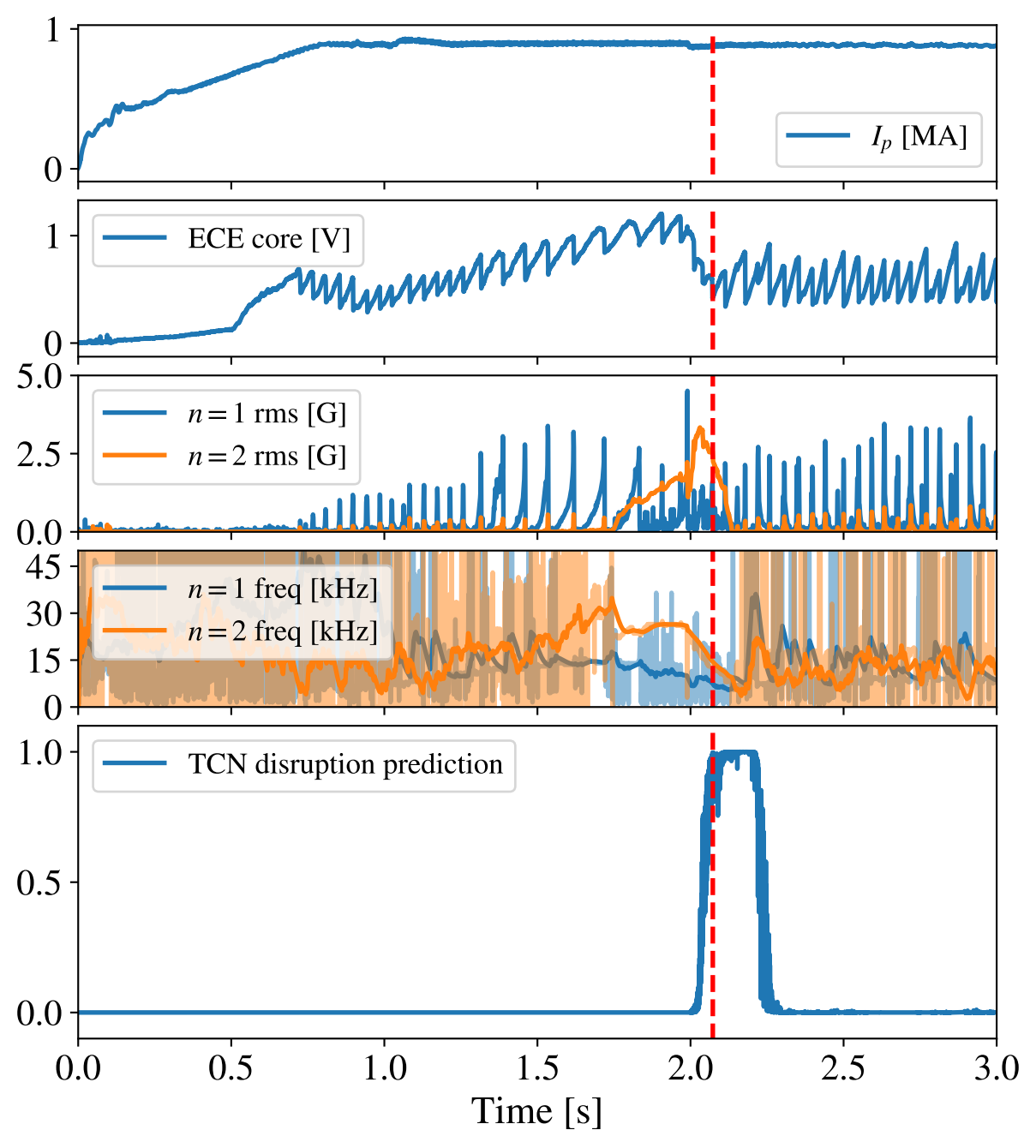

Cross-entropy loss
Train
Validation
10^0
10^{-1}
Loss
0
200
400
600
800
1000
Epochs


Validation metrics
Validation accuracy
Validation F1-score
1.0
0.9
0.8
0.7
0.6
0.5
0
200
400
600
800
1000
Epochs
slurm-2575472.out


TPR
FPR
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0


Accumulated fraction of detected disruptions (%)
100
90
80
70
60
50
40
30
20
10
0
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Disruption time - Alarm time [s]


I_p [MA]
ECE core [V]
$n=1$ rms [G]
$n=1$ freq [kHz]
TCN disruption prediction
Time [s]


1
0
I_p [MA]
1
0
ECE core [V]
5.0
2.5
0.0
$n=1$ rms [G]
$n=2$ rms [G]
45
30
15
0
$n=1$ freq [kHz]
$n=2$ freq [kHz]
1.0
0.5
0.0
TCN disruption prediction
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Time [s]