A DEEP LEARNING APPROACH TO FORECASTING TRANSIENT STABILITY IN POWER GRIDS

by

Oliver Paul

September 2020

A Dissertation submitted in part fulfilment of the Degree of Master of Science in Energy Systems and Data Analytics

The institute for Energy
Bartlett School of Environment, Energy and Resources
University College London

In partnership with
Hitachi ABB Power Grids

Word count: 9978

Contents

List of Figures

List of Tables

Abstract

Online monitoring in power systems is gaining increasing attention as grids become more complex, and as the consequences of power outages become ever more significant. Rotor angle stability is of critical importance in stability monitoring, and is often the first component of power system instability to initiate a critical cascading event. This thesis benchmarks the performance of a number of deep learning architectures for online forecasting of rotor angle trajectories, with a focus on LSTM implementations. Each model is formulated as a multi-variate, multi-step, multi-label predictor where all system rotor angle trajectories are forecast in a single shot by one model. This study shows that such a formulation can effectively forecast rotor angle trajectories up to forecasting periods of 2 seconds and 200 time steps using features from typical PMU data. A number of considerations where identified which currently limit the scope of these models in real-world settings which should inspire future work. These include a high sensitivity to noise, and a limited ability to capture the relationship between system dynamics and topology though PMU features only pertaining to the systems voltage-current phasor relationship.

Acknowledgements

I would like to thank my dissertation supervisor Aidan O'Sullivan for his support during this thesis period. I would also like to thank him for igniting my passion for data science and machine learning. At ABB Power Grids, I would like to thank Alexandre Oudalov for supporting me over the summer and for helping me get to grips with the many complexities of modern power systems.

1 Introduction

1.1 The big picture

System reliability is a primary consideration for electricity transmission and distribution operators, which describes the ability to provide continuity of electric service to customers. This requires the system to continuously supply adequate generation to meet demand, and react dynamically to both random and routine permutations within the system. Failure to maintain this balance can lead to disastrous consequences in the form of blackouts. Even short term outages can result in significant economic damage, both to energy suppliers in the form of maintenance and lost revenue, but also to end users through loss of earning or contingency expenses. The US Department of Energy estimates that power outages may be costing the US economy $150 billion annually (Hussain 2018). For further illustration, Table 1 summarises the approximate costs of some historical blackout events (Royal Academy 2014).

Location	**Date**	**Cause**	**People affected**	**Duration**	**Cost**
Canada / Northeast US	August 2003	technical fault / human error	50 million	1 - 4 days	$4.5 billion - $8.2 billion
California	2000/2001	capacity crisis	1.5 million	rolling 1 year	$40 billion
Europe	2006	network failure	20 countries	up to 2 hrs	$100 million
Italy / Switzerland	2003	network failure	56 million	up to 19 hrs	1,182 million

Table 1: Significant blackout events in Europe and North America

Beyond the economic consequences of blackouts, the social cost can be critical. For a customer using a television or laptop a power outage could be considered a mere inconvenience; conversely, for a customer connected to a dialysis machine such an incident could be a matter of life and death.

Data from blackouts in the USA between 2008 and 2015 show an increase in the number of power outages outages by 64%, and a strong correlation between the number of outages and frequency of irregular weather conditions, with California accounting for 25% of all outages during this period (Haes Alhelou et al. 2019). Indeed, weather events are the leading cause of blackouts worldwide, followed closely by human error or equipment failure events. Such risks should be expected to increase due to climate change, with the consequences of outages ever increasing due to an increased rate of urbanisation and growing electricity demand in the developing world. Further, frontier risks in the form of cyber attacks are becoming a reality, and will require new data driven approaches to tackle, given the nature of blackouts in the past were mostly hardware related, often caused by equipment faults from fallen trees, animals or vehicle accidents.

Beyond external risks, the energy transition is driving unprecedented change in the power grid. Driven largely by the need for decarbonisation, and increased geographical interconnection, power grids are becoming more complex and increasingly unpredictable. Some factors contributing to this increased complexity are more renewable energy sources (RES) being embedded in local distribution systems, electric vehicles, heat pumps, high voltage DC interconnectors and smart grids with their associated control systems.

1.2 Power grid stability

Power system stability can be defined as *" the ability of an electric power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with most system variables bounded so that practically the entire system remains intact"*

(Kundur et al. 2004), and is governed by three elements; rotor angle, frequency and voltage stability. Blackouts do not occur instantaneously following a disturbance (often referred to as a contingency), but occur as the result of a sequence of events, known as cascades. These events can occur over a period of several seconds to minutes, and following an initial contingency are usually the result of grid protection systems tripping a piece of equipment to protect it from damaging conditions.

Frequency instability is most directly associated with power outages and is directly related to the balance between supply and demand in the system. The most common scenario is under frequency, where demand exceeds supply. This could be the result of multiple trippings in the network, disconnecting generation from load centres. The following figure illustrates a typical frequency profile during a real-world blackout event which took place in Argentina and Uraguay in June 2016.

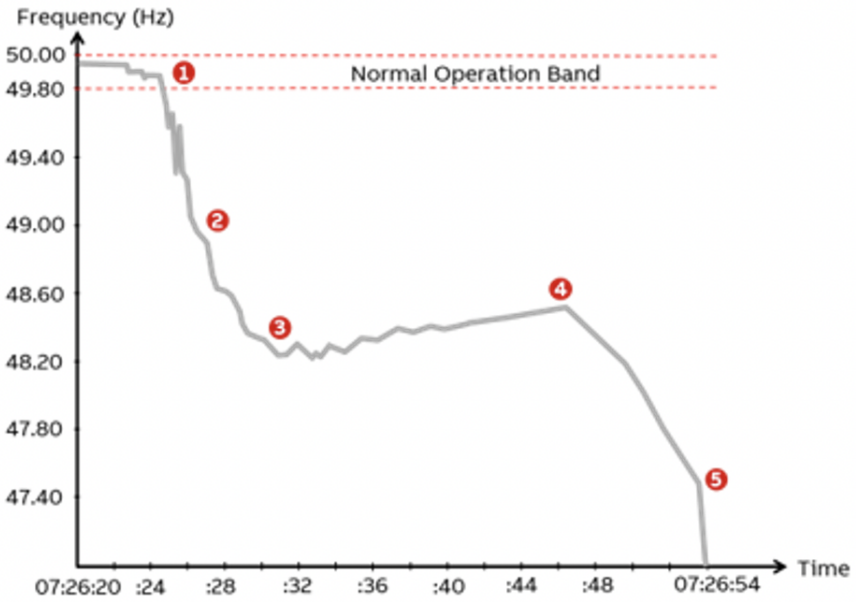

Figure 1: Chronological development of real-world system blackout - ABB Power Grids

1. A single phase fault is caused on a single line, the exact cause is unknown, but possibly related to weather conditions.
2. DAG scheme did not function as expected leaving 1,200 MW of excessive generation in the northeast. A loss of synchronisation is observed between the northeast and rest of Argentina.
3. Nuclear and thermal power plants are tripped.
4. The remaining generators in the system ramp up the frequency.
5. Low frequency protection systems trip synchronous machines, causing frequency collapse.

Whilst system frequency is a primary feature of interest with respect to large scale blackouts, it is often a secondary consequence of rotor angle instability (or transient instability), which can typically occur within a matter of hundredths of a second from a fault clearance. The speed by which transient instability can occur makes these events particularly difficult to anticipate. Transient stability is concerned with the ability of all synchronous machines in the system to remain in synchronism following a disturbance. In the case that one machine runs faster than another the angular position of the rotor of the faster machine will advance on the slower one and transfer some load from the slower machine. Following such a disturbance, the system will remain transient stable if these oscillations diminish and all machine angles return to a new steady state. In a transient unstable case, angular swings of one or more machines will increase until they are tripped to avoid pole slipping (generally a machines

relative rotor angle is not allowed to exceed 180 degrees). It is highly challenging to anticipate rotor angle behaviour following a contingency since it is dependent on the initial operating conditions of the system (which change in real time), and the severity of the disturbance (which cannot be anticipated). However, having some indication of how rotor angle trajectories would evolve following a contingency could be greatly beneficial in preventing subsequent cascades. For example, in the case highlighted in *figure 1*, knowing which machines would lose synchronisation and trip, even a few hundredths of a second before the event, could allow for pre-emptive interventions to take place such as load shedding to protect vulnerable areas.

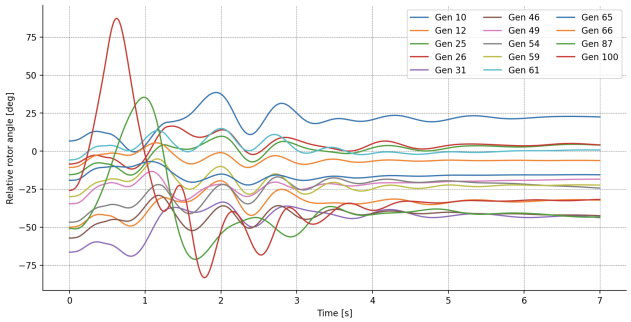

Figure 2: Three phase fault on line 100-103 clearing time 0.3 seconds - Transient stable

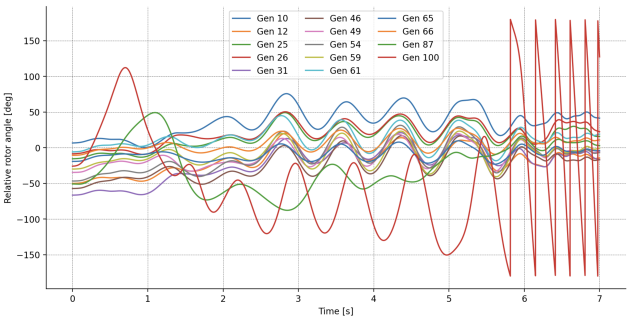

Figure 3: Three phase fault on line 100-103 clearing time 0.4 seconds - Transient unstable

Figures 2 and 3 illustrate transient stability for the IEEE 118 bus model used in this study. In the stable case generator 100 has a strong first angular swing, and whilst it does cause a momentary critically stable separation from generator 31 of 145 degrees, the swing does not cause a loss of synchronism within the system and all machine begin to return to a new steady state after 2 seconds. In the unstable case, generator 100 begins a trajectory toward a loss of synchronisation after the first swing, and pole slips as the angle passes 180 degrees after 5.8 seconds. This type of trajectory is what this study aims to monitor and forecast in real time.

1.3 Motivating machine learning in transient stability analysis

Any contingency can be simulated offline using dynamic power system software packages such as in PowerFactory or Matlab Simulink. These simulations are, however, computationally expensive, typically involving thousands of differential equations solved through numerical integration. Given the short time scales of interest with respect to transient instability, it is not feasible to to explicitly calculate dynamic behaviour in real time.

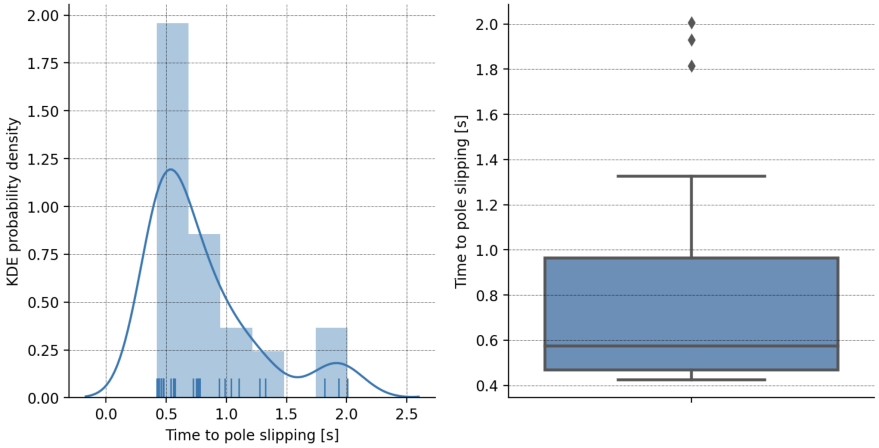

Figure 4: Time to critical pole slipping events (rotor angle passes 180 degrees) for IEEE 118 bus system with 26% wind penetration - sample size 30

As seen in *figure 2*, the majority of pole slipping events took place following the primary rotor angle swings on a time scale of approximately 0.5 seconds following fault clearing, extending to 2 seconds for secondary and tertiary swings. Any attempt to to forecast these trajectories would need to make estimates fast (in the order of milliseconds) with only a short window of time series data available. Further, explicit rotor angle data is not typically available in real-time operations, but is rather derived from signals recorded by Phasor Measurement Units (PMUs) or Wide Area Monitoring Systems (WAMS).

Haes Alhelou et al. summarise some key research gaps related to blackout risk reduction. Some gaps which could be filled by machine learning research include:

- Considering the real-time monitoring technologies for improving decision-making in the emergency situations in power systems.
- Investigating the ability of wide-area measurement systems in improving the reliability and security of the modern power system.
- Proposing new protective methods that make use of wide area measurement system (WAMS) technologies.
- Increasing the protection against the cyber-attack issues.

Indeed, it is possible that machine learning, and a data driven approach could fill a number of these gaps, particularly with respect to online monitoring of signals recorded by PMU's in the grid. Such methods would need to recognise patterns in multiple signals related to stability and consistently provide accurate trajectories within adequate time frames.

1.4 Research questions

This thesis aims to determine whether machine learning can be appropriately used to forecast rotor angle trajectories in power grids, and hence provide an online monitoring tool for transient stability. Whilst transient stability is only one element of power system stability, it is usually the first cause of cascading events. Should machine learning be an effective tool in this case, such methods could be extended to online monitoring for various other blackout related features of power systems. The research questions to be fulfilled by this thesis are detailed below:

- Can a single machine learning model forecast the trajectories for all generator rotor angles in the grid, formulated as a multi-input, multi-label, multi-step sequence problem? How does such a model compare in performance to a naive baseline, and existing works in literature?
- How does performance compare between models trained trained on pre-calculated rotor angle data as inputs and models trained only on features pertaining to PMU signals?
- Can such a model provide an estimate of confidence or uncertainty in time series predictions? What implications does this have on model validation and real-time operations?
- What are the limitations of such a model in terms of look ahead forecasting time, noise or interference, and can such a model detect false data possibly the source of manipulation or cyber attacks? What are the implications of such limitations in real-world applications?

2 Literature Review

Current literature related to machine learning applications in blackout analysis is mostly split between two approaches to stability modelling. The first, where much of the machine learning research to date has been focused, is with respect to steady state power system stability. Significantly less machine learning research is related to dynamic or transient stability, possibly related to the added complexity in modelling dynamic systems. This is of cause for concern with respect to real-world applications of machine learning for blackout mitigation. One of the first publications on power system stability in 1937 highlights that in general, the transient stability limit of a system is considerably lower than the steady state limit, and hence of greater importance in practice (unknown 1937).

One such research paper by Shuvro et al. (2019) uses a steady state approach to assess a number traditional machine learning algorithms to predict cascading events in the IEEE 118 bus model, quoting an accuracy over 0.91 for Support Vector Machine and Random Forest models. The authors use MatPower to generate a large dataset using DC power flow. In the case of non-convergence, a line becoming overloaded or a steady state rotor angle being larger that 360 degrees, such an example is labelled as a system failure and models are trained to predict such events. Such an exercise is useful; since dc power flow is far less expensive than dynamic simulations, large datasets can be produced in a short amount of time. Further since such a dataset will contain features pertaining to load flow, models may be able to learn the relationship between potential angular instability events network topology. The practical limitations of such an approach, however, is that important transients may be missed between steady states, and without accounting for the dynamics of the system, new steady states may be invalid.

Beyond the above cited work, numerous articles, including (Crucitti et al. 2004) and (Rohden et al. 2016) model blackout related failures as event-triggered sequences of steady-states via power flow equations. The applicability of such approaches within the context of real world scenarios should be called into question. Highlighted by Schäfer et al. (2018) real-world blackout events are not only defined by network topology and the static distribution of electricity flow, but more critically the collective transient dynamics of the entire system. Schäfer stresses that whilst large blackout events are the culmination of many cascading events, their causes, or triggers are fast acting transient dynamics which are often overlooked in research.

Such a notion highlights the relevance of online monitoring in power systems, particularly with respect to dynamic transients. Two of the earliest research examples on machine learning for such monitoring purposes are by Rovnyak et al. (1994) and Amjady & Majedi (2007). Both early works highlight two major challenges. The first is long algorithm training time, particularly when scaled to large systems, the second is the limited availability of PMU's in the grid which implies online monitoring in real-world application has limited scope. For this reason, both studies use directly derived rotor angles from simulation packages as model inputs, rather than signals available in real-time operations. Since the publication of these works, advances in computing power is making training complex machine learning models more viable in complex systems such as power grids. This, and the issue of PMU availability is addressed by Yu et al. (2018), with the cost of PMU's reducing it is now not unrealistic to assume PMU's are available at all generator terminal buses. Looking at a real-world example in one of the worlds fastest growing power grids, the Power Grid Corporation of India (PGCIL) plan to install PMU's at all substations and generating stations in the country (Jai 2017). Like most works in estimating transients as a sequence problem, Yu et al. (2018) formulate the problem as a classification exercise, where their machine learning model predicts the Transient Stability Assessment (TSA) of the system. That is their model predicts unstable if n is below a given threshold, where $n = (360 - \delta_{max})/(360 + \delta_{max})$ where δ_{max} is the maximum rotor angle separation of any two machines in the system. A similar approach was employed by Zhang et al. (2015), and both publications claim a high degree of accuracy in their classifications (typically above 98% for the IEEE 50 bus system).

However, in both cases, the problem is formulated as a sequence of steady-states, and hence may not reflect the true dynamics of the system. Further, this author would argue that treating transient stability assessment as a binary classification problem does not yield enough information to be acted upon in modern protection systems. Rather, the problem should be treated as a regression exercise, where the trajectories of each generator is predicted for the entire system.

A recent paper (Liu et al. 2019) formulates the problem in such a way using a Long-Short-Term-Memory (LSTM) architecture on the IEEE 39-bus system. Their model uses PMU based data as input features forecasting 200 time steps ahead based on an input window of 20 time steps. Their results show promising accuracy, however, a possible drawback to their approach is that they have trained a single LSTM model for each generator in the system, which is a high computational burden. The 39-bus model only has 10 generators, so such an approach may not scale well to larger systems. Their model also uses a large number of input features pertaining to rotor angle for each model, which increases the computational burden on data generation. Whilst this work provides a useful benchmark, there is no probabilistic interpretation to their model forecasts. Ideally an estimate of uncertainty would be available, which could in turn be incorporated into the grids protection schemes.

There is a great deal of diversity in features used as inputs to machine learning models to forecast rotor angle trajectories or estimate TSA. Yu et al. (2018) use only voltage phasor angles, whereas Liu et al. (2019) use 21 PMU variables for each machine in the system. Using the $TAN(\delta)$ method, it has been shown that rotor angles can be calculated in real-time using three features of current magnitude, voltage magnitude and phase angle between current and voltage for each respective terminal bus (Heidary 2018). This study benchmarks the results from the $TAN(\delta)$ method against rotor angle results from dynamic RMS simulations in PowerFactory and finds a percentage error typically within the range of 0.1%.

Most machine learning research with respect to rotor angle or transient stability estimation, including the works cited above use the TSA to infer the transient stability status of the system. However, in almost all cases they treat δ_{max} as unbounded having a range [0, ∞], therefore the output of $n = (360 - \delta_{max})/(360 + \delta_{max})$ becomes negative if the absolute difference between any two machines passes 360 degrees. At least from a power system perspective, this interpretation is wrong. Since dealing with relative rotor angles, the correct range for any two angular difference in the system is [-180, 180]. If pole slipping occurs, this behaviour is usually framed as the number of slips, not the displacement from a given reference point (in reality, pole slipping is not allowed). This interpretation by machine learning researchers may be in an effort to feed more easily interpretable signals to machine learning models, without trigonometric transformations, cyclic input features can be a challenging. A similar example would be a time feature, a machine learning model has no context that 23:59 is close to 00:00 without a sin/cosine transformation. In a study pertaining to transient stability analysis in high wind scenarios (Jimenez 2017), the author correctly defines the range of relative rotor angles as [-180,180], and using the TSA equation above defines three risk zones: unstable ($n \leq 0.4$), critically stable ($0.4 < n \leq 0.6$) and stable (> 0.6). Jimenez highlights the importance of using a stability metric, since it provides a measure of stability of the system as a whole, but also points out that such a metric can show the system to be stable, yet not identify a dangerous separation, or islanding between machines once returned to steady state conditions following a contingency. This is further justification for forecasting the rotor angle trajectories, whilst simultaneously providing a stability metric from these predictions, given prediction accuracy is within an acceptable margin.

To the best of the authors knowledge, there are no examples of forecasting system wide rotor angles using a single model, formulated as a multi-variate, multi-step, multi-label problem. Such a formulation makes the problem more complex, since the model need to learn the sequential relationship between all machines in the system. Whilst not related to power systems, a study in forecasting bus trip times for all links in a city (Petersen et al. 2019), uses a deep learning model to achieve a

similar outcome. The author finds that combining LSTM models (popular in rotor angle forecasting) with Convolutional Neural Network (CNN) architectures effectively extracts the temporal patterns in input time series data, for a multi-variate, multi-label problem. Such an approach could be relevant in the case of this study, since we are predicting relative rotor angle trajectories with inputs and outputs spatially and temporally correlated. The question of incorporating uncertainty estimates in time series forecasting is under explored, and remains an open question in literature. Uber conducted a study on forecasting extreme events (Zhu & Laptev 2017), using a similar architecture to Petersen, but trained their model using a series of dropout layers with a given dropout probability. Traditional Bayesian models estimate the posterior distribution of a networks weight parameters, rather than a point forecast. Uber point out that due to the non-conjugacy of deep neural networks, exact posterior inference is rarely available. Instead they propose Monte-carlo dropout (Gal & Ghahramani 2015), where predictions are made with active dropout layers, providing a distribution of estimates. With this distribution two things can be achieved, first, by taking the distribution mean, typically model performance is increased. More relevant to this study, the variance of the distribution can be used to infer an uncertainty estimate.

Most works on forecasting or estimating transient stability conclude their studies with quoted test set accuracy. These results provide useful benchmarks for further work, however, moving toward practical implementations of such models, an investigation into their robustness with respect to external events and possible threats should be investigated. In a recent study by Wang et al. (2019), some critical risks to power grids, and scenarios which could cause difficulties for machine learning algorithms are quoted as; abnormal noise, data injections where an attacker may manipulate signals with incorrect data and lines being taken out of service for maintenance. One study by Gupta et al. (2017) investigates the effect of various noise levels on a trained recurrent neural network (RNN), and finds a significant impact on performance. These two works motivate further investigation of these scenarios in this study.

3 Methodology

3.1 The IEEE 118 bus system

The IEEE 118 bus model was chosen as a case study in this thesis. Typically, similar studies have been conducted on smaller grid models, such as the IEEE 14 bus or 39 bus models. The 118 bus model was chosen to evaluate the performance of machine learning models on a larger system, with greater inertia, therefore producing benchmarks which may be a closer reflection of real-world systems. The standard 118 bus model contains 19 generators, 35 synchronous condensers, 177 lines, 91 loads and represents a simplified version of the power grid in the US Midwest. The grid is defined by four major zones.

Whilst the typical IEEE 118 bus model is suitable for steady state analysis, transient stability analysis requires a dynamic model, where the correct dynamic parameters for the synchronous machines exciter and governors are defined. These parameters are defined in the model used in this study (Kios 2015). However, this model does not containt dynamic properties for 20 condensers and 15 motors in the system. These parameters were therefore defined using benchmark data from (Demetriou et al. 2017). Following a previous study by ABB Power Grids (Nouti 2017), to create a more realistic test bed, each generator is connected to the high-voltage transmission system through ideal transformers as per the report by Demetriou et al. (2017). As a result of these modifications, the 118 bus model used in this study contains 172 buses, 185 lines, 76 transformers, and 91 loads which consume a nominal load of 3668MW and 1438 MVAr. The 118 bus system was modelled, and all data generated using DlgSILENT PowerFactory 2020. All data was generated with N-0 contingencies, meaning that all equipment is functional. Modern power systems are designed to remain stable up to N-1 contingencies, meaning any one element in the grid can be removed for servicing whilst still providing a continuity of service.

3.2 Data generation

As noted by Jimenez (2017), the system loading, and system inertia and spinning reserve highly influence the transient dynamics of the system. Given the increasing contribution of wind and other static forms of generation in power grids (ie. solar), various configurations of the 118 bus dynamic model were made, and the effects of increased wind penetration, and hence reduced spinning reserve were analysed. These scenarios were 0% wind, 12% wind, 25% wind, 35% wind, 50% wind and 60% wind penetration. Each scenario was constructed by removing a synchronous generator respectively, and replacing its nominal power reserves with a number of parallel 2MW wind turbines, simulating a wind farm. These turbines were taken from PowerFactory's standard static generator libraries. With respect to the 0% scenario, it was found that high levels of system inertia reduced the transient dynamics of the system following a contingency, and it was difficult to produce meaningful extreme events to train a machine learning model. Conversely, in scenarios above 35%, extreme transients become too frequent, causing long data generation times and less diversity in the time series'. As a compromise, all machine learning models in this thesis were trained on simulated data with 25% wind generation, including 15 synchronous machines and 4 wind farms.

With a grid model built, PowerFactory was used in engine mode, and interfacing was made using Python scripts. This allows for multiple scenarios to be run in a loop, and grid parameters changed on the fly. The dataset generated for this study consists of 1200 simulations. Each simulation is distinct and sampled at 0.001 second time steps. Each simulation starts at -20 seconds to allow the system time adjust to steady steady state. A fault then occurs at time 0 - (clearing time), and time series data is recorded from 0 - 5 seconds. Each simulation therefore consists of 5000 time steps. Schäfer et al. (2018) indicate the time frame of interest for transients as 3 - 7 seconds, and 5 seconds is consistent with other literature. In each simulation case, the contingency is a three fase fault at 50% of a transmission line. Since tripping all 185 lines proved to be too great a computation burden, 40 lines from the grid were randomly sampled as failure lines. For each line contingency the following

set of parameters were adjusted.

- Clearing time [seconds] - {0.1, 0.2, 0.3, 0.4, 0.5, 0.6}
- Total system loading from nominal [%] - {80, 90, 100, 110, 120}

Clearing times of 0.5 and 0.6 seconds are a stretch from reality, however these were made to create simulations with more extreme events. Initially, monte-carlo sampling was used to generate loads for the system from a probability distribution. In many cases, however, this led to non-convergence issues, hence a linear ramping of loads relative to the models nominal load was used. The exact data generation loop is detailed below.

Algorithm 1 Data generation loop

```
for line in random.sample(lines,40) do
  for time in clearingtimes do
    for load in loads do
      Create short circuit event
      Create clearing event
      Calculate initial conditions and load flow
      Execute RMS simulation
      Fetch data and cache
      Return grid to nominal state and clear all events
    end for
  end for
end for
```

As per Heidary (2018), the features recorded in each simulation are voltage magnitude (pu), current magnitude (pu), phasor angle between voltage and current (deg), and rotor angles w.r.t the reference generator (deg). Since there are 15 generators in the modified 118 bus model with 25% wind penetration, this leads to 60 features for each time series simulation.

3.3 Data processing

Data preparation and normalisation

All data processing, modelling and analysis was made using Python 3.7. All machine learning models were trained using a Tensorflow 2.3.0 backend with Tesla P100 GPU provided by Google Colab.

The dataset generated from PowerFactory consists on 1200 time series simulations, and represents a less common time series problem. Classical time series forecasting involves a continuous time series, where are training and validation split is made in time, with a testing set typically coming from data collected in the future. All time series in this study are independent, and machine learning models need to be trained to generalise to any operational scenario from time 0, rather than a continuation of a past time series.

Firstly, each time series was down sampled by a factor of 10 from 0.001 seconds to 0.01 seconds. Running a Fast Fourier Transform on the time series, it was found that the most important frequencies lied between 0.5 and 0.7 seconds. Down sampling means this signal can be more easily captured with smaller data windows, possibly increasing model performance and limiting exploding gradients issues due to long sequences. Down sampling also decreases training time and memory requirements. The dataset was converted to a numpy array with shape [simulations, time steps, features], and randomly shuffled along the first axis. The data was then split into training, validation and testing sets. The split was made along the first axis yielding 70% of the data for training, 20% for validation and 10% for testing. Training, validation and test sets hence had the following shapes. Since the reference generator rotor angle is zero for all time steps, this feature was removed from the dataset.

- Training - [840, 500, 59]
- Validation - [240, 500, 59]
- Testing - [120, 500, 59]

In order to aid model convergence, particularly since LSTM models utilise a hyperbolic tangent activation function which is centered on zero, all features were normalised by subtracting the feature mean and dividing by the feature standard deviation. To preserve the relative magnitudes of each feature with respect to each simulation, a chaining operation was made whereby normalisation is made for each feature (the third dimension), across each simulation (the first dimension). In order to avoid data leakage, and ensure the models have no access to the validation set statistics, only the training set mean and standard deviations were used for normalisation.

Data windowing and input pipeline

In order to create a large number of independent time series sequences, data windowing was used to create feature and label pairs to train machine learning models. This also reflects a real world monitoring scenario, where a model is required to make a forecast based on short windows of data in real time. The following procedure was implemented for training, validation and test sets.

1. *Figure 5* illustrates the data windowing process. In the following example, a model takes 5 time steps of input data, and predicts the next 5 time steps as one sequence. These windows are shifted for each time step, hence, for each time series the number of input output pair windows is equal to the number of time steps - output sequence length. By unrolling across the first dimension of the data matrix, for the below example there are 415800 data windows in the training set.

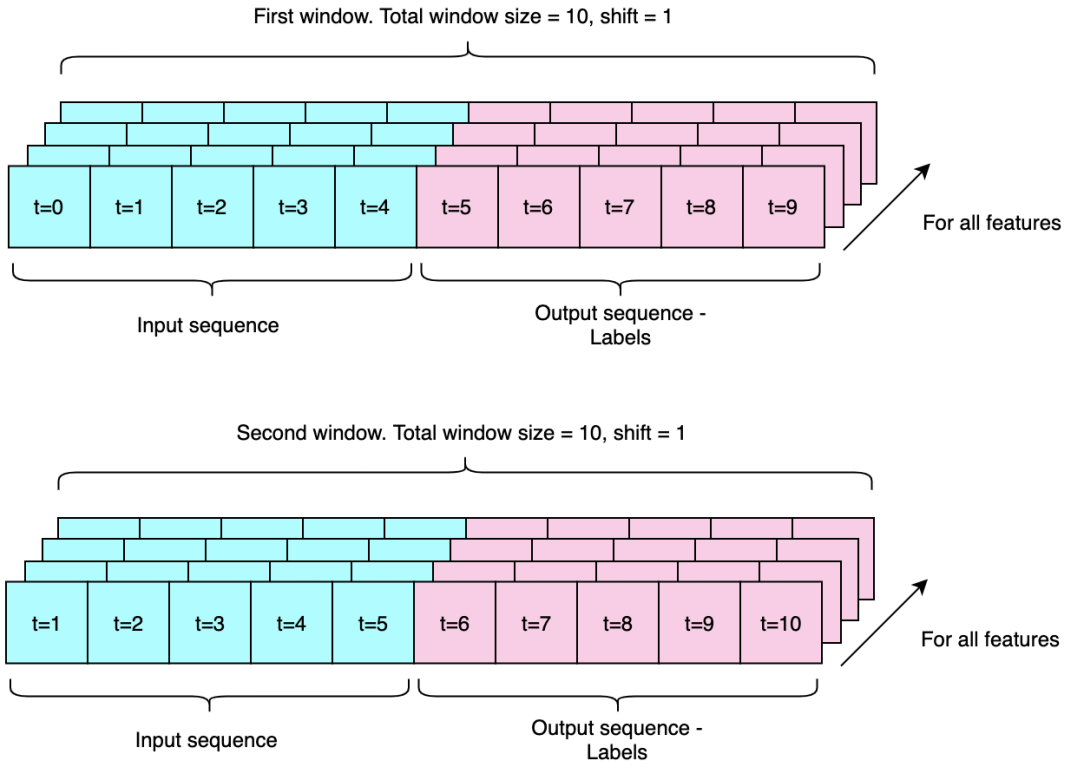

Figure 5: Example of a windowed dataset

2. To evaluate the difference in performance between models trained directly on rotor angles from PowerFactory, and models trained on only input features pertaining to PMU signals, two different

windowed datasets were created. These will be referred to as dataset 1 and dataset 2 respectively in this report. Dataset 1, as illustrated by *Figure 6* takes the relative rotor angles of each machine in the system as inputs. The output labels are the same features, shifted by one time step, with a length as long as the specified output sequence to be forecast.

Dataset 2 takes as inputs only PMU signals. These features are; V_t - the magnitude of the generator terminal voltage measured by PMU (pu), I_a - the magnitude of the generator current measured by PMU (pu) and φ - the phase angle of the generator current with respect to its terminal voltage measured by PMU (deg). The input sequences contain these three features for each of the 15 machines in the system, hence the input dimensionality is 45. The output sequence is identical to dataset 1, containing only the relative rotor angles for each machine. In this case, we expect a model to learn the function mapping between PMU signals and relative rotor angles, plus the sequential patterns in the data.

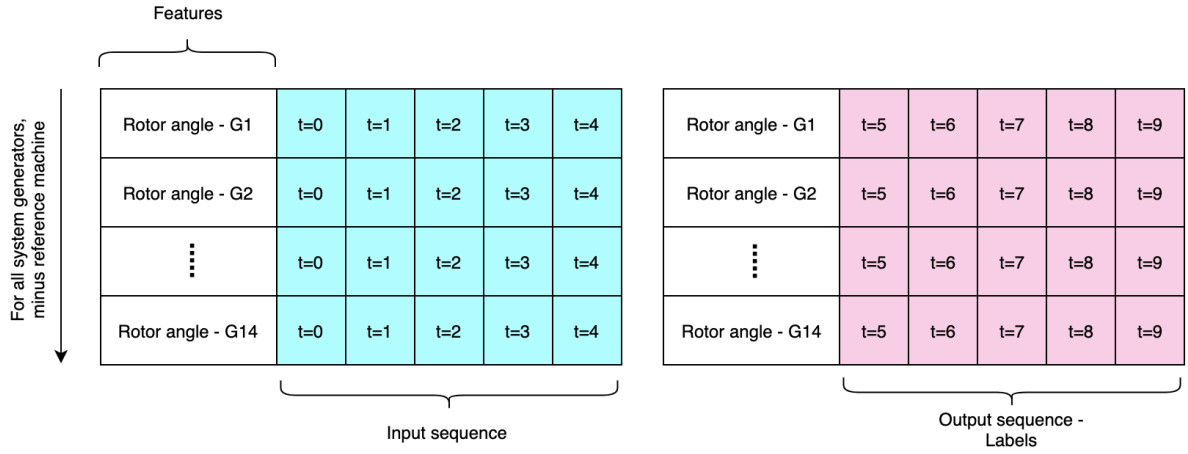

Figure 6: Example input and output labels for dataset 1

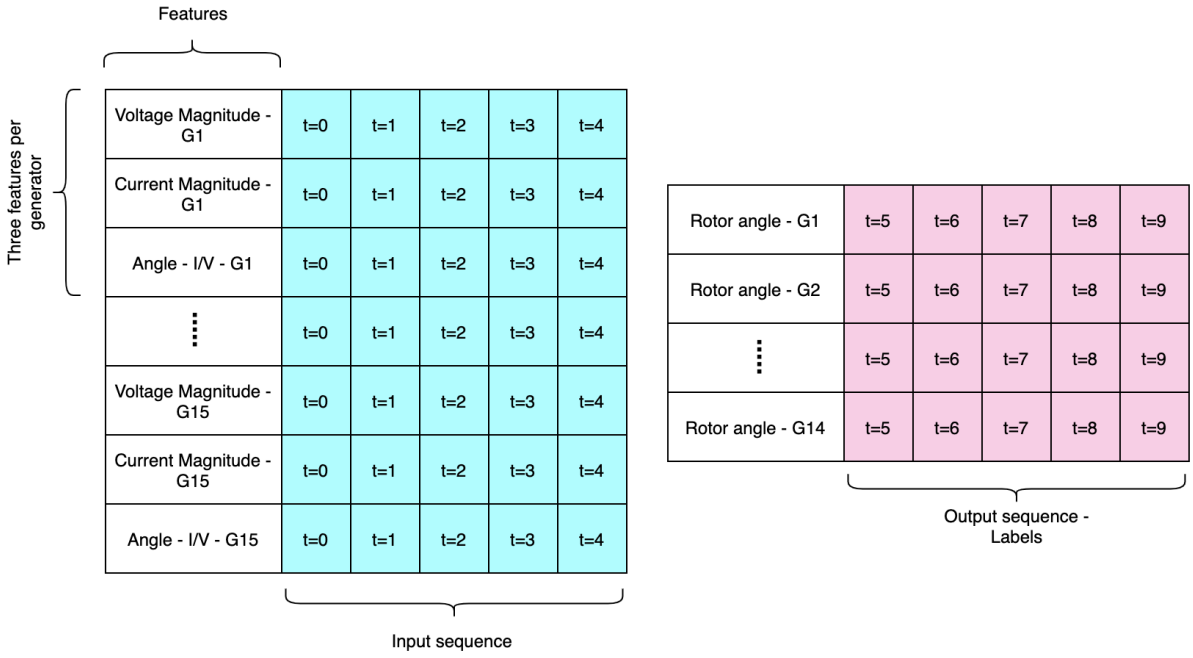

Figure 7: Example input and output labels for dataset 2

3. Before training, the windowed datasets were batched. Since gradient updates are made as an average loss across the batch, choosing an appropriate batch size is often a trade off between training speed, convergence speed, and loss noise. Various batches sizes were investigated, and 16 was chosen as optimal with respect to these factors. The datasets are therefore converted to 3D tensors of shape [batch size, time steps, features]. Using Tensorflow's tf.Data API, these batches are shuffled, cached and prefetched, meaning a new batch will be pre-loaded to GPU during the previous back-propagation step to increase training efficiency.

Unless otherwise stated, for model evaluation and comparison, this study uses windowed datasets of an input length of 30 time steps and output length of 20 time steps. 30 time step inputs was deemed long enough to capture important system dynamics, yet not so long that prediction are made too late following a disturbance. In later sections, this output window is increased and compared with existing benchmarks in literature.

3.4 Machine learning models and architecture

Baseline

Before training any complex model, it is important to establish a naive baseline. In a single step forecasting problem, such a baseline could be an average of a number of previous time steps or an ARIMA model. Since this is a multi-step forecasting problem, an appropriate baseline is to forecast that all output timesteps will take the same value as the last value in the input window. If any model cannot significantly outperform this simple baseline then we can conclude no significant learning is taking place and such a model is not suitable for this specific problem. *Figure 8* illustrates this baseline for an input window of 30 time steps, and output window of 20 time steps for a single generator in the system.

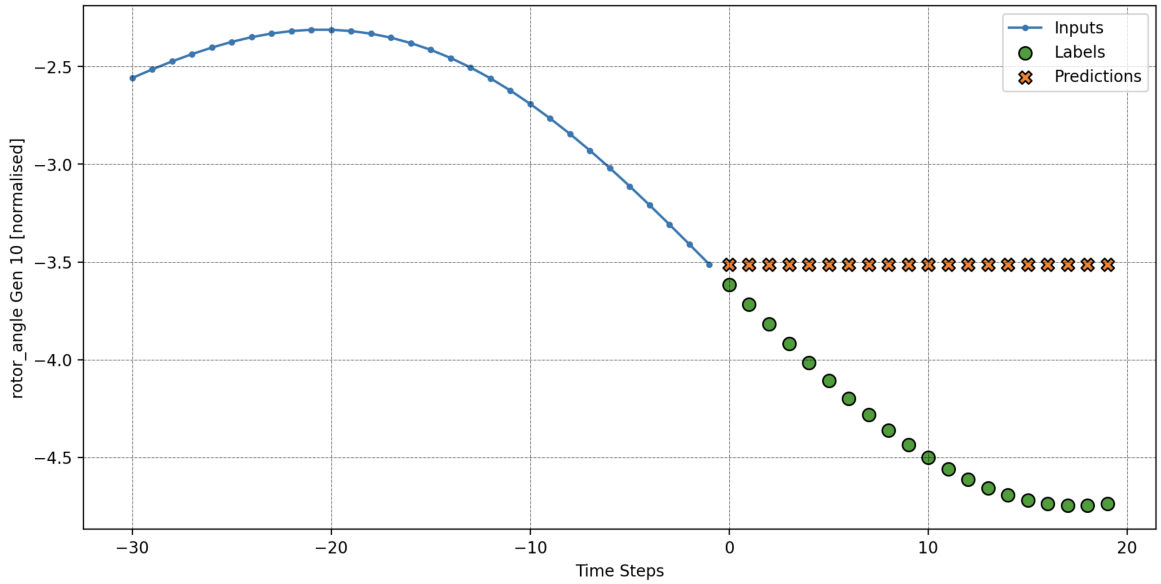

Figure 8: Baseline predictions for a windowed dataset for a single machine

Non-sequential models

In this study, we aim to use neural networks to find an optimal function $\mathbf{y} = f^*(\mathbf{x})$, which maps an input sequence vector $\mathbf{x}$ to an output vector sequence $\mathbf{y}$. A neural network defines this mapping as $\mathbf{y} = f(\mathbf{x}; \mathbf{W})$ where the networks parameters $\mathbf{W}$ are learnt in a stage wise fashion to produce a best function approximation.

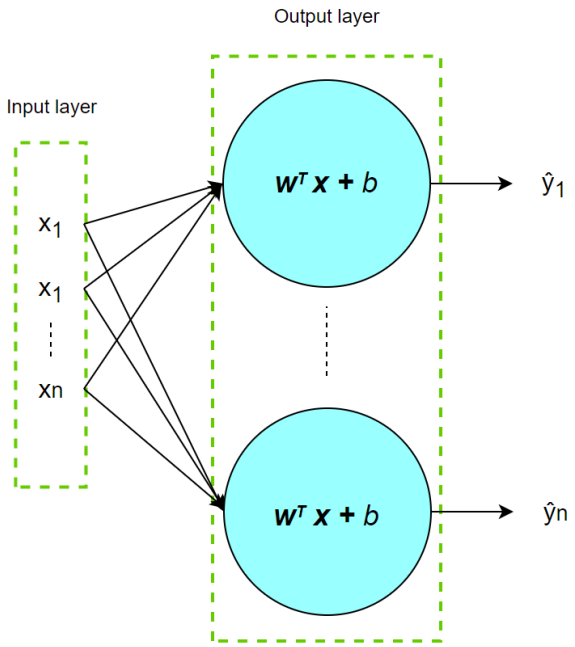

Figure 9: Neural network structure for multivariate linear regression

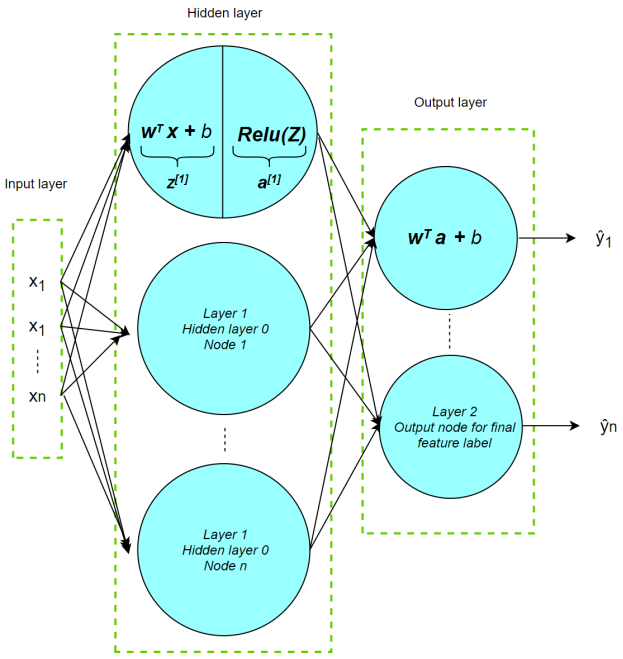

Figure 10: Neural network with single hidden layer and ReLu activation function

Figures 9 and 10 show the graph architecture of linear regression and feed forward implementations. The key difference is that linear regression contains no non-linear functions, and is equivalent to an approximation of solving a linear system of the form $\mathbf{Ax} = \mathbf{b}$. For a feed forward network, non-linear activation functions are introduced which can capture complex nonlinearities in the data. These networks are optimised using gradient descent and back propagation (Goh 2017), whereby the gradient of the $loss(y, \hat{y})$, propagate back though the network. The values of the networks weights are then adjusted with respect to the negative gradient of the loss in small steps known as the learning rate. One backward propagation through the network computes the Jacobian transpose vector product $(J^T V)$, which contains all the partial derivatives with respect to the loss for each node in the network. Using vectorisation, these gradient computations can be made in a single step for each layer in the network.

A number of non-sequential models were evaluated to determine whether significant performance gains could be achieved from more complex neural networks. These models were in part inspired from previous literature and include; a feed forward neural network with a single hidden layer and 512 units, a 1D convolutional neural network learning 256 filters (Petersen et al. 2019) and a simple linear model as per *figure 9*. The advantage of a linear implementation is it provides a secondary baseline, and is easy to interpret by inspecting the weights (parameters) of the model.

Long-Short Term Memory Network

The key difference between LSTM networks and the previously described models is that LSTM's conduct back propagation through time, and can store long term sequential dependencies. As illustrated by the following figure, an LSTM network is a series of neural networks which feed information between cells using a for loop (Goodfellow et al. 2016). *Figure 12* highlights the architecture for each LSTM cell implemented in this study.

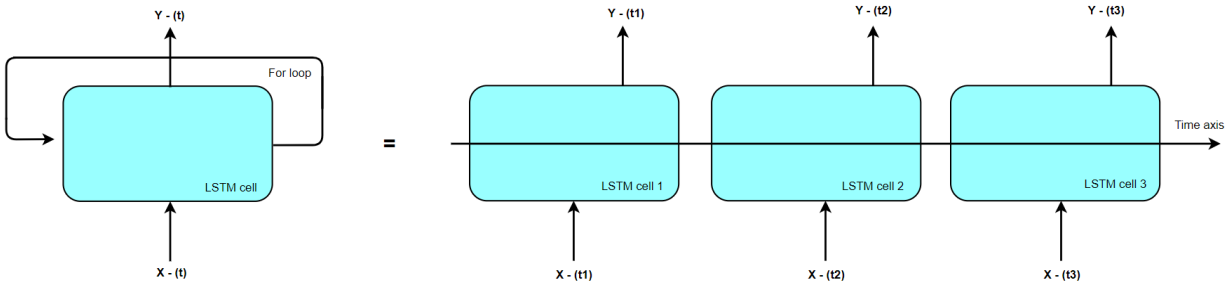

Figure 11: LSTM Network unrolled in time

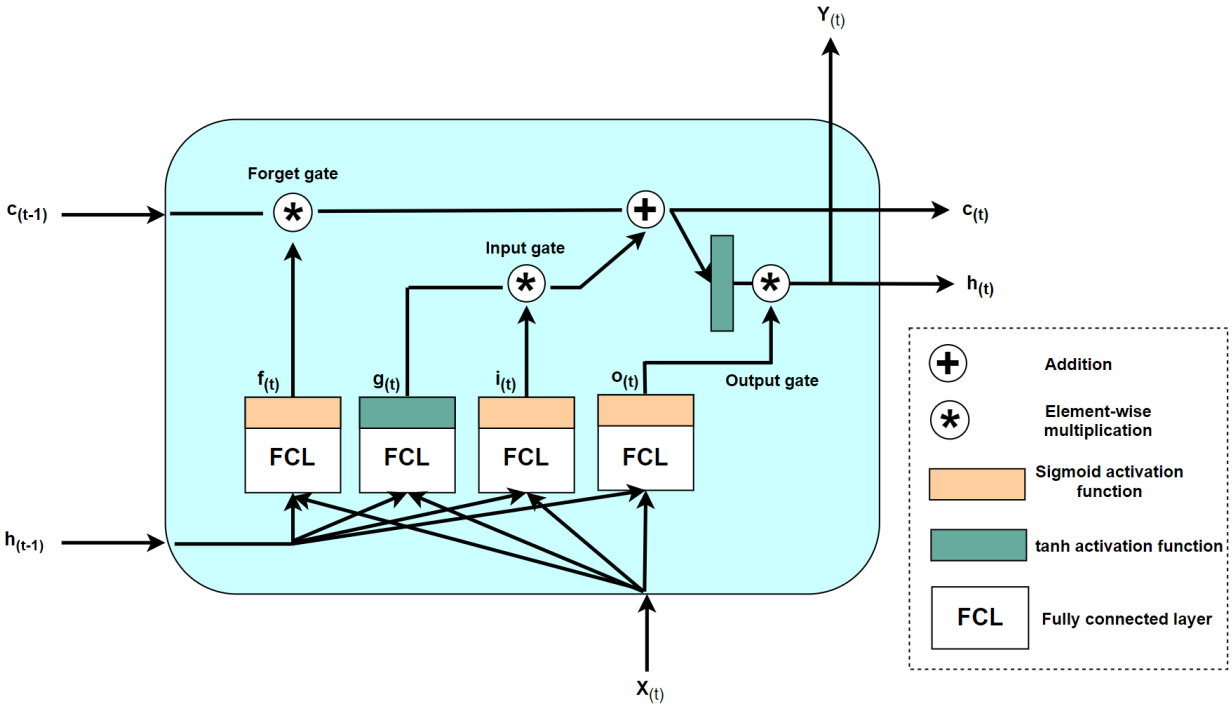

Figure 12: LSTM cell architecture

c represents the long term state of the network, which passes from cell to cell left to right. **h** is the short term state, which is equal to the cells output **y** for each time step and is passed only to the adjacent cell in time. **h**, and the input vector **x** are fed into layer **g** which produces an activation in the range of [-1,1]. Layers **i** and **o** are gate controllers, and are responsible for determining how much of this signal should pass to the next short term state and long term states **h** and **c** respectively. These controllers use a sigmoid functions with a range of [0,1] and since these controllers feed into vector multiplication gates, it is clear to see that an output close to zero would diminish the output of **g**, and effectively "open" the gates with activation's close to 1. The activation of **i** will also determine how much this cell should contribute to the long term state of the network through an addition operation. The sigmoid activation from **f** determines how much long term memory state should pass through the cell. The matrix-vector computation which govern each controller and cell output are as follows.

$$\mathbf{i}_{(t)} = \sigma(\mathbf{W}_{xi}^T \mathbf{x}_{(t)} + \mathbf{W}_{hi}^T \mathbf{h}_{(t-1)} + \mathbf{b}_i) \tag{1}$$

$$\mathbf{f}_{(t)} = \sigma(\mathbf{W}_{xf}^T \mathbf{x}_{(t)} + \mathbf{W}_{hf}^T \mathbf{h}_{(t-1)} + \mathbf{b}_f) \tag{2}$$

$$\mathbf{o}_{(t)} = \sigma(\mathbf{W}_{xo}^T \mathbf{x}_{(t)} + \mathbf{W}_{ho}^T \mathbf{h}_{(t-1)} + \mathbf{b}_o) \tag{3}$$

$$\mathbf{g}_{(t)} = tanh(\mathbf{W}_{xg}^T \mathbf{x}_{(t)} + \mathbf{W}_{hg}^T \mathbf{h}_{(t-1)} + \mathbf{b}_g) \tag{4}$$

$$\mathbf{c}_{(t)} = \mathbf{f}_{(t)} * \mathbf{c}_{(t-1)} + \mathbf{i}_{(t)} * \mathbf{g}_{(t)} \tag{5}$$

$$\mathbf{y}_{(t)} = \mathbf{h}_{(t)} = \mathbf{o}_{(t)} * tanh(\mathbf{c}_{(t)}) \tag{6}$$

Where $\mathbf{W}_{xi}$, $\mathbf{W}_{xf}$, $\mathbf{W}_{xo}$, $\mathbf{W}_{xg}$, are the weight matrices for each of the fully connected layers with respect to the input $\mathbf{x}_{(t)}$, $\mathbf{W}_{hi}$, $\mathbf{W}_{hf}$, $\mathbf{W}_{ho}$, $\mathbf{W}_{hg}$ are the weight matrices for each layer with respect to the short-term state output from the previous cell, and $\mathbf{b}$ represents bias vectors for each layer.

This study also used a bi-directional implementation of an LSTM network (Schuster & Paliwal 1997). In this case we duplicate the original LSTM layer, and hence double the networks parameters. The hidden and long term states move forward in time for the original layer, and backward in time for the duplicated layer, allowing the model access to sequential information from both past and present for a given sequence.

Autoencoder

An autoencoder is a pair of functions, $h : \mathbb{R}^d \to \mathbb{R}^m$ and $g : \mathbb{R}^m \to \mathbb{R}^d$, where d is the dimentionality of the original space, and m is the dimentionality of the autoencoder embedding (or compression). In this study, h is a two layer bidirectional LSTM network, with 200 units and 60 units respectively, g contains a repeat vector, which copies the embedded space in a row wise fashion by the length of the input sequence, and is followed by a bidirectional LSTM layer with 200 units and a fully connected dense layer which learns to reconstruct the sequence. The reconstruction error for $f(h; g)$ is optimised using Mean Absolute Error such that $Loss(W) = \sum_{n=1}^{B} Loss(\mathbf{x}_i, g(h(\mathbf{x}_i)))$, where B is the number of batches and $\mathbf{x}_i$ is a given sequence.

The motivations behind using an encoder decoder architecture is partly inspired by large gains in performance quoted from Zhu & Laptev (2017) when dealing with complex multi-step time series forecasting over single LSTM models where they report a 35% increase in performance. In this study, the encoder-decoder model works in two blocks. The autoencoder described above compresses the time series batches into a lower dimensionality subspace (embedding). This embedding is then fed into a separate prediction network which produces forecasts. It was found that once the autoencoder was trained, this architecture reduced training times for the forecast net when compared to a stand alone LSTM network, likely due to the simplified input data. This architecture was also explored to address the issue of data generation to train models in this application. Without the autoencoder block, models need to be trained on fixed input-output sequence lengths. Conversely, the autoencoder can be trained on arbitrary sequence lengths. It is therefore possible that the autoencoder block could be retrained frequently in a real-world scenario independent of the forecasting block, increasing the quality of embeddings due to an increased exposure to real-world PMU data. The practical aspects of this will need to be explored in further study. The autoencoder block was trained inpependently of of prediction block in this study, but in practice the autoencoder block could be re-trained when new transient data becomes available. The prediction block can continue to make predictions using the weights saved from the end previous autoencoder training cycle. The exact model architecture is detailed in the following figure.

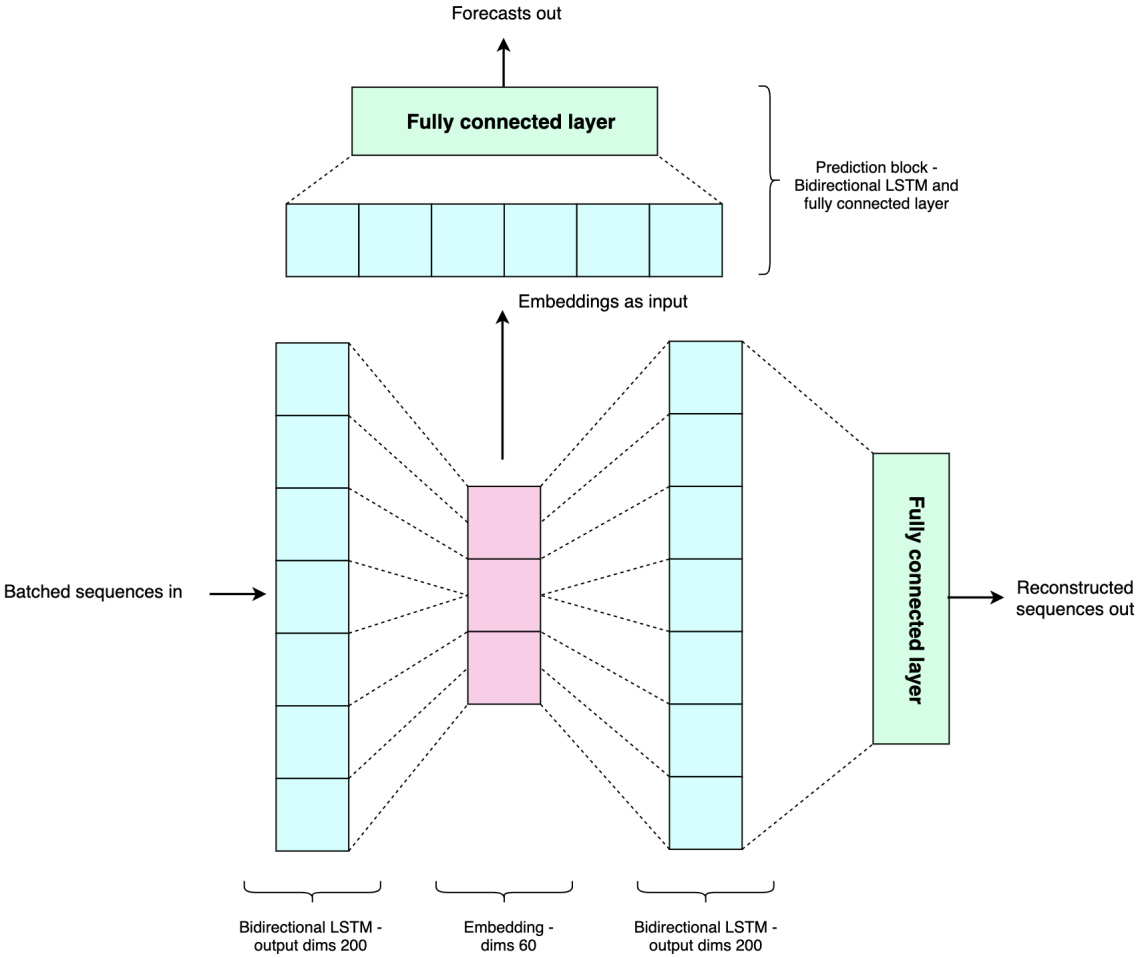

Figure 13: Encoder-decoder architecture

Monte-Carlo dropout

Looking forward, if machine learning methods for online transient stability monitoring are to be integrated into the grid, they must operate as part of the grids existing protection systems. Currently such systems typically operate under known thresholds, with well understood and predictable behaviour. For example, a fuse will melt at a specific operating point, or a breaker may trip at a specified and measurable over voltage threshold. A point forecast from a neural network gives no indication of uncertainty in that forecast, as such it may be difficult to justify actions on such a forecast given the consequences of grid interventions can he high. Such an application in deep learning is referred to as Bayesian Neural Networks (BNN) and aims to find the posterior distribution of the model weights (W). Gal (2016) , provides an overview of the state of the art in BNN, but points out that most methods require complex training schemes, and an increased number of model parameters often lead to large training times. In this study we instead use Monte-Carlo dropout as a Bayesian approximation (Gal & Ghahramani 2015). We apply dropout layers in the forecasting block with a dropout probability of 0.2. A dropout layer is applied at each time step, and a further dropout layer is applied to the activations of the connected layers in each LSTM cell (known as the recurrent dropout layer) (Chollet 2017). The algorithm procedure used is exactly as follows: given a new input sequence $\mathbf{x}$ predict the output sequence $\hat{\mathbf{y}}$, with dropout layers activated. The stochastic prediction is repeated $\mathbf{B}$ times, where $\mathbf{B}$ is the number of inference cycles to obtain $\{\hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_B\}$. The model uncertainty is therefore estimated by (Gal 2016):

$$\hat{Var}(f^W(\mathbf{x})) = \frac{1}{\mathbf{B}} \sum_{b=1}^{B} (\hat{y}_b - \bar{\hat{y}})^2 \tag{7}$$

Where:

$$\bar{\hat{y}} = \frac{1}{\mathbf{B}} \sum_{b=1}^{B} \hat{y}_b \tag{8}$$

Two drawbacks were found with this method. First, increasing inference cycles increased the accuracy of the model predictions, however this also increased inference time. In a real-world scenario, where there may only be fractions of seconds at hand to make accurate predictions, this method may add an unacceptable delay to prediction times, therefore it could be more appropriate for offline accuracy and model stability assessments. Further, such an implementation of recurrent dropout cannot be run on GPU, hence models with recurrent dropout needed to be trained on CPU. Training times were in the order of 24 hours, as opposed to 30 minutes for models training without recurrent dropout.

3.5 Evaluation metrics and hyperparameters

Throughout the training process, model performance was monitored using the validation dataset, the test set was held out during training, and only used to show performance on unseen data. All models were trained using the same strategy, with a maximum of 40 epochs, and early stopping in the case that validation accuracy does not improve after 2 epochs.

Multiple loss functions were explored, and their performance examined on the validation set. It was found that Mean Absolute Error (MAE) ($\frac{1}{N}\sum_{i=1}^{N} \|y - \hat{y}\|$) yielded the best performance and with the exception of linear regression (which was optimised using Mean Squared Error (MSE)), all models were optimised using an MAE loss function. The fact that the data is non-gaussian, and extreme events are less frequent could explain why MSE under-performed MAE. MAE is robust to outliers, however its gradient is continuous at 1 for ($\hat{y} > y$), -1 for ($\hat{y} < y$) and non-differentiable at ($\hat{y} = y$). This means MAE penalises small errors with the same magnitude as large errors, which can lead to divergence issues for a continuous learning rate. To mitigate this issue, Adam optimisation was used with an adaptive learning rate (Kingma & Ba 2014). For each model, an optimum learning rate was found over 100 epochs, for each epoch, the learning rate was increased from 1e-8 over a given search space. Typically, optimal learning rates were found between 0.0001 and 0.001. For LSTM models, exploding gradients was observed after the first 6-8 epochs, evident by monitoring the gradient magnitudes and a related increase in training loss. To mitigate this issue, all partial derivatives with respect to the loss were clipped with respect to their $l2$ norm at a value of 1.0, thereby diminishing the gradients magnitude, but preserving orientation.

Since this is a multi-output, multi-variate forecasting problem, model performance is given as an average MAE across the entire output sequence, averaged across all output features.

3.6 External events

To explore model performance against the external events quoted by Wang et al. (2019), two scenarios were created. The first adds various levels of noise to model inputs. This noise is Gaussian, with a mean of 0 and specified variance. Noise level 1 has a variance of 0.1, each subsequent noise level doubles this variance such that noise level 5 has a mean of 0 and variance of 1. Data injections were also simulated by manipulating data at a single generator bus in the system. *Figure 14* illustrates the impact of these noise levels on a sample sequence feature.

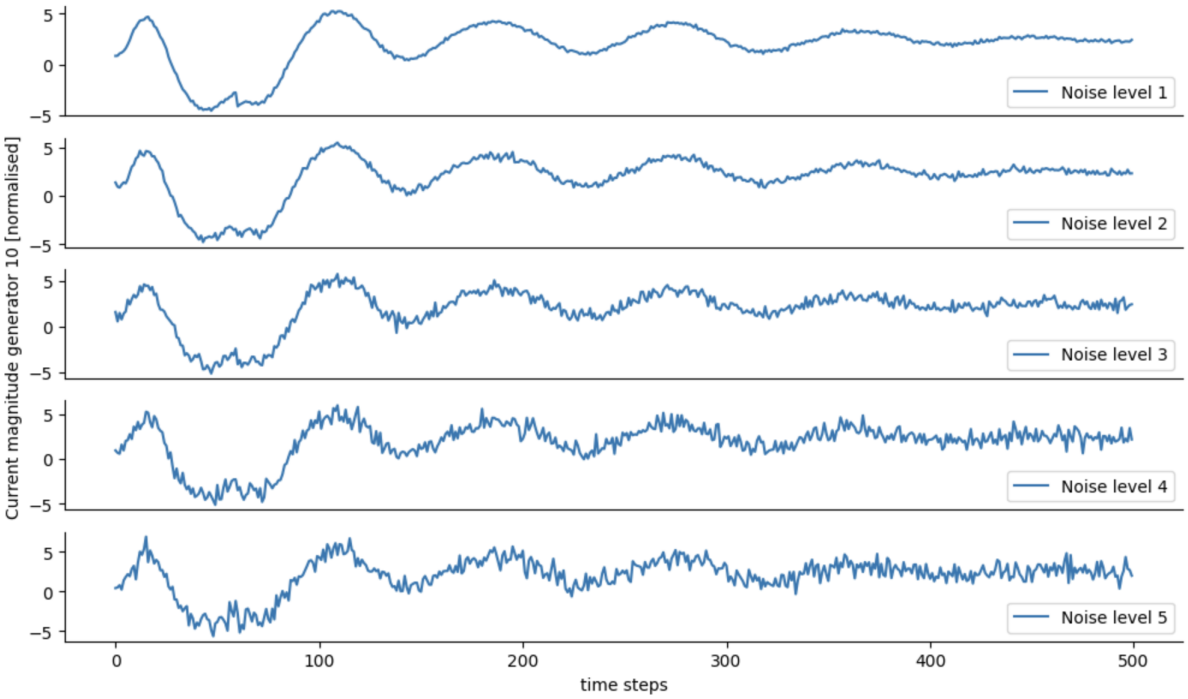

Figure 14: Noise levels added to input signals

To understand how the model performs under topology changes five new test sets were generated. Four have lines out of service in each of the systems zones respectively under N-1 criterion, and in the fifth, two lines out of service in zones 1 & 2 under N-2 criterion. The same data generation loop was repeated as per the main dataset, however only a single line was tripped to provide a direct comparison between scenarios. A three phase fault on line 49-69 was chosen as a contingency since this line is at the intersections of zones 1, 2 and 3 and a contingency on this line had not been see by the model. Lines out of service (OOS) in each scenario are as follows:

- Zone 1 - Line 15-17
- Zone 2 - Line 54-59
- Zone 3 - Line 82-83
- Zone 4 - Line 103-105

4 Results

4.1 Model performance comparison

Datasets 1 & 2

Model accuracy for datasets 1 and 2 are highlighted in *figures 15 and 16*.

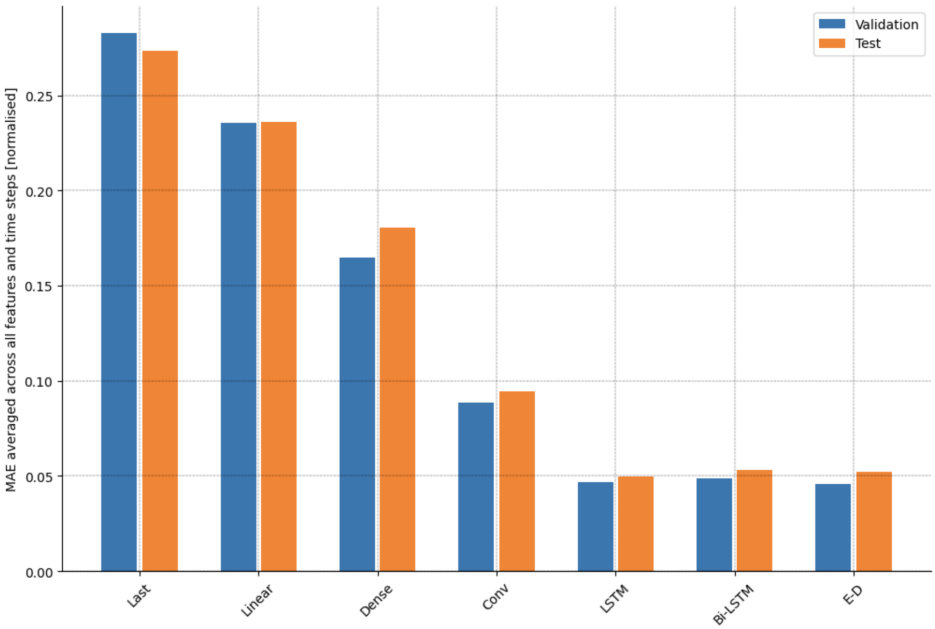

Figure 15: Performance across all models for dataset 1

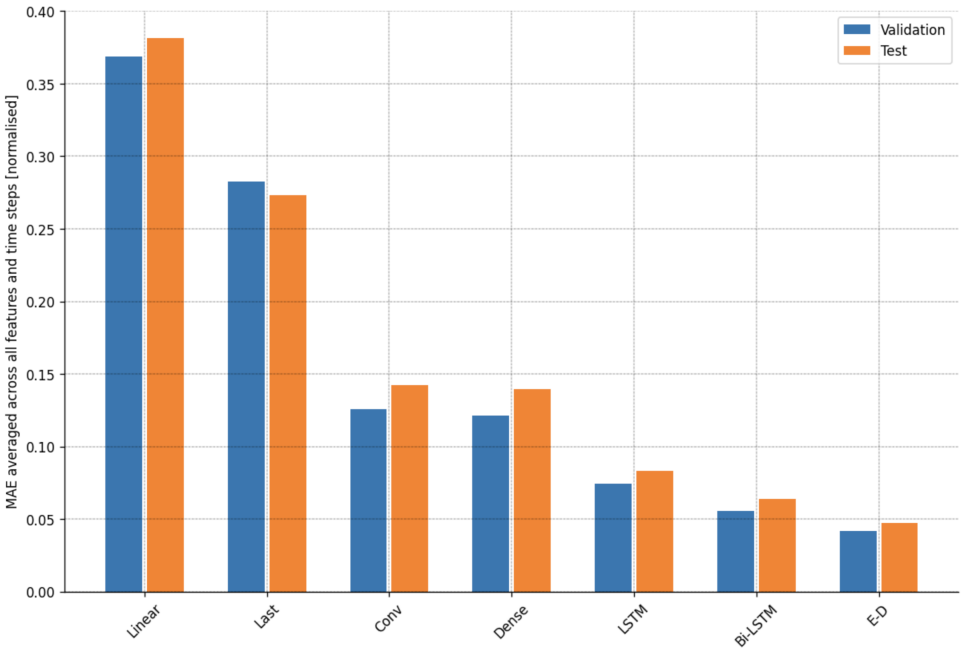

Figure 16: Performance across all models for dataset 2

Model	Validation loss dataset 1	Validation loss dataset 2	Test loss dataset 1	Test loss dataset 2
Last	0.283	0.283	0.273	0.273
Linear	0.236	0.381	0.237	0.382
Dense	0.165	0.122	0.180	0.139
Conv1d	0.0887	0.126	0.0946	0.142
LSTM	0.0466	0.0745	0.050	0.0831
Bi-LSTM	0.0485	0.0557	0.053	0.0641
Encoder-decoder	0.0458	0.0421	0.0520	0.0471

Table 2: Validation and test results across all models for dataset 1 and 2 quoted in MAE [normalised]

The added complexity for dataset 2 is evident from the linear models poor performance when compared with the simple baseline. The linear model forecasts with a linear projection, and can only handle a low dimensional slice of the temporal behaviour of the input sequence. For dataset 1, performance gains are significant with the introduction of LSTM architectures, but plateaus as the complexity of these models is increased, largely down to overfitting taking place during the training process. Conversely, the encoder-decoder model shows a greater performance on dataset 2 than the far simpler dataset 1. *Figure 17* compares the predictions for the best and worst performing models on dataset 2 (linear and encoder-decoder), for two prediction windows. It is clear to see that the linear model has a tendancy to make a continuous projection in time, whereas the encoder-decoder model can effectively capture non-linearity in output sequences.

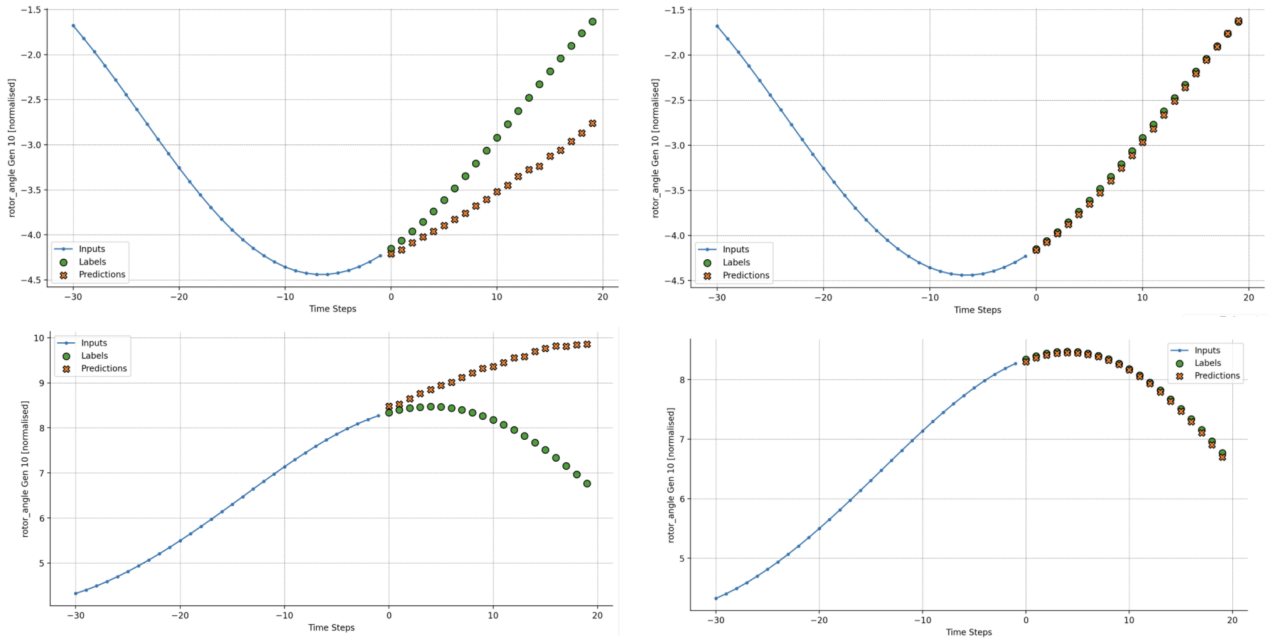

Figure 17: Left: linear model, Right: Encoder-decoder model. Inputs refer to the actual rotor angles with respect to the input features

We can conclude that, at the cost of increased model complexity, PMU signals can be effectively used as input features to machine learning models to forecast relative rotor angles, and can even result in increased performance. The remainder of this analysis therefore only considers the encoder-decoder model on dataset 2.

Forecasting over extended time horizons

Extending the output sequence to be forecast up to two seconds (200 time steps) we observe an expected degradation in performance, however, considering all predictions are made in a single shot with only a 30 time step input sequence, the model remains relatively robust over longer forecasting periods. With a more than doubling of forecasting length from 0.8 seconds to 2 seconds, we only observe a 20% change in forecasting accuracy. Whilst using normalised values provides a qualitative comparison of model performance, it can be difficult to contextualise these values. *Table 3* therefore provides accuracy results for the encoder-decoder model on dataset 2 across all forecasting periods in degrees, with the data normalisation reversed. Comparisons can therefore more easily be made across existing works in literature, and indeed this study can be benchmarked by providing accuracy results across various metrics.

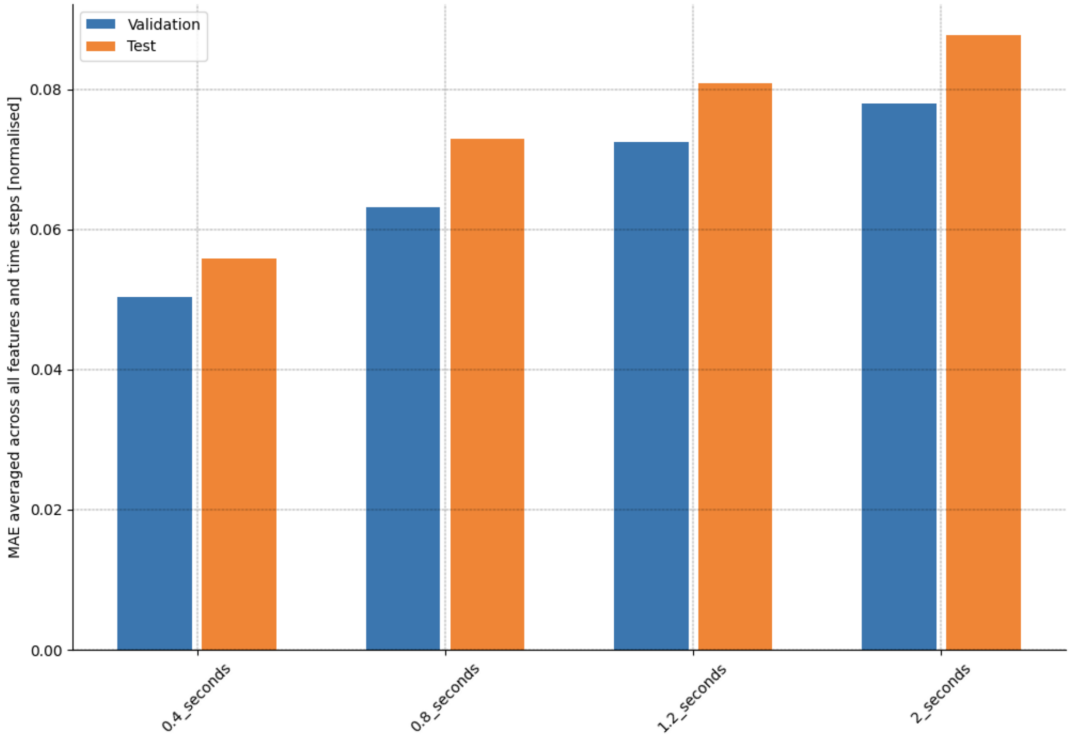

Figure 18: Validation and test set accuracy across multiple forecasting periods (MAE)

Forecasting window	MAE [deg]	MSE [deg]	RMSE [deg]
0.2 seconds	0.179	0.451	0.671
0.4 seconds	0.212	0.717	0.846
0.8 seconds	0.280	1.154	1.074
1.2 seconds	0.311	1.355	1.164
2 seconds	0.339	1.604	1.266

Table 3: Results for encoder-decoder model across multiple forecasting periods quoted in degrees

Given the absolute range of possible forecasted rotor angles is 360 degrees, an MAE of 0.339 degrees for two seconds appears to show strong forecasting performance for the encoder-decoder model. It is difficult to make direct comparisons within existing literature, since no authors have formulated the problem as multi-label multi-step on a grid as large as the IEEE 118 bus model. (Liu et al. 2019) present their results for the much smaller IEEE 39 bus model (with half as many generators) in terms

of RMSE. Whilst they have further simplified the problem by training an individual LSTM's for each generator, for 2 second look ahead times their accuracy results for each generator in the system fall within a range of [0.286, 1.662]. Results for this study fall within this range, which shows great promise considering the added complexity in this case. One difficulty in benchmarking against (Liu et al. 2019) is that they only present results from one specific output sequence in their test set, whereas results in this study are an average of many thousands of sequences, where outliers certainly increase the mean of our error measurement.

With the exception of Gupta et al. (2017) very little work has been done in literature to test the limitations of machine learning models for transient stability analysis. Accuracy is only part of the picture and, thinking practically, if the objective of building such models is to understand whether such approaches are appropriate in real-world power grids, then the limitations of such models need to be rigorously explored and exploited. The following sections attempt to define such limitations.

4.2 Regression and classification error

Figure 19, shows the distribution of errors for all sequences in the test set, and paints a more compelling picture of model performance than the plots in the previous section.

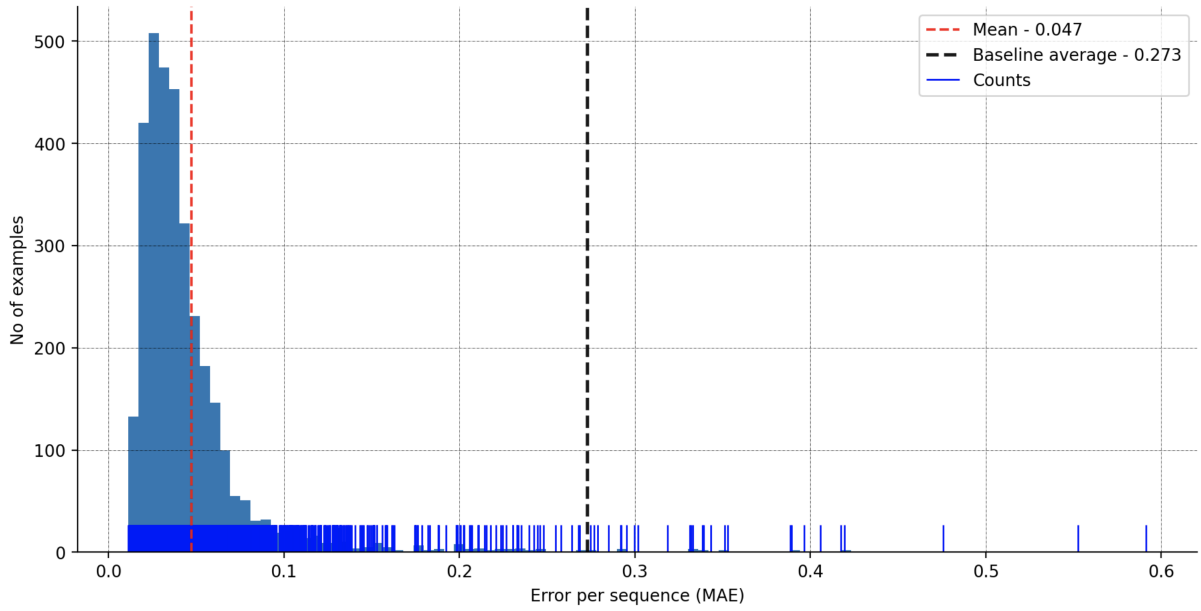

Figure 19: Distribution of test set errors

It is clear that whilst our average MAE over all sequences is good, there are a significant number of sequences with accuracy's far from the mean. We also see that in some cases, these sequences show a worse performance than the average baseline MAE. In the context of online transient forecasting, one has to consider whether such outliers are acceptable, when they occurred, and whether such errors would result in a catastrophic situation being missed as a false negative. Once these questions are answered, strategies can be devised for model optimisation. Using the TSA classification metrics from Jimenez (2017), we can analyse a pseudo classification error for our test set predictions. The confusion matrix in *Figure 20* compares the predictions for each time step in each sequence in the test set against the actual rotor angle for those time steps. In doing this, must like in literature, the TSA classification accuracy is high ($> 99\%$), but on inspection of the plot it is clear to see why such metrics should be treated with scepticism.

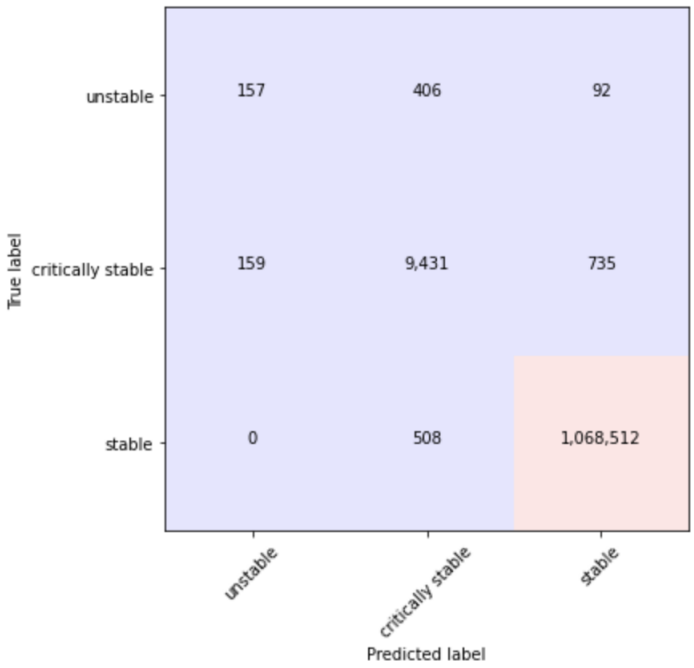

Figure 20: Confusion matrix to show the TSA classification error - accuracy 0.9982, missclass 0.0018

Firstly, our model has predicted false positives in 70% of all unstable time steps. This is cause for concern, since the sole objective of such a model is to provide an early warning system and mitigate such scenarios. This is in part due to the severe "class" imbalance in the dataset. It is clear to see that the model has been trained on significantly more low variance (or stable) timeseries, that unstable series. This reflects some naivety in the data generation process, a process which is shared by most works in literature. Unstable events are rare by nature, so rather than generating data from random contingencies, a more probabilistic approach to data generation should be employed, where only faults with a high likelihood of causing instability should be initiated. This will result in a more balanced dataset and will likely improve accuracy in unstable cases. Another approach could be to intentionally weaken the grid to N-1 or N-2 before initiating contingencies.

4.3 Monte-Carlo dropout

Using Monte-carlo dropout, we are no longer concerned with point forecasting our time series, but rather estimate the mean of a sample of predictions where the sample size is equal to the number of inference cycles. With a well trained model, such an implementation can boost model predictive performance. In our case, with 100 inference cycles per sequence, marginal improvements in accuracy are observed with an MAE of 0.0427 for the test set. This, of course, comes with increased predictive time complexity, scaling linearly with the number of inference cycles. More importantly, this approach provides an estimate of model uncertainty in the form of the sample variance of each sequence. *figure 21* compares a complete set of time series predictions for two machines in the system, comparing models trained with and without Monte-carlo dropout.

Observing *Figure 21*, we see an increase in predictive accuracy in the final two thirds of the time series. The Monte-carlo plots also show an increase in model uncertainty in the beginning period of the series. Generalising this anaylsis, *figure 22* shows this same trend on average for all time series in the test set, with peaks in model uncertainty during the early transient swings in the series. Ideally, this plot would show a uniform distribution of uncertainty. As it stands, the period of greatest model uncertainty is also the most critical time period in the series, the time period where a swing is most likely to trip a generator and potentially initiate a cascading event.

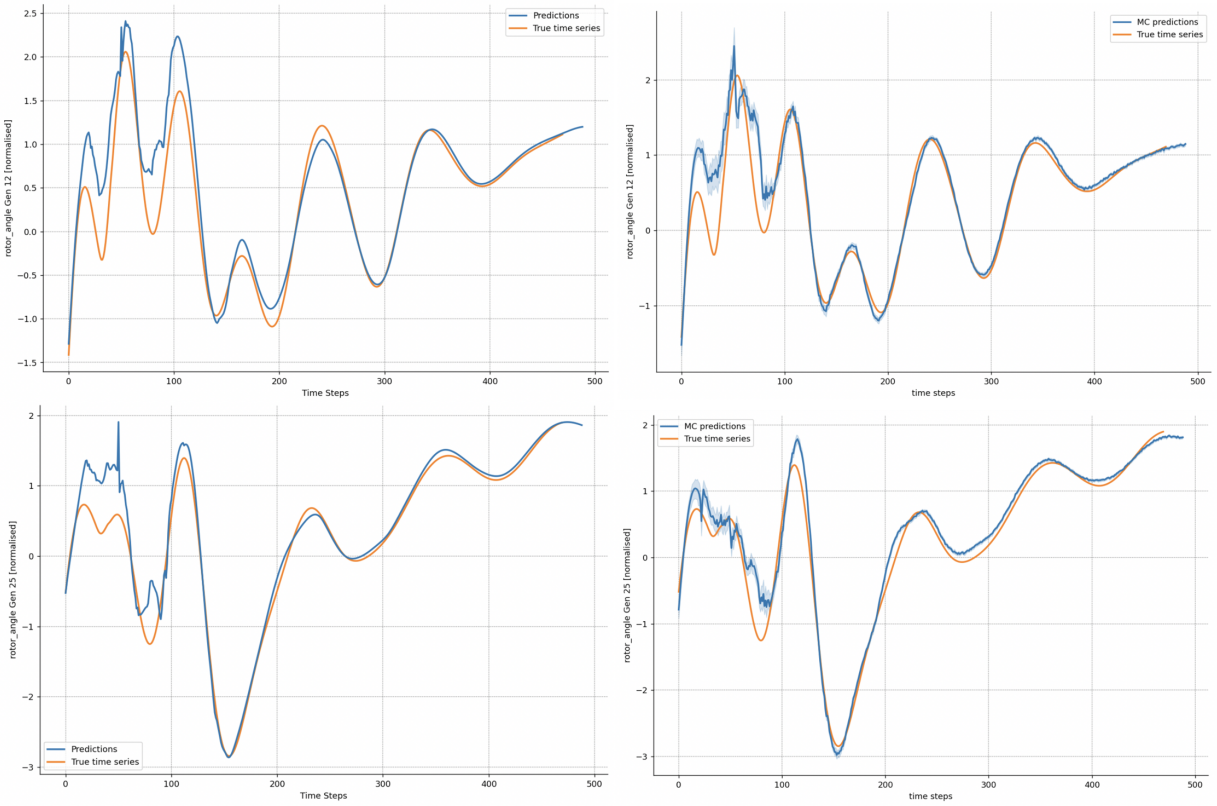

Figure 21: Left: Generator 25 and 12 predictions and actual values for an entire time series, Right: The same time series and generators with Monte-carlo sequences, 95% confidence interval

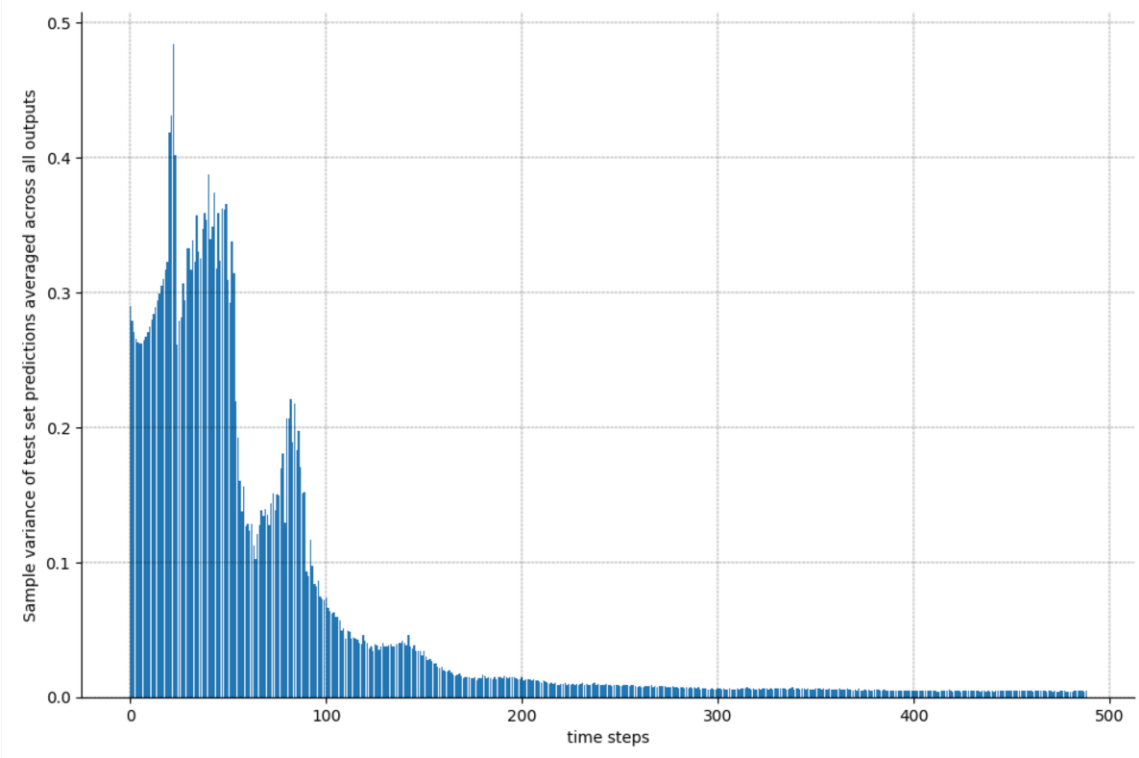

Figure 22: The average sample variance for each time step in the test set across all system generators

This again, is likely due to dataset inbalance. The system is naturally damped, so the model has trained on a greater number of sequences with small swing amplitudes that critical larger swings. Extending the length of time series to train machine learning models would exacerbate this issue, however, as indicated by Jimenez (2017) it is important to forecast beyond the primary swings of the system to capture dangerous islanding effects, where the system is not technically unstable. As well as the methods mentioned in the previous section, supplementing model training with shorter time series could help address the issue.

For this case, an application of Monte-carlo dropout is useful on two fronts. First it can help guide the model training process, allowing training, architecture and data generation to be optimised until a state of uniform uncertainty is reached across the time frame of interest. Second, in an online setting, where such a model is integrated to the grids protection system it will provide an indication of uncertainty for each time step forecast. A minimum threshold of uncertainty could be defined, for which action will only be taken if the sample variance is below this point. The number of inference cycles would need to be optimised with respect to the mean accuracy during training, and the maximum allowable inference time such that prediction delays do not result in a missed critical event.

4.4 Model response to external events

Anomalous signals

The advantage of using an encoder-decoder architecture is that we can simultaneously obtain the autoencoder blocks reconstruction error with each prediction. There is no stochasticity in the autoencoders outputs, and can supplement the models confidence estimates from Monte-carlo dropout. From (Wang et al. 2019) we simulate a data-injection on Generator 100's terminal bus. Scenario (a), is a complete permutation of data for Generator 100's signals, scenario (b) is slightly more subtle, where the signals are inverted for all time steps. All other features in the dataset remain unchanged. Of course, a significant drop in test set accuracy is observed for the two scnarios, however, in an online setting, accuracy is an unknown variable until the forecasting window has passed in time. A combination of Monte-carlo dropout and reconstruction error could therefore give a real time indication as to whether a prediction should be trusted, or whether there is an unusual disturbance in the PMU signals.

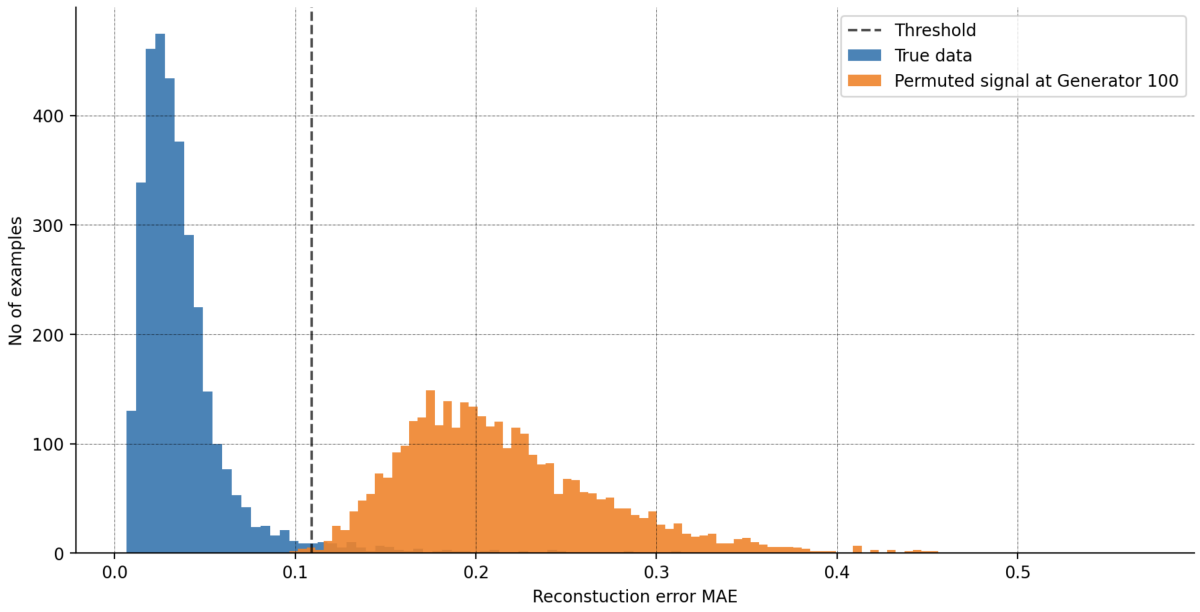

Figure 23: Scenario (a), random permutation on generator bus 100

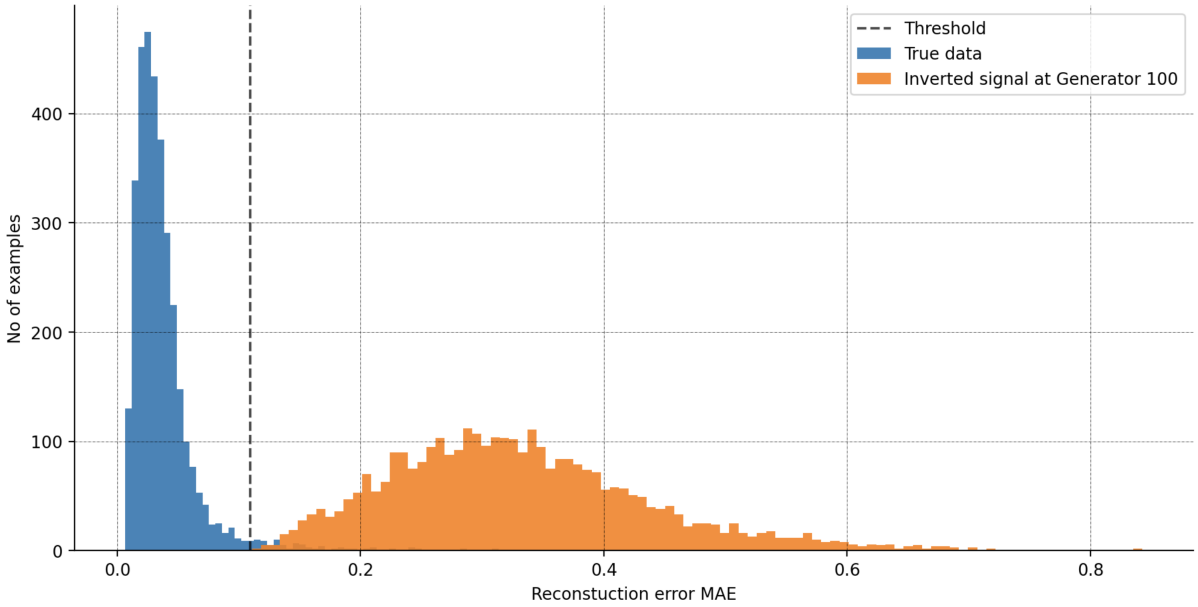

Figure 24: Scenario (b), inverted signal on generator bus 100

Setting a threshold of two standard deviations from the mean of the true testing data, in scenario (a) this approach misclassified 8 out of 3375 sequences in the permuted dataset. For scenario (b) only one sequence is misclassified.

Noise

Noise could be the result of a malicious attack on the system, or inherent in normal operations. *figure 25* highlights the impact on model performance when introduced to noise levels 1-5, and *figure 26*, the respective reconstruction error for each level.

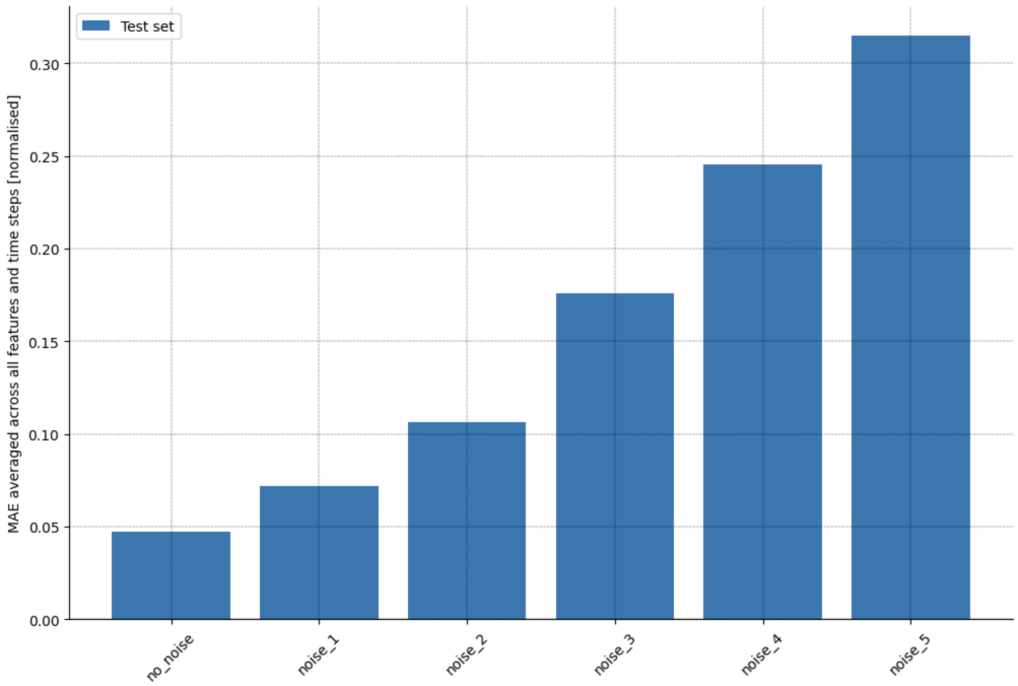

Figure 25: Test set accuracy for each noise level

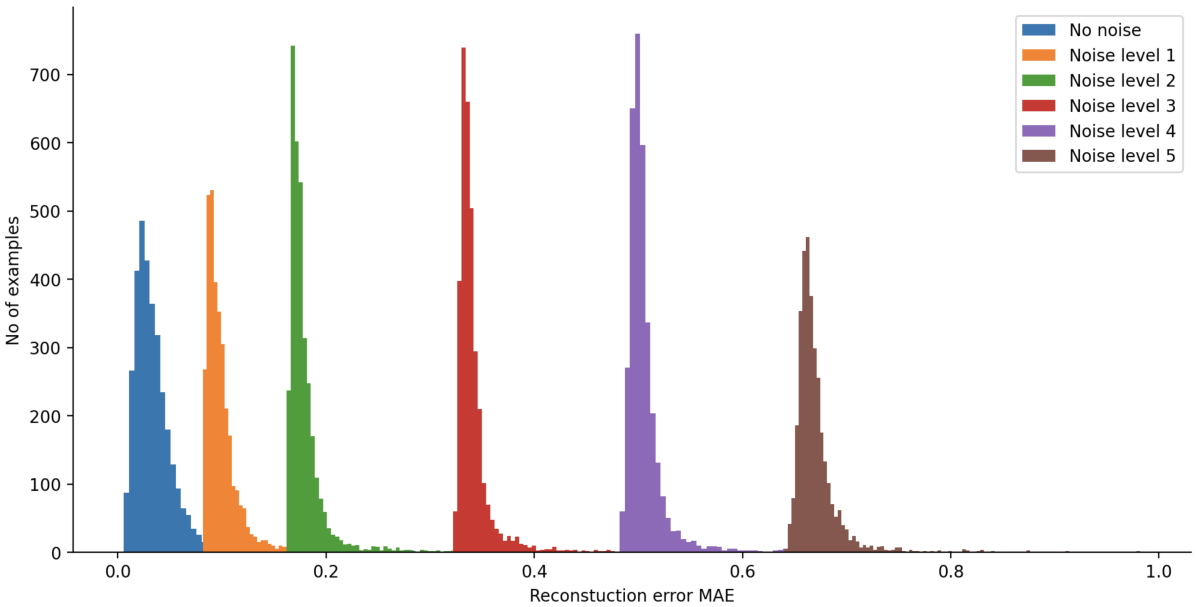

Figure 26: Reconstruction error for each noise level

From *figure 25* we see the model is highly sensitive to increased levels of noise in the signal. If real-world data is expected to be clean, or noise has a known and fixed quantity then this may not present an issue. Indeed noisy signals could then be easily classified as outwith normal operating conditions as illustrated in *figure 26*. However, if some deviation in signal quality or noise levels is to be expected in real-world scenarios, training such models directly on signals obtained from simulators such as PowerFactory may result in significant performance issues. In an attempt to mitigate the issue, and follow on from the work of Gupta et al. (2017), the encoder-decoder model was retrained with noise level 1 added to all features. This model was then re-evaluated on the same test sets shown in *25*.

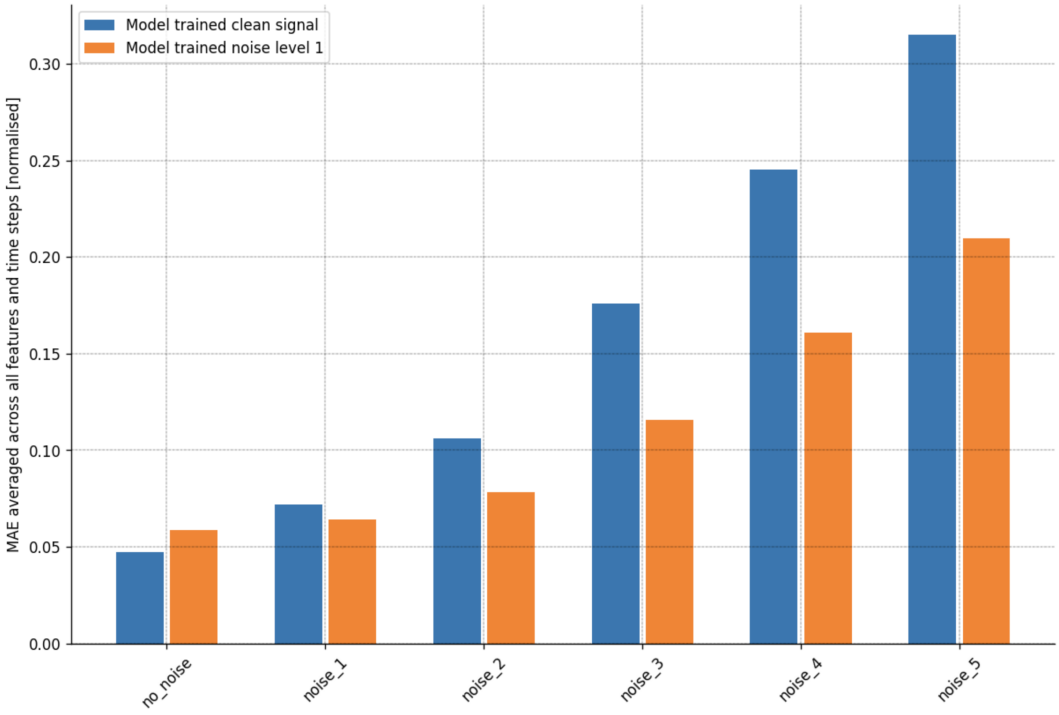

Figure 27: Comparison of test set accuracy on encoder-decoder model trained with and without noise

As seen in *figure 27*, adding a small amount of noise (mean 0, variance 0.1) to the training process increases the models robustness to increased noise levels, particularly in lower levels, with a percentage

difference in accuracy of 28% between noise level 2 and no noise test sets, compared with a 77% difference for the model trained on clean signals directly from PowerFactory. With a combination of both increasing the amount of data available for training, and increasing training epochs, the accuracy for the model trained on noise would increase further. In future study, the exact nature of noise to be expected in real-world scenarios should be identified, and models trained on data which reflects this reality to increase robustness and generalisation in real-world settings.

Topology changes

Topological changes in power grids are routine. A generation unit or line could be taken out of service for routine maintenance. Since power grids are designed to N-1 criterion, such interventions should not cause grid instability, but may increase its vulnerability. With ever increasing grid interconnection these topology changes may become more frequency, with less transparency to regional TSO's. Any machine learning model deployed in the grid must be robust to such changes in technology, and maintain a high level of performance during these times of increased grid vulnerability. Due to the size of the 118 bus model, and the high level of connectivity between zones, it makes for a good test case to evaluate model performance under such scenarios. *Figure 28* shows model performance on each out of service (OOS) scenario.

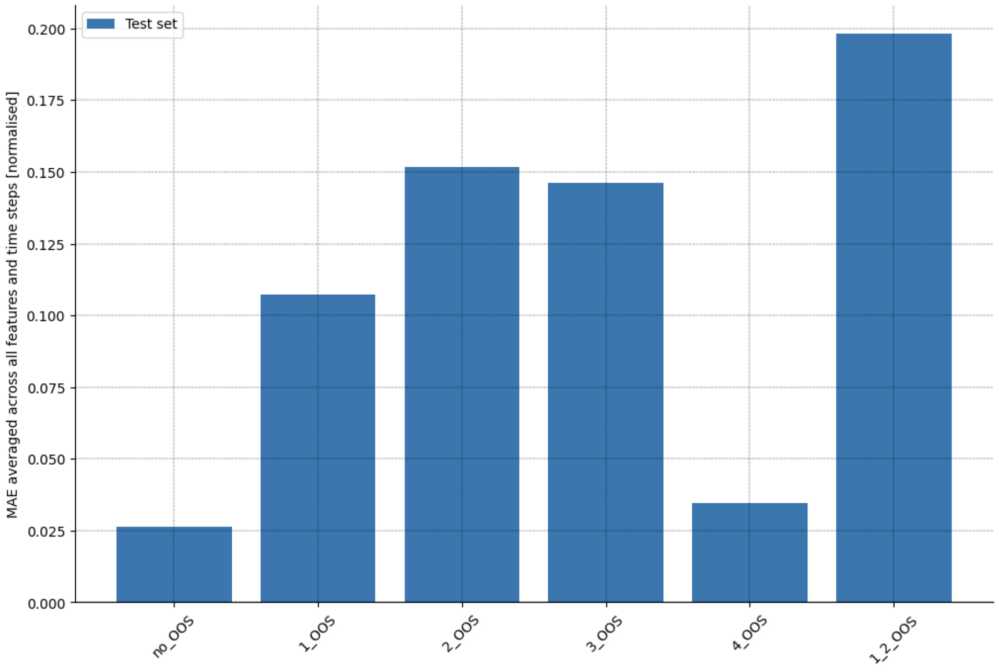

Figure 28: Test set accuracy for encoder-decoder model across topology scenarios

It is clear to see that the model has not learnt the relationship between system dynamics and system topology, but has rather learnt the dynamical relationship between generators only for the N-0 topology that it was trained on. There was a limited impact on accuracy with a line OOS in zone 4, however, zone 4 is the least connected zone in the system and it was found during previous analysis that contingencies in the zone have the lowest overall effect on the system as a whole. Conversely, zones 1-3 are highly meshed and small topological changes in these areas have a large impact on system load flow and dynamic behaviour. All studies reviewed in literature which use PMU data to predict rotor angle stability use features only pertaining to generator dynamics (as in this study), however based on this analysis, such an approach is not appropriate in capturing topological relationships in the system. Training models on features pertaining to topology will cause the dimensions of the training data to increase significantly, and future work should determine which features best capture this information, and how best to represent this information with a practical level of dimensionality.

5 Conclusion and future work

Using modern machine learning architectures, all machine rotor angles can be effectively forecast for the 118 bus system using a single model. Further, this task can be achieved using only signals obtained from PMU's. There are, however, a significant number of challenges with respect to applying such models to real-world situations. One such challenge stems from the staggering number of possible operating conditions within a modern power-grid. Sampling even a large number of contingencies cannot represent all possible dynamic responses of the system. Given this challenge, quantifying the risk of a model failing to detect a critical event remains an open question. A clear limitation to this study was the randomised nature by which contingencies were sampled, and is related to the above challenge. This sampling process resulted in a highly imbalanced dataset which had a number of negative implications. However, given how expensive the data generation process is, it is difficult to quickly generate new data samples, or conduct in depth exploratory analysis to determine these issues prior to model training and validation, and hence the data generation methodology was mostly literature driven. These lessons can be carried forward in the future to make more targeted generation strategies.

This study shows that an encoder-decoder architecture could be used to determine whether abnormal or malicious permutations are being made on PMU signals, however, the exact nature of typical operational noise needs to be well defined in order to effectively classify abnormal cases. Whilst a small number of PMU features can be used to forecast rotor angle trajectories, such features alone are not sufficient to capture the relationship between grid topology and generator dynamics. Based on the above conclusions from this study, the following areas of future work are proposed:

- To improve model generalisation on unknown extreme events, and reduce model uncertainty across the time period of interest the data generation strategy needs to be re-evaluated. Random sampling of contingencies results in a disproportionate number of stable time series. Further study should identify a probabilistic approach to contingency sampling, where only contingencies with a high probability of causing critically stable or unstable cases are initiated.
- In depth study into the most appropriate methods to model system topology for the purpose of training machine learning models should be conducted. This will improve the models ability to generalise under routine topological variations. One such approach could be through the use of graph neural networks (GNN), where a graph representation of the grid is used as a feature for each time step.
- Further work should identify typical noise profiles to be expected in normal operations. Such noise should be incorporated into the model training process.

Bibliography

Amjady, N. & Majedi, S. F. (2007), 'Transient stability prediction by a hybrid intelligent system', *IEEE Transactions on Power Systems* **22**(3), 1275–1283.

Chollet, F. (2017), *Deep Learning with Python*, 1st edn, Manning Publications Co., USA.

Crucitti, P., Latora, V. & Marchiori, M. (2004), 'Model for cascading failures in complex networks', *Phys. Rev. E* **69**, 045104.
URL: *https://link.aps.org/doi/10.1103/PhysRevE.69.045104*

Demetriou, P., Asprou, M., Quiros-Tortos, J. & Kyriakides, E. (2017), 'Dynamic ieee test systems for transient analysis', *IEEE Systems Journal* **11**(4), 2108–2117.

Gal, Y. (2016), Uncertainty in Deep Learning, PhD thesis, University of Cambridge.

Gal, Y. & Ghahramani, Z. (2015), 'Dropout as a bayesian approximation: Representing model uncertainty in deep learning'.

Goh, G. (2017), 'Why momentum really works', *Distill* .
URL: *http://distill.pub/2017/momentum*

Goodfellow, I., Bengio, Y. & Courville, A. (2016), *Deep Learning*, MIT Press.
URL: *http://www.deeplearningbook.org*

Gupta, A., Gurrala, G. & Sastry, P. S. (2017), Instability prediction in power systems using recurrent neural networks, *in* 'Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17', pp. 1795–1801.
URL: *https://doi.org/10.24963/ijcai.2017/249*

Haes Alhelou, H., Hamedani Golshan, M. E., Njenda, T. & Siano, P. (2019), 'A survey on power system blackout and cascading events: Research motivations and challenges', *Energies* **12**, 7.

Heidary, M. (2018), *Estimation of rotor angle based on operating variables measured by PMU.*

Hussain, A. (2018), 'A Day Without Power: Outage Costs for Businesses — Bloom Energy'.
URL: *https://www.bloomenergy.com/blog/a-day-without-power-outage-costs-businesses*

Jai, S. (2017), 'Power grid's mega plan to monitor supply'.
URL: *https://www.business-standard.com/article/companies/power-grid-s-mega-plan-to-monitor-supply-117072900021*$_1$*.html*

Jimenez, J. A. (2017), 'Evaluation of frequency and transient stability indicators in future power systems with high levels of wind power generation'.

Kingma, D. P. & Ba, J. (2014), 'Adam: A method for stochastic optimization'.

Kios (2015).
URL: *https://www2.kios.ucy.ac.cy/testsystems/index.php/ieee-118-bus-modified-test-system/*

Kundur, P., Paserba, J., Ajjarapu, V., Andersson, G., Bose, A., Canizares, C., Hatziargyriou, N., Hill, D., Stankovic, A., Taylor, C., Van Cutsem, T. & Vittal, V. (2004), 'Definition and classification of power system stability ieee/cigre joint task force on stability terms and definitions', *IEEE Transactions on Power Systems* **19**(3), 1387–1401.

Liu, L., Li, Y., Cao, Y., Liu, F., Wang, W. & Zuo, J. (2019), 'Transient rotor angle stability prediction based on deep belief network and long short-term memory network', *IFAC-PapersOnLine* **52**(4), 176 – 181. IFAC Workshop on Control of Smart Grid and Renewable Energy Systems CSGRES 2019.
URL: *http://www.sciencedirect.com/science/article/pii/S2405896319305117*

Nouti, D. (2017), 'Future power grids technical report'.

Petersen, N. C., Rodrigues, F. & Pereira, F. C. (2019), 'Multi-output bus travel time prediction with convolutional lstm neural network', *Expert Systems with Applications* **120**, 426 – 435.
URL: *http://www.sciencedirect.com/science/article/pii/S0957417418307486*

Rohden, M., Jung, D., Tamrakar, S. & Kettemann, S. (2016), 'Cascading failures in ac electricity grids', *Phys. Rev. E* **94**, 032209.
URL: *https://link.aps.org/doi/10.1103/PhysRevE.94.032209*

Rovnyak, S., Kretsinger, S., Thorp, J. & Brown, D. (1994), 'Decision trees for real-time transient stability prediction', *IEEE Transactions on Power Systems* **9**(3), 1417–1426.

Royal Academy (2014), Counting the cost : the economic and social costs of electricity shortfalls in the UK A report for the Council for Science and Technology, Technical Report November.
URL: *www.raeng.org.uk*

Schuster, M. & Paliwal, K. (1997), 'Bidirectional recurrent neural networks', *Signal Processing, IEEE Transactions on* **45**, 2673 – 2681.

Schäfer, B., Witthaut, D., Timme, M. & Latora, V. (2018), 'Dynamically induced cascading failures in power grids', *Nature Communications* **9**.
URL: *www.nature.com/naturecommunications*

Shuvro, R. A., Das, P., Hayat, M. M. & Talukder, M. (2019), 'Predicting cascading failures in power grids using machine learning algorithms', *51st North American Power Symposium, NAPS 2019* .

unknown (1937), 'First Report of Power System Stability', *Transactions of the American Institute of Electrical Engineers* **56**(2), 261–282.

Wang, D., Wang, X., Zhang, Y. & Jin, L. (2019), 'Detection of power grid disturbances and cyber-attacks based on machine learning', *Journal of Information Security and Applications* **46**, 42 – 52.
URL: *http://www.sciencedirect.com/science/article/pii/S2214212618305866*

Yu, J. J. Q., Hill, D. J., Lam, A. Y. S., Gu, J. & Li, V. O. K. (2018), 'Intelligent time-adaptive transient stability assessment system', *IEEE Transactions on Power Systems* **33**(1), 1049–1058.

Zhang, R., Xu, Y., Dong, Z. Y. & Wong, K. P. (2015), 'Post-disturbance transient stability assessment of power systems by a self-adaptive intelligent system', *IET Generation, Transmission Distribution* **9**(3), 296–305.

Zhu, L. & Laptev, N. (2017), 'Deep and confident prediction for time series at uber', *2017 IEEE International Conference on Data Mining Workshops (ICDMW)* .
URL: *http://dx.doi.org/10.1109/ICDMW.2017.19*

Step A6 – MSC ESDA Dissertation Ethics Declaration

Statement of Risk Assessment & Ethics Approval Requirements
Student Candidate Number [HQCJ5]: **Supervisor Name:** [Aidan O'Sullivan]: Supervisor UCL Email Address: [aidan.osullivan@ucl.ac.uk]: **Dissertation Research Proposal** [FILL IN]: • Title / Topic: Probabilistic machine learning methods to predict cascading failure events in power grids • Research Question(s) / Aims & Objectives: Can methods in machine learning provide a reproducible framework to predict, and mitigate cascading failure events? Can such methods generalize in diverse grid topologies? Can such methods provide humans greater insight into the risk associated with remedial actions in the grid? • Data & source (specify all data to be used; if none, explain why): Scandinavian 49 bus dynamic grid model, PowerFactory • Method(s) (specify all methods to be used): Deep learning – Graph Convolutional Neural Networks, LSTM networks. (Full research proposal attached)
I have read and understood **Step A1 'Does the research require a Risk Assessment?'** and: [DELETE ONE STATEMENT]: • This planned research does NOT require a risk assessment.
I have read and understood **Step A2 'Does the research require External research ethics approval?'** and: [DELETE ONE STATEMENT]: • This planned research does NOT require external ethics review.
External ethics approval is *not required* and I have read and understood **Step A3 'Is the research Exempt from the need for ethics approval?'** and: [DELETE ONE STATEMENT]: • This planned research IS EXEMPT from the need for research ethics approval.
~~The research is *not exempt* from the need for ethics approval and~~ ~~I have read and understood **Step A4 'Does the research require High Risk ethics approval?'** and:~~ ~~[DELETE ONE STATEMENT]:~~ • ~~This planned research IS deemed high risk and approval from the UCL Research Ethics Committee will be secured before the research starts.~~ • ~~This planned research is NOT deemed high risk.~~
~~The research is *not exempt* from the need for ethics approval, does not require high risk ethics approval and:~~ ~~I have read and understood **Step A5 'Does the research require ESDA low risk ethics review for questions-based methods OR BSEER low risk ethics review for other methods?'** and:~~ ~~[DELETE ONE STATEMENT]:~~ • ~~This planned research requires MSc ESDA Low Risk Ethics approval for questions based methods and approval will be secured before data collection starts.~~ • ~~This planned research requires BSEER low risk ethics approval (for other methods), which will be secured before data collection starts.~~
I confirm that: • the information I have provided is accurate to the best of my knowledge. • if the answers to any of these questions changes, I will go through this protocol again.

NEXT STEPS:

- **STUDENT:** Copy the text of the *completed* statement above into an email and email it to your supervisor.
- **SUPERVISOR:** Reply to the email confirming your approval of the completed statement.
- **STUDENT:**
  - Submit this A6 Statement (matching the one approved by your supervisor) on Moodle as your Dissertation Ethics Declaration.
  - Include this A6 Statement as a Dissertation Appendix after you have BLACKED OUT YOUR NAME & EMAIL ADDRESS so the second marker can mark anonymously.

The Dissertation mark sheet asks the second marker whether this form was filled out correctly and, if not, what % mark deduction they recommend.