Imperial College London
Department of Electrical and Electronic Engineering

Towards Intelligent Operation of Future Power System: Bayesian Deep Learning Based Uncertainty Modelling Technique

Tingqi Zhang

A thesis submitted to Imperial College London
For the degree of Doctor of Philosophy
October 2021

I hereby declare that this thesis and the work reported herein was composed by and originated entirely from me. Information derived from the published and unpublished work of others has been acknowledged in the text and references are given in the list of sources.

Tingqi Zhang (2021)

Abstract

The increasing penetration level of renewable energy resources (RES) in the power system brings fundamental changes of the system operating paradigms. In the future, the intermittent nature of RES and the corresponding smart grid technologies will lead to a much more volatile power system with higher level uncertainties. At the same time, as a result of the larger scale installation of advanced sensor devices in power system, power system engineers for the first time have the opportunity to gain insights from the influx of massive data sets in order to improve the system performance in various aspects. To this end, it is imperative to explore big data methodologies with the aim of exploring the uncertainty space within such complex data sets and thus supporting real-time decision-making in future power system. In this thesis, Bayesian Deep learning is investigated with the aim of exploring data-driven methodologies to deal with uncertainties which is in the following three aspects.

(1) The first part of this thesis proposes a novel probabilistic day-ahead net load forecasting method to capture both epistemic uncertainty and aleatoric uncertainty using Bayesian deep long short-term memory network. The proposed methodological framework employs clustering in sub-profiles and considers residential rooftop PV outputs as input features to enhance the performance of aggregated net load forecasting. Numerical experiments have been carried out based on fine-grained smart meter data from the Australian grid with separately recorded measurements of rooftop PV generation and loads. The results demonstrate the superior performance of the proposed scheme compared with a series of state-of-the-art methods and indicate the importance and effectiveness of sub-profile clustering and high PV visibility.

(2) The second part of this thesis studies a novel Conditional Bayesian Deep Auto-Encoder (CBDAC) based security assessment framework to compute a confidence metric of the prediction. This informs not only the operator to judge whether the prediction can be trusted, but it also allows for judging whether the model needs updating. A case study based on IEEE 68-bus system demonstrates that CBDAC outperforms the state-of-the-art machine learning-based DSA methods and the models that need updating under different topologies can be effectively identified. Furthermore, the case study verifies that effective updating of the models is possible even with very limited data.

(3) The last part of this thesis proposes a novel Bayesian Deep Reinforcement Learning-based

resilient control approach for multi-energy micro-grid. In particular, the proposed approach replaces deterministic network in traditional Reinforcement Learning with Bayesian probabilistic network in order to obtain an approximation of the value function distribution, which effectively solves Q-value overestimation issue. The proposed model is able to provide both energy management during normal operating conditions and resilient control during extreme events in a multi-energy micro-grid system. Comparing with naive DDPG method and optimisation method, the effectiveness and importance of employing Bayesian Reinforcement Learning approach is investigated and illustrated across different operating scenarios. Case studies have shown that by using the Monte Carlo posterior mean of the Bayesian value function distribution instead of a deterministic estimation, the proposed BDDPG method achieves a near-optimum policy in a more stable process, which verifies the robustness and the practicability of the proposed approach.

Acknowledgements

I would like to thank my supervisor Prof.Goran Strbac for giving me the valuable opportunity to chase this cutting-edge research topic. He guides me with his professional experience and insight thoughts of power system and encourages me to make more contributions.

Specifically, I would like to thank Dr.Mingyang Sun for the discussions, guidance, and supervision during my PhD life. I also would like to appreciate Dr.Fei Teng and Dr.Jochen Cremer for the advises and valuable discussion.

Special thanks to Dr.Zihang Dong, Dr.Cheng Hu, Dr.Yu Sang, Dr.Xi Zhang, Dr.Xin Xiang, Dr.Min Yu, and Mr.Wangkun Xu, Mr.Zilin Feng for their excellent discussions not only in the field of research but also in my daily life.

I also thank all my friends worldwide: Thank you all for having faith in me.

At last, there is no word in this world that I can describe my appreciation to my family in China. You are my final support in the toughest time of this long journey.

Contents

List of Tables

List of Figures

Nomenclature

T_{sample} Number of Monte Carlo Sampling

$\widetilde{\mathbb{E}}$ Expectation

$\mathcal{N}$ Gaussian Distribution

b Neural Network Bias

x Neural Network Input

y Neural Network Output

η Neural Network Activation Function

σ Noise

ω Neural Network Weights

EVs Electric Vehicles

ANN Artificial Neural Network

BDAC Bayesian Deep Auto-Encoder

BDDPG Bayesian Deep Deterministic Policy Gradient

BDL Bayesian Deep Learning

BDLSTM Bayesian Deep Long Short-Term Memory

CBDAC Conditional Bayesian Deep Auto-Encoder

CNN Convolution Neural Network

DDPG Deep Deterministic Policy Gradient

DL Deep Learning

DSA Dynamic Security Assessment

KL Kullback-Leibler

LSTM Long Short-Term Memory

MC Monte Carlo

MDP Markov Decision Process

MILP Mixed-integer Linear Programming

NN Neural Network

OCs Operating Conditions

PDF Probability Density Function

HPs Heat Pumps

PMUs Phasor Measurement Units

RES Renewable Energy Source

RL Reinforcement Learning

RNN Recurrent Neural Network

SP Stochastic Programming

SSA Static Security Assessment

TDS Time-Domain Simulations

TSOs Transmission System Operators

VI Variational Inference

Chapter 1

Introduction

1.1 Research Background and Current Challenges

Before the concept of modern smart grid techniques, the energy consumption of most developed countries was mainly from fossil fuels with less proportion of supplementary, e.g. nuclear or hydro powers. Carbon dioxide emission from those traditional resources thus are the major cause of changing climate. In recent decades, climate change and energy security have become major topics for discussion around the world. Lots of efforts have been made in order to align regimes worldwide to agree on targets of carbon emissions reduction and also to explore more advanced alternative technologies with the aim of allowing the incorporating of lower carbon emissions. A remarkable breakthrough is the Paris Agreement in 2015, and since then, there was a global consent on carbon emissions reduction in a joint effort. For instance, 20% of the EU energy consumption is expected to be from renewable energy sources (RES) by 2020, which is further expected to have a greenhouse gas emission reduction of 80% in 2050. As a result, the EU electricity system would be significantly decarbonised by 2030, with the contribution from the increased penetration level of RES, and also from the increased share of electricity demand, which is driven by the large-scale implementation of multi-energy sectors and transport sectors (e.g. heat pumps (HPs), electric vehicles (EVs), etc.).

Traditionally, power system is designed to send energy from large power generators to large demand with flexible and controllable units that have relatively lower uncertainty level. Operational flexibility thus is expected to be provided mainly by conventional generators, which are flexible enough to match variations in demand and are able to adjust its output in response to various operating scenarios. However, in the future, the shifting to low carbon electricity system would be characterised by its higher level of energy mix, including intermittent RES (e.g. wind and solar), the growing penetration of multi-energy sectors etc. Unlike the majority of conventional generation that could provide both energy and control services (e.g. energy balancing and network support services), RES is highly variable in time, which means the prediction would suffer from high level uncertainty and more importantly, it cannot be controlled in order to maintain normal operation of the power system. Furthermore, due to the fact that RES always exists as non-synchronous power sources, which do not contribute to system inertia, the overall system inertia will decrease. As a result, it could increase the possibility of system compromising to disturbance and thus will change the requirements for system reliability and resilience.

In other words, merging RES is a shifting process not only from the aspect of the sources of the energy (i.e. from using fossil fuel based energy toward using electricity), instead, it is the trend that power system starts using electricity at the point-of-use (i.e. more distributed), such as charging stations for EVs, PV panels on residential rooftops etc. Although we cannot guarantee how the power system operation will be like in the following decades, it is certain that it will be very different from what it was designed. Because when the system was designed, the target of the system was to transport bulk powers from generation side to demand side and more importantly, in single direction. In the future, distributed generations, energy storage devices, EVs, etc. would lead to a fundamental paradigm shift from one-directional power system to that allows multi-directional power flows. Hence, power system in the future would employ additional flexibility ancillary techniques, more electronic devices, and distributed generation. Such implementation was less anticipated in the original design stage, thus the challenge is to support the system during this paradigm shift. Furthermore, due to the massive cost of changing

physical infrastructures, it is more reasonable and realistic to find more advanced approaches to control and operate the system under this shift. In summary, when integrating RES into the power system, the unpredictability and the corresponding lack of inertia, which are driven by the nature of RES uncertainties, will bring considerable difficulties to the power system operation in many aspects. These difficulties require a comprehensive review of the current methodologies with the aim of improving the system planning (forecasting), safety operation (security rules) and decision making (real-time control).

This thesis is primarily concerned about the uncertainties brought by the RES. In order to accurately analyse the low carbon power system with uncertainties driven by RES, it is necessary to transform the existing deterministic model to probabilistic model in order to incorporate the stochastic elements. Bayesian Deep Learning (BDL) methods thus are implemented in this thesis to model the uncertainties from RES in three different task scenes: load forecasting, dynamic security assessment and resilient control. In addition to uncertainties from RES, there are also specific challenges in each topic area. Therefore, in the following sections, detailed challenges of each area will be introduced.

1.1.1 Primary Challenges in Load Forecasting

Decarbonization of electricity systems drives significant and continued investments in distributed energy sources to support the cost-effective transition to low-carbon energy systems. However, the rapid integration of distributed photovoltaic (PV) generation presents great challenges in obtaining reliable and secure grid operations because of its limited visibility and intermittent nature, which significantly diminishes the predictability of the residential net load. This effect may be further intensified by the stochasticity in onsite renewable generation injected from the macrogrid [5]. Under this circumstance, the primary challenges addressed in this area are summarized as follows:

1) ***PV Visibility***: In general, distributed PV is invisible to the distribution system operators and retailers as a result of its behind-the-meter installation, which injects additional uncertainty

into the net load and renders it harder to accurately predict, especially in the context of high PV penetration. However, with the development of advanced metering technologies, some residential customers have installed meters that can separately measure electricity consumption and rooftop PV output to make the distributed PV generation partially visible to stakeholders with fine-grained data. To this end, developing methods to fully exploit the partially visible or entirely visible PV to enhance the net load forecasting performance at the aggregated level will be one of the fundamental challenges that is investigated in this study.

2) ***Massive Stochastic Uncertainty***: For the net load at the aggregated level, uncertainty is composed of the load uncertainty and the distributed PV uncertainty, which is a more challenging task than either load forecasting or PV forecasting alone. In this case, we use the term *stochastic uncertainty (aleatoric uncertainty)* to represent the uncertainty within the net load injected from different sources such as climate variability, intermittent power generation, and aperiodic human activities. In recent years, although a number of probabilistic forecasting methods have been proposed to capture these massive amounts of uncertainties in the load or the net load, most of the existing methods can only provide the prediction interval (i.e., the upper and lower bounds), which does not give detailed information about the distribution of the forecast at each individual time step. In addition, most probabilistic forecasting models are inherently deterministic models with limited performance in explicitly capturing stochastic uncertainty. These models usually either produce a density forecast by employing the probability density function (pdf) of the residuals to the point forecast or perform post-processing to several point forecasts to generate quantiles. On the other hand, deep learning has demonstrated a fair performance in load forecasting; however, most of the existing models are not able to represent uncertainty. Consequently, it is crucial and imperative to investigate and develop a pure probabilistic deep learning model to handle the massive *stochastic uncertainty* in the net load and to provide confidence bounds for decision making.

3) ***Uncertainty in the Model***: *Model uncertainty (epistemic uncertainty)* refers to the uncertainty in the model parameters and the model structure. Beyond the aleatoric uncertainty, *model uncertainty* is also a critical part of uncertainty in the task of probabilistic net load

forecasting to indicate how much uncertainty the model has about its outputs. Among a vast number of potential model structures and parameters, it is important to understand how much the selected combinations might be able to accurately predict the net load under different conditions (e.g., seasons, weekends/weekdays and social factors). In the remainder of this part of work, we will illustrate in detail how the proposed Bayesian deep learning-based method can effectively handle the aforementioned challenge.

1.1.2 Primary Challenges in Dynamic Security Assessment

The penetration of RES not only injects massive uncertainties but also increases the complexity in the context of the power system modelling and operation. Dynamic Security Assessment (DSA) for the future power system is expected to be increasingly complicated with the higher level penetration of RES and the widespread deployment of power electronic devices, which drive new dynamic phenomena. As a result, the increasing complexity and the severe computational bottleneck in real-time operation encourage researchers to exploit machine learning to extract offline security rules for the online assessment. However, traditional machine learning methods lack in providing information on the confidence of their corresponding predictions. A better understanding of confidence of the prediction is of key importance for Transmission System Operators (TSOs) to use and rely on these machine learning methods. Specifically, from the perspective of topological changes, it is often unclear whether the machine learning model can still be used. Hence, being aware of the confidence of the prediction supports the transition to using machine learning in real-time operation. Under this circumstance, the primary challenges addressed in this topic are summarized as follows:

(1) ***The lack of ability of confidence awareness:*** Numerous applications of machine learning approaches in the area of power system have been investigated in the last decades. Despite a promising performance in various tasks, one of the fundamental limitations that restrict the practical implementation is that the existing models do not have the ability of interpretability. More specifically, most of the current methods cannot capture uncertainties

and thus fail to express confidence. From the perspective of TSOs, how to make decisions under the uncertainties of the power system? The complicated reality implies that simply developing the modelling technique to improve its accuracy may not be enough in practice. The TSOs need additional information that shows whether or quantifies how the model is confident about its output (i.e. confidence information). To this end, model uncertainty (epistemic uncertainty) reflects the uncertainty in the model parameters, and the model structure becomes vital. For safety-critical applications, it is also of significant importance to capture the confidence information so that the abnormal data points, which are different from training data sets, could be accurately detected. However, the state-of-the-art machine learning methods usually model the uncertainties by rigidly simulating noises and thus can hardly be explained as an epistemic learning process (i.e. uncertainty can be explained away given enough data)[6]. Hence, what we really need is a more advanced model with the real ability of confidence awareness. The influence from massive injected uncertainties thus can be alleviated, and more importantly, TSOs could be offered the flexibility of system operation in the decision-making process.

(2) ***Multi-contingency issue:*** The N-1 security criterion provides a preventive standard for system safety operation. From the perspective of machine learning, the significantly enriched database brings the opportunity of training a better model. However, challenges occur as well that more advanced modelling technique is required in order to make full use of the abundant data. In prior works, the authors in [7] treat the multi-contingency issue as independent tasks, and they set up the DTs model for each contingency. However, when deep learning methods are considered to improve the performance further, this strategy will be challenged with a series of critical issues. One of the most serious is that the total workload of the hyperparameters tuning task grows significantly and becomes nearly infeasible, especially in the context of a real-world large scale system. In contrast, the proposed method in [8] employed one-hot coding so that the contingency label can be included within the training data. It is inspired by the theory of multi-task learning, and the model is expected to learn not only correlations among individual contingencies but also the ability to distinguish the difference within a single model. The work thus has the benefits of using one single deep learning model rather than N-1 models

in terms of the computational time. However, it actually sacrifices training efficiency since the data dimension is increased. More importantly, the multi-contingency problem is essentially different from the multi-task problem since the data sets in a multi-task problem come from different sources. Therefore, using a single model could cause conflict and thus affect model performance. To this end, developing a method that fully exploits the contingency information will be investigated in this study.

(3) ***Model updating strategy when faced with system topology changes:*** Topology changes, either scheduled behaviours or those accidentally happened such as circuit breaker faults, could cause fundamental changes of system OCs. Those changes, such as line flows, might be very different for different topology, necessitating timely updating. There are various approaches to deal with system topology changes. One is to employ a real-time system topology monitoring scheme as an indicator. The other is to update the model under an experience-based timely basis. However, depending on the size of the system, the corresponding computational cost, which includes but not limited to data from new contingency domain, personnel and time etc., of updating might vary a lot. More importantly, in some cases, the current model can be kept even if the system topology changes. This brings the potential challenge that in practice, in order to obtain the operational flexibility, how could we avoid the unnecessary updating cost? In other words, could the model have self-confidence awareness about its results so that the system operator is able to aware of the proper time? Two further questions that arise and will be addressed in the following chapters thus are: (1) How does the data size contribute to model updating? (2) What would happen if the model performance is already good enough?

1.1.3 Primary Challenges in Resilient Control

The intermittent nature of RES and the increasing incorporating of multi-energy elements bring massive uncertainties thus increases the complexity in the context of the power system modelling and operation. Intensified effects could be identified due to the concerns of enhancing resilience on local power network and to reduce the import of conventional fuels in response to the security

of energy supplies[9]. Under this circumstance, the primary challenges addressed in this field are summarized as follows:

(1) ***The paradox between resiliency and the de-carbonizing electricity system:***

In the foreseeable future, increasingly larger power transfers across the network would potentially exacerbate system vulnerability against natural disasters since system infrastructures tend to operate near their designed capacity limit. Hence, the chances of system exposed to large disturbances are escalated, resulting in a drop of system resilience and potentially to large and prolonged blackouts[9]. Under the current system operation and design philosophy, not only a significant proportion of the existing conventional plants should be reserved as back-up capacity, there are even requirement of additional electricity infrastructure reinforcements[10][9]. In other words, the pursuing and transforming of de-carbonizing future electricity system might be established at the cost of sacrificing the power system performance in the aspects of steady and cost-effective operation. This brings the challenges that in order to enhance system resilience, power system should be able to deal with extreme events with large scale and severity level, even with low possibilities. Furthermore, the paradigm shift from redundancy in assets to intelligence also urges the need of more sophisticated control that incorporates more advanced technologies.

(2) ***The necessity of high-resolution real-time critical load restoration control scheme:***

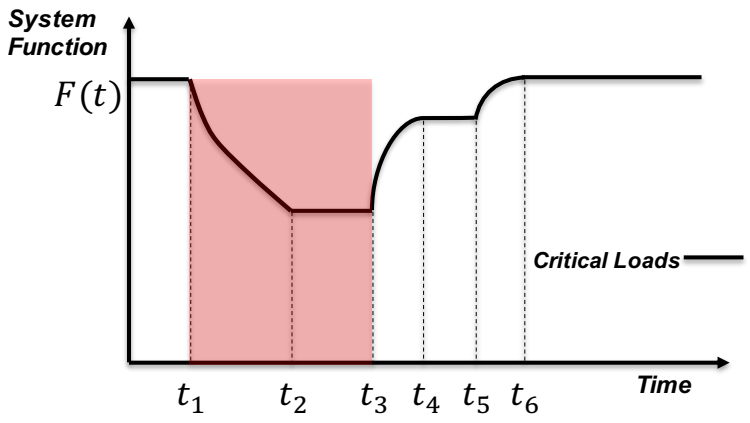


Figure 1.1: Illustrative resilience curve through an extreme event

In recent years, the awareness of resilience has been increasingly recognized and merged into the design and operation strategy for the critical infrastructures[11]. Researchers and electrical engineers have been gradually aware that it is unlikely to resist all events at all time, thus strategies beyond traditional reliability are needed to maintain the service of critical infrastructures under extreme events[12]. Critical infrastructures are facilities that could support economic, business, and social activities, including the assets, services, and systems[13]. Hence, any loss of the critical infrastructures could result in massive costs or even casualties, which means power system has the obligation to support these critical infrastructures regardless the costs (e.g. imagine a scene when an earthquake happens during a surgery in the hospital, or power outage when people are stuck in the subway facing the risk of suffocated to death).

The conceptual resilience curve in Fig. 1.1 shows the process when power system goes through an extreme event. For instance, t_1 is the point when extreme events happen. System function $F(t)$ denotes the capability of the system to provide services, thus in this case, power system critical demand can be regarded as system function. t_3 is the point when resilience controller starts to work and try to re-support the critical demand. Hence, $t_3 - t_1$ is the period that critical demand is failed to be served. In existing works, optimisation approaches are widely proposed and improved to give the control decision at t_3, but the computing time is high. For example, in [14], the $t_3 - t_1$ period is ~ 2.5 hour, in IEEE 123-node network. In addition, existing works usually assume that extreme events last for hours, e.g. 6h in [15], 12h in [16] and the critical load is always assumed to be a proportion of the total load, which is kept constant during the events e.g. 30% in [16]. In other words, existing approaches achieve the critical load restoration by analytically solve an optimisation problem with system state measurements at t_2. However, it is possible that extreme events have longer impact period, and thus the critical loads could also have its own time-changing profile with a certain level of uncertainty. Furthermore, building optimisation model needs exact system details, which could be hard for a larger system. In the existing literature, optimisation models are designed only to deal with a sets of pre-defined events, while resilience strategy could vary a lot regarding different events. Hence, it is not possible to model all the combination of fault locations, lines, losses of generations and it will

be too late to build a new model if an unexpected event happen.

Under this reality, it is crucial and imperative to investigate and develop a model-free, fast-reacting approach which is able to provide real-time control strategy. In this case, our approach does not require any knowledge inside the multi-energy micro-grid network nor any information related to the system elements.

(3) ***Robust performance under the uncertainties of extreme natural disasters and for the value function estimation***

Extreme natural disasters, such as earthquake, floods, snowstorms, or man-made cyber/physical attack, could result in cascading consequences to the power system. However, due to their stochasticity and spatio-temporal unpredictable nature, it is extremely difficult and computationally expensive to obtain accurate modelling of these events. In fact, even though an accurate forecasting is given, each time the same event could have different impacts on the power system. In other words, there are too many potential scenarios but too few data that can be collected. Hence, conventional deep learning systems might not 100% ready (fully trained) for these events.

However, for Bayesian models, less data are required to train the model. By integrating prior knowledge into learning systems, Bayesian models can effectively address the over-fitting problem by imposing a prior on hidden units or neural network parameters, even with small/insufficient data sets. In addition, though policy gradient methods in RL have earned many credits in dealing with continuous space control tasks, the stochastic nature of these problems makes deterministic value estimation difficult. As a result, Q-Learning based RL methods are found to overestimate action values under common conditions[17]. In other words, there is potential possibility that the RL agent selects the greedy, short sighted, high-risky or unreasonable actions with high Q value for the current state. From the perspective of a system operator, the obvious question will be: how to believe that the decision from the RL agent is reasonable? The key to the question lies in that whether the model has a comprehensive exploration process over the entire value function domain so that it could extract more information from the environment as the experience accumulates. Therefore, it is reasonable to think about estimating a distribution

of value function rather than in a deterministic fashion by fitting the value function with a Bayesian Neural Network. By using the posterior mean of this distribution of value estimation, we are expecting to achieve a more stable learning process and significantly better results against a traditional deterministic value function estimation.

1.2 Contributions

To address the presented research challenges, this thesis makes the following original contributions, which are summarised as follows:

- The study in load forecasting work makes the following original contributions:

  (1) A clustering-forecasting-aggregation probabilistic day-ahead net load forecasting strategy is proposed to make full use of smart meter data and partially visible PV output data.

  (2) Bayesian theory and deep LSTM networks are combined to generate aggregated level probabilistic net load forecasts with the target of capturing both epistemic uncertainty and aleatoric uncertainty. To the best of the authors' knowledge, this is the first attemption to exploit Bayesian deep learning for net load prediction.

  (3) A comprehensive comparison with a series of state-of-the-art methods is conducted. The superior performance of the proposed scheme is demonstrated with respect to both the deterministic and probabilistic forecasting results. Additionally, it is shown that the forecasting performance can be effectively enhanced in the context of high PV visibility.

- In the second part of the work, a Bayesian deep auto-encoder based methodological framework is proposed, which is able to solve multi-contingency issue and provide confidence information. Key contributions of this part can be summarized as follows:

  (1) A confidence-aware machine learning framework for DSA of the large-scale electrical system is proposed. To the best of the authors' knowledge, this is the first work that achieves confidence awareness by exploiting Bayesian deep learning in the DSA problem.

(2) The concept of conditional training is introduced. The proposed framework thus enhances the performance when facing multi-contingency issue within a single model.

(3) A confidence-oriented model updating strategy is proposed. The proposed strategy only requires small sample data to update the model.

(4) A series of comprehensive case studies are conducted. The superior and robustness performance of the proposed method is demonstrated and compared with other state-of-the-art approaches, which is based on different system topology.

- Original contributions in the third part of this work are:

1) A Bayesian DRL-based real-time decision making scheme has been proposed, which is designed to deal with the resilient operation of multi-energy micro-grid system. During the resilience mode, the target is to keep only essential loads served. During normal condition, the proposed RL approach is able to help the TSO to achieve a near optimal real-time control with minimum system operation costs.

2) Bayesian Deep Learning theory and Reinforcement Learning are integrated to generate real-time system control strategy with the aim of capturing uncertainties and avoiding value function estimation during the training process in the multi-energy micro-grid system. To the best of the authors' knowledge, this is the first work to exploit Bayesian Deep Reinforcement Learning in the area of system resilient control.

3) A series of comprehensive case studies are conducted, which consider the uncertainties in the extreme events. For example, the extreme events are assumed to have longer duration period and their own time-changing profiles. Compared with the state-of-the-art methods, the superior performance and the robustness of the proposed approach are studied and analysed through various operating scenarios.

1.3 Thesis Structure

This thesis has seven chapters in order to illustrate the relevant works, which are summarised as the follows.

In Chapter 2, a brief introduction of deep learning and reinforcement learning is given. Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) as good examples of deep learning network structures are introduced. In terms of reinforcement learning, classic methods such as Q-learning, DQN, DPG and DDPG are briefly introduced. The chapter finishes by explaining the necessity of using Bayesian Deep Learning as probabilistic uncertainty modelling technique.

In Chapter 3, a novel probabilistic day-ahead net load forecasting framework that captures both epistemic uncertainty and aleatoric uncertainty is illustrated. The proposed methodological framework employs clustering in sub-profiles and considers residential rooftop PV outputs as input features to enhance the performance of aggregated net load forecasting. Numerical experiments have been carried out based on fine-grained smart meter data from the Australian grid with separately recorded measurements of rooftop PV generation and loads. The results demonstrate the superior performance of the proposed scheme compared with a series of state-of-the-art methods and indicate the importance and effectiveness of subprofile clustering and high PV visibility.

In Chapter 4, a novel Conditional Bayesian Deep Auto-Encoder (CBDAC) based security assessment framework to compute a confidence metric of the prediction is illustrated. This informs not only the operator to judge whether the prediction can be trusted, but it also allows for judging whether the model needs updating. A case study based on IEEE 68-bus system demonstrates that CBDAC outperforms the state-of-the-art machine learning-based DSA methods and the models that need updating under different topologies can be effectively identified. Furthermore, the case study verifies that effective updating of the models is possible even with very limited data.

In Chapter 5, Bayesian Deep Reinforcement Learning algorithm as an advanced real-time control scheme is investigated, which is designed to provide both the energy management during normal operation conditions and resilient control during extreme events in a multi-energy micro-grid system. The proposed Bayesian model explores the entire action domain and thus could identify more cost-effective control strategies during normal operation period meanwhile maintain the system critical service during the resilience period. The effectiveness and importance of employing Bayesian reinforcement learning approach is investigated across different operating scenarios and compared with naive DDPG method and other optimisation method. Case studies have demonstrated that the proposed BDDPG method manages to learn a near-optimum policy in a more stable process, which demonstrates the superior performance and highlight the contribution of the proposed Bayesian deep learning-based method.

Chapter 6 contains the concluding remarks and the potential further work directions.

1.4 List of Publications

- M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang, "Using Bayesian Deep Learning to Capture Uncertainty for Residential Net Load Forecasting," IEEE Trans. Power Syst., vol. 35, no. 1, pp. 188–201, 2020.

- T. Zhang, M. Sun, J. L. Cremer, N. Zhang, G. Strbac, and C. Kang, "A Confidence-Aware Machine Learning Framework for Dynamic Security Assessment," IEEE Trans. Power Syst., p. 1, 2021.

- T. Zhang, M. Sun, X. Zhang, G. Strbac, and C. Kang, "A Bayesian Deep Reinforcement Learning-Based Real-Time Control Scheme to Enhance Power System Resilience Considering Uncertainties in Extreme Events," To be submitted.

Chapter 2

Bayesian Deep Learning: A Probabilistic Uncertainty Modelling Framework

2.1 Deep Learning and Reinforcement Learning

2.1.1 Deep Learning

Deep learning is a particular type of machine learning that enables a model to establish complicated non-linear functions from high-dimensional features and obtain domain knowledge with experience and data.

One of the basic deep learning models is called deep feed-forward network. For example, $y = f(x)$ can be used to denote a classifier that maps an input vector x to a category y. A feed-forward network defines a mapping function $y = f_\theta(x)$ and trains the parameters θ that result in the best function approximation. These models are called feed-forward since information flows from x, through the intermediate layers that define f, and finally reach the output y. In other words, the model is regarded as a acyclic graph describing how the functions are stacked together. For

example, assuming the network has three layers with each layer denoted by functions f^1 , f^2, and f^3 respectively. These functions are connected in a chain, which forms $f(x) = f^3(f^2(f^1(x)))$. In this case, f^1 is called the first layer of the network, the second layer f^2 is connected directly to the first layer, and the final layer f^3 is called the output layer. The overall length of the chain determines the depth of the model and the dimensionality of these hidden layers determines the width of the model.

Feed-forward networks are important since they form the basis of many other applications, which are designed to fulfill various functions such as Convolution Neural Network (CNN) for the classification tasks or Recurrent Neural Network (RNN) for the regression tasks.

Recurrent Neural Network

Recurrent neural networks are a family of neural networks for processing sequential data, i.e. $x^{(1)}, \ldots, x^{(\tau)}$, with the time step index t ranging from 1 to τ.

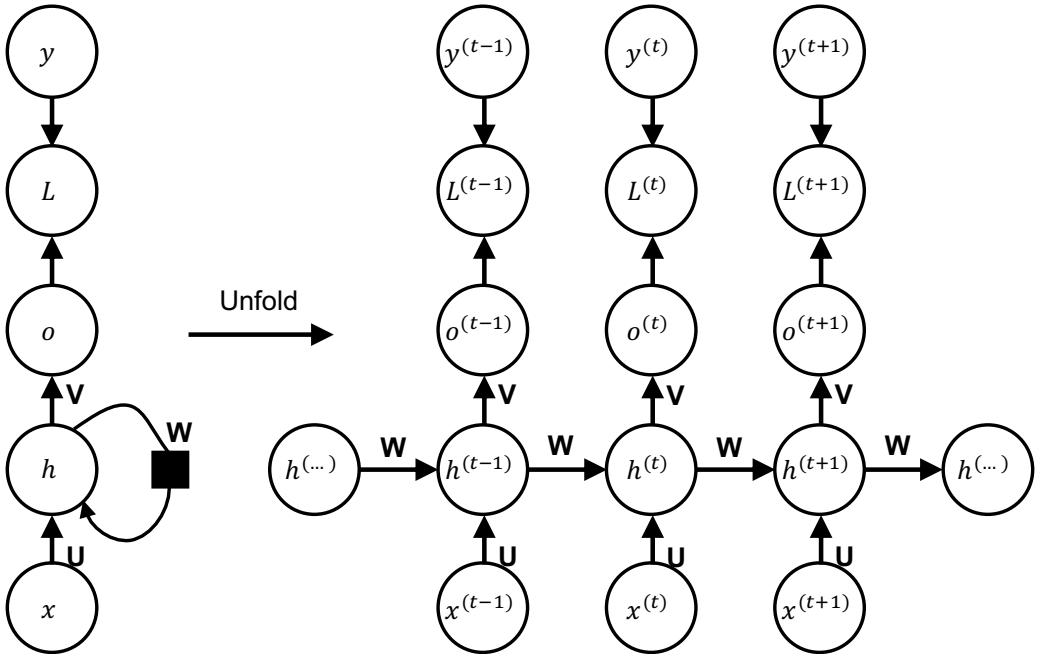


Figure 2.1: The computational graph of RNN[1]

Fig.2.1 illustrates the computational graph of a recurrent network. The network maps an input sequence of x to a sequence of output o. Loss function L evaluates the accuracy with

respect to the corresponding target sequence y. Entering from the input unit, the RNN has its input-to-hidden connections parametrized by a weight matrix U. The recurrent hidden-to-hidden connections are represented by a weight matrix W. Finally, hidden-to-output connections are denoted by a weight matrix V.

The definition of forward propagation equations for the RNN in the left side of Fig.2.1 is given as follows[1]:

$$a^{(t)} = U \cdot x^{(t)} + W \cdot h^{(t-1)} + b \tag{2.1}$$

$$h^{(t)} = tanh(a^{(t)}) \tag{2.2}$$

$$o^{(t)} = c + V \cdot h^{(t)} \tag{2.3}$$

$$\hat{y}^{(t)} = softmax(o^{(t)}) \tag{2.4}$$

where the parameters b and c are the bias vectors along with the weight matrices U, V and W respectively and the activation function is assumed to be hyperbolic tangent. In addition, it is also assumed that the output is discrete, i.e. the RNN is used to predict words or characters. Softmax operation is thus implemented in order to post-process the discrete output o and eventually to obtain a vector $\hat{y}$ of normalized probabilities.

The unfolding process in the right side of Fig.2.1 results in rolling scheme of parameters updating across the deep network structure. For instance, consider the classical form of a dynamical system with an external input $x^{(t)}$

$$s^{(t)} = f_\theta(s^{(t-1)}, x^{(t)}) \tag{2.5}$$

where $s^{(t)}$ is called the state of the system at time t.

Equation 2.5 demonstrates that the definition of $s^{(t)}$ at current time t depends on the previous

state $s^{(t-1)}$. Hence, in terms of a finite number of time steps τ, the recurrent network structure can be unfolded through the direction of time stream by applying equation 2.5 $\tau - 1$ times. By doing this, it has yielded an expression that can now be represented by a traditional directed acyclic computational graph.

Convolution Neural Network

Convolutional network is another specially designed neural network structure with the aim of processing grid-like data. Examples include time-series data, which can be regarded as a 1-D grid sampling at regular time intervals, and image data, which can be regarded as a 2-D grid of pixels. Convolution is an operation on two functions of a real-valued argument.

$$s(t) = \int x(a)w(t-a)da \tag{2.6}$$

The convolution operation is typically denoted with an asterisk:

$$s(t) = (x * w)(t) \tag{2.7}$$

In convolution operation equations, the first term is often referred to as the input, and the second term as the kernel. The output is sometimes referred to as the feature map.

In addition, when dealing with data from real environment, time is always discrete at regular intervals rather than continuous. Hence, it might be more realistic to assume that the time index t can take only integer values. Assuming that x and w are also defined only on integer t, the discrete convolution can be defined as follows:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.8}$$

Fig.2.2 has illustrated an example of convolution applied to a 2-D tensor.

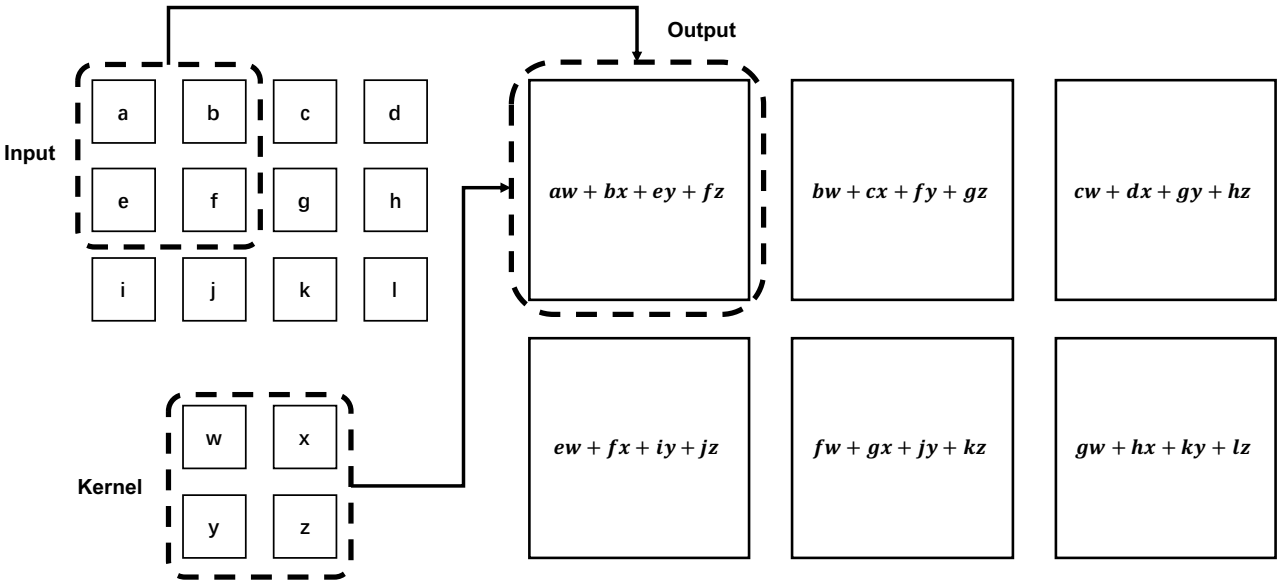


Figure 2.2: The rolling window computation of CNN[1]

In traditional feed-forward networks, each output unit is connected and interacts with each input unit. Hence, this interacting process is described by applying matrix multiplication. On the other hand, in terms of convolutional network, typically it has sparse interactions, which is also referred to as sparse connectivity or sparse weights. This is realised by establishing a kernel that is smaller than the input, i.e. as indicated in the small window in the right-top of Fig.2.2. Take the image processing as an example. The input image might have more than millions of pixels. However, with small kernels that could occupy only hundreds of pixels, more detail, meaningful features such as the edges of a picture can be detected. This means that less parameters are required to be stored, leading to both reduction of the memory requirements and improvement of its computational efficiency. For instance, if the input and output dimensions are m and n respectively, matrix multiplication would require $m \times n$ parameters, and the training algorithms would take $O(m \times n)$ operations for every training epoch. If the number of connections that each output may have is limited to k, then this sparsely connected approach would require only $k \times n$ parameters and $O(k \times n)$ operation per training epoch. By doing this, it allows the network to efficiently describe complicated interactions between large-scale variables by constructing such interactions from small kernels that each describe only sparse interactions[1].

2.1.2 Reinforcement Learning

Reinforcement Learning Background

In the context of RL, an agent takes actions sequentially to interact with the environment following a pre-defined rule, which is designed to maximize the cumulative reward (or minimize the pre-designed cumulative cost). In general, RL is described as a Markov Decision Process (MDP) which includes: 1) a state space S; 2) an action space A; 3) a state transition probability $p(s_{t+1}|s_t, a_t)$, which satisfies the Markov property, i.e., $p(s_{t+1}|s_t, a_t) = p(s_{t+1}|s_1, a_1, ..., s_t, a_t)$; 4) a reward function r: $(S, A) \rightarrow R$ and 5) a policy function $\pi(s_t) = a_t$, which is used to govern the agent when choosing the action a_t at a certain state s_t.

When the agent interacting with the MDP, a series of states, actions and rewards are generated: $s_1, a_1, r_1, s_2, a_2, r_2, ..., s_t, a_t, r_t$, ... over t time steps. The cumulative return $R = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$ is the sum of discounted reward where $\gamma \in [0, 1]$ is the discount factor. It is used to represent how an agent is going to balance the effect from current and future states. The Q-value function $Q_\pi(s_t, a_t) = E[R|s_t, a_t, \pi]$ represents the estimation of the cumulative return given an action a_t, at state s_t, and following the policy π from the selected states on-wards. An optimal policy can be obtained from the optimal Q value $Q^*(s_t, a_t) = \max Q_\pi(s_t, a_t)$ by selecting the action that contributes the highest Q-value at each state.

It is notable that RL can be sub-categorized into model-based and model-free algorithms. In general, model-free algorithms are more popular in the literature since they require less computational burden and also do not need to develop a model of the environment, which makes it more convenient for the researchers[18]. More importantly, the integration of RL with Deep Learning has shown significant performance improvement, which has enabled them to be applied in many tasks such as auto-driving vehicles. In the following sections, several state-of-the-art model-free RL algorithms will be introduced.

Q Learning

Q Learning as one of the classic RL algorithms constructs Q-value function in a recursive format according to the Bellman equation[19]:

$$Q_\pi(s_t, a_t) = E[r_t + \gamma Q_\pi(s_{t+1}, \pi(s_{t+1}))] \tag{2.9}$$

It is an off-policy, model-free RL algorithm, which has discrete state action spaces. The agent's policy is stored in its Q table and the updating procedure of the Q-value after taking action a_t at state s_t with the observed reward r_t and the resulting state s_{t+1} follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \tau\delta_t \tag{2.10}$$

$$\delta_t = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \tag{2.11}$$

where $\tau \in [0, 1]$ is the learning rate, $\gamma \in [0, 1]$ is the discount factor, δ_t represents the TD error for the estimation of the Q-value function, and $r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$ represents the target Q-value at time step t.

Q-learning has shown merits of simplicity. However, as the number of considered (A, S) dimensionality increases, the size of the Q table also grows, which will eventually makes the problem intractable.

In addition, the resolution of the discretization of the state and action spaces affects the performance of Q-learning significantly. A higher resolution could result in better performance but is more likely hard to implement in practice. Therefore, the implementation of Q table becomes its bottleneck and limits its practical application when it comes to continuous action, multi-dimensional state space problems.

Deep Q Network: DQN

In order to address the aforementioned limitations of Q-learning, researchers propose to employ DNN so as to approximate the Q-value function[20]. The employed DNN is parameterized by θ, which is shown in equation 2.12:

$$Q(s_t, a_t) \approx Q(s_t, a_t|\theta) \tag{2.12}$$

Comparing to the Q-learning approach, DQN could receive a continuous state s_t as input and then outputs a series of estimation of Q-value for each discrete action. The action with maximum Q value will then be selected at the given state. Similar to Q-learning, DNN is trained based on minimizing the TD error as shown in equation 2.13:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}|\theta) - Q(s_t, a_t|\theta))^2 \tag{2.13}$$

$$\theta \leftarrow \theta + \tau \nabla_\theta \mathcal{L}(\theta) \tag{2.14}$$

where N is the size of mini-batch, $\gamma \in [0, 1]$ is the discount factor and then the updating is applied to the DNN weights θ, as shown in equation 2.14. It can be observed from the above equation that unlike supervised learning, in the context of RL, there is no clear label for the data sets. Instead, RL explore and exploit when interacting with the environment and obtain feedback from the action results. In other words, there is no 'correct answers (i.e. labels)' from the perspective of RL agent. It further implies that the training process of DNN could be unstable as the RL agent balances itself of exploration and exploitation.

In order to solve this issue, two innovative techniques are proposed: (1) the use of the experience replay and (2) the use of target network[20]. For instance, the reply buffer can store the stochastic experiences, which is shuffled in order to obtain the data independence. The DNN is

trained with randomly sampled mini-batch from the replay buffer, towards the direction of TD error descending. On the other hand, a target network is constructed with its parameters fixed temporarily during training, hence the learning process can be stable.

DQN has shown good performance when dealing with continuous state space. However, there are still limitations since DNN are trained to generate discrete Q value estimations rather than continuous actions. Therefore, the challenge of finding a precise action that indicates the highest Q value at a given state still exits, which drives the researchers to find more advanced RL methods.

Deep Policy Gradient: DPG

As mentioned above, DQN still has the drawbacks when it is required to perform multiple, continuous actions simultaneously. To this end, researchers have proposed several solutions. In [21] and [22], the authors use a DNN (with parameters denoted by ϕ) to estimate a stochastic policy π_ϕ, which represents the probabilities of selecting an action at the given state s_t instead of estimating the Q value directly. As claimed in [23], the probability distribution of the agent's action is usually modelled by a Gaussian distribution $N(\mu, \sigma^2)$. Hence, the task of the DNN is to predict the mean μ and the variance σ^2 of it, which is referred to as a Gaussian Policy. It is also notable that DPG now has the advantage to deal with multi-dimension action space by sampling the Gaussian policy from each action dimension k, as shown in equation 2.15.

$$\pi_\phi(a_{t,k}|s_t) \sim N(\mu_k, {\sigma_k}^2), \forall k = 1, ..., |A|. \tag{2.15}$$

In terms of the training process, the parameter ϕ is updated directly according to the policy gradient, whose target is to maximize the expected reward, as shown in equation 2.16[22]. Monte Carlo samples $x_i, p(x_i|\theta)$ are used in order to calculate the gradient at the end of an episode, with the samples $x_i, r(x_i)$ collected during the game as trajectory[22].

$$\nabla_\phi \mathcal{J}(\pi_\phi) = E_{\pi_\phi}[\nabla_\phi \log \pi_\phi(a|s) R] \tag{2.16}$$

Comparing to DQN approach, it can be found that the action probability in fact can provide the randomness for the model to explore more states while DQN only generates deterministic decisions. However, the limitation of DPG lies in the low sampling efficiency in its learning process since the integration is applied to both state and action space. In addition, the gradient estimation often has high variance, resulting in slow convergence[24].

Deep Deterministic Policy Gradient (DDPG)

With the aforementioned limitations of DQN and DPG, DDPG is proposed with an actor-critic architecture, which is realized by employing two DNNs with different purposes [25].

For instance, the actor network is denoted as μ_θ, with θ indicating its parameters. Given a state s_t, it outputs a continuous action $a_t = \mu_\phi(s_t)$ with respect to the policy. The critic network is denoted as Q_ϕ, with ϕ indicating its parameters. Given a state s_t and an action a_t at time t, the critic network outputs the Q value function $Q_\phi(s_t, a_t)$, which performs as evaluating the action decision. Similar as before, the target of DDPG is also to search for the optimal policy that could give the maximum Q value. Among the previously introduced RL algorithms, greedy search is often implemented in order to obtain the maximum of the Q-value, i.e., $\mu(s_{t+1}) = \arg\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$. However, when it turns to the multi-dimensional continuous action spaces, greedy policy tends to be intractable. Therefore, the training and updating process of DDPG with the actor-critic structure achieves the target through the following procedures:

(1) For a given state s_t, actor network μ selects the action a_t from the policy and exploration noise. A random Gaussian noise $N(0, {\sigma_t}^2)$ with exponential decay is usually chosen and added to the actor's output $\mu_\theta(s_t)$. By doing this, the RL agent acquires the ability of maintaining an appropriate balance between exploration and exploitation. In other words, the agent is

able to learn both the best decision from the given information and more information from the environment.

(2) Action a_t is feed forward to the critic network, then reward r_t and new state s_{t+1} are observed, which will be stored in an experience pool (i.e. reply buffer) as transition tuple (s_t, a_t, r_t, s_{t+1}). It is notable here that since actions and states are generated by interacting with the environment sequentially, the samples of experiments are temporally correlated. As a result, the reply buffer must be shuffled before a mini-batch is generated. In addition, when the reply buffer is full, more recent experiences will replace the more previous ones.

(3) Once a random mini-batch is generated, gradients should be calculated in order to update the networks. For instance, in order to train and update the critic network, the following loss function should be minimized:

$$\mathcal{L}(\phi) = \frac{1}{N}\sum_{t=1}^{N}(r_t + \gamma Q_\phi(s_{t+1}, \mu_\theta(s_{t+1})) - Q_\phi(s_t, a_t))^2 \tag{2.17}$$

where $r_t + \gamma Q_\phi(s_{t+1}, \mu_\theta(s_{t+1}))$ represents the target Q value at time step t. In order to stabilize the updating process, a target network is constructed, which is denoted by $\hat{Q}_\phi(s_t, a_t)$. The target critic network is initialized by copying the parameters from the online critic network. The parameters of the target network are updated by slowly tracking the parameters of the online network. Expressed in equation as:

$$\phi' \leftarrow \tau\phi + (1-\tau)\phi' \tag{2.18}$$

with $\tau \ll 1$.

(4) To update the actor μ_θ, the sampled policy gradients are used and expressed as follows:

$$\nabla_\theta \mathcal{J}(\mu_\theta) = \frac{1}{N}\sum_{t=1}^{N} \nabla_A Q_\phi(s_t, a_t)_{a_t=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t) \tag{2.19}$$

where $\nabla_A Q_\phi(s_t, a_t)_{a_t=\mu_\theta(s_t)}$ indicates the direction towards the higher Q value, while gradient $\nabla_\theta \mu_\theta(s_t)$ indicates the direction towards the optimal action at the given state s_t. Following the same rational, a target actor network is constructed, which is initialized with the online actor network parameters. Soft update is implemented as before, which is shown in equation below:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \tag{2.20}$$

with $\tau \ll 1$.

The overall framework of DDPG algorithm is given in Fig.2.3.

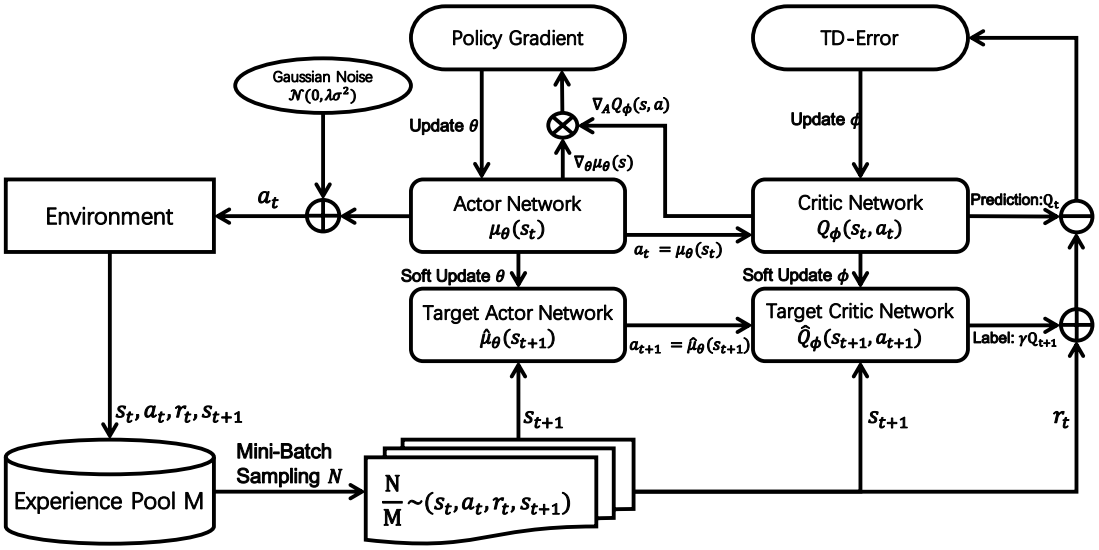


Figure 2.3: DDPG overall workflow

2.2 Why Bayesian Deep Learning?

Deep learning has demonstrated state-of-the-art performance in a vast number of tasks; however, as illustrated in [26], it still suffers from a series of limitations that need to be investigated and resolved, including *1) uninterpretable black boxes; 2) being weak in its uncertainty representation; 3) being data hungry.* To address these challenges, in this part, the benefits and rationale of employing BDL to conduct different tasks will be explained.

2.2.1 Inherently probabilistic model

BDL is inherently a probabilistic model that allows a deep learning model to represent uncertainty. Unlike traditional neural networks, which have fixed parameters once trained, Bayesian neural network parameters (i.e., the weights and bias) are expressed as distributions. As a result, the Bayesian model generates its result by directly sampling from its parameters rather than adding noise to the output or setting up multiple input scenarios. In other words, the Bayesian model is fundamentally probabilistic rather than deterministic in nature. To further illustrate the difference between BDL model and traditional models, an simple example is given below.

We start from a simple example of $y = \sin x$ function. In Fig. 2.4, BDL model is used. The yellow line represents the actual value and the red line is mean value of the prediction. $x \in [-5, 5]$ is the training range. The blue shaded area is formed by the direct sampling of BDL model, thus is the probabilistic predictive distribution of the results.

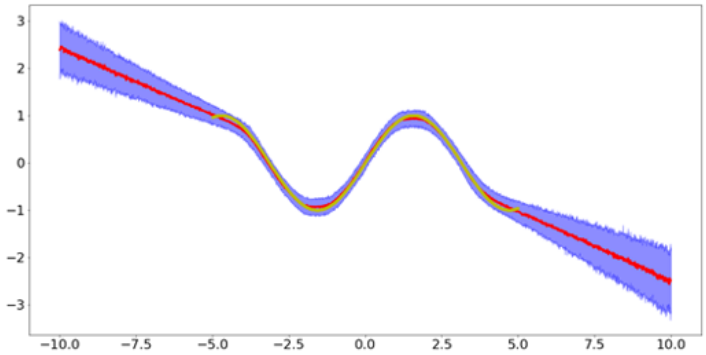


Figure 2.4: The probabilistic prediction results of BDL model.

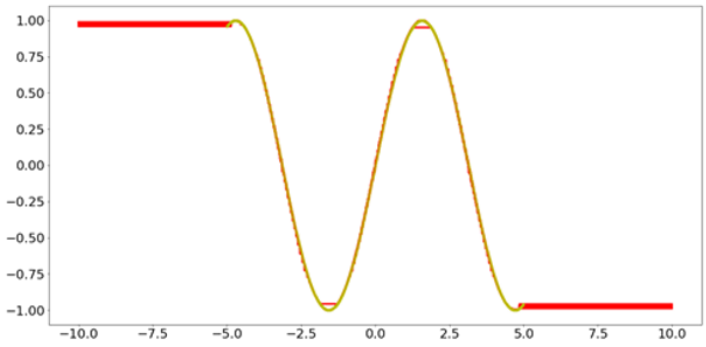


Figure 2.5: The probabilistic prediction results of GBQR model.

In contrast, we choose an ensemble of GBQR models in Fig. 2.5 to provide a traditional fashion of probabilistic machine learning prediction. The technique uses an ensemble of deterministic models, meaning that each model in the ensemble produces a point estimate rather than a distribution. It is notable that usually, it works by independently training many models of slightly different parameters on the same data-set, or different random sub data-sets upon the same model. However, these type of models only output a single value of the their prediction. The so called 'probabilistic interval' is usually formed by an upper bound and a lower bound, which is simply the maximum and minimum values of a series of prediction. In classification models, the probability vector obtained at the end of the softmax function is often erroneously interpreted as model confidence. As we can see from Fig. 2.5, when the input testing data is beyond the training range, the model only provide a wrong prediction and no probabilistic distribution can be correctly generated.

To emphasize the point of this research and to clarify the ambiguity: The author believe that the true 'probabilistic' ability of a model comes from the model itself, which gives the model the ability to generate the distribution of its prediction results. Correspondingly, this generated distribution should reflect the epistemic learning process (i.e. the model enhances its performance upon the defined tasks with growing knowledge from the given information) of model. There are already exploration about this type of property of the model, which is often interpreted as capturing model uncertainties (or epistemic uncertainty). For example, in Fig. 2.4, when the input testing data is out of the training range ($x \leq -5$, $x \geq 5$), the model cannot recognise the data, thus gives uncertainty on its results rather than simply provide a wrong prediction.

2.2.2 Captures both epistemic uncertainty and aleatoric uncertainty

The word epistemic comes from 'episteme', which is a Greek word for 'knowledge'. In other words, epistemic uncertainty is 'knowledge uncertainty'. Aleatoric comes from the Latin 'aleator', or 'dice player', which means that aleatoric uncertainty is the 'dice player's' uncertainty.

In addition, epistemic and aleatoric uncertainties could also be referred to as reducible and irreducible uncertainties respectively, since epistemic uncertainty can be reduced with more knowledge learned from the input data, while aleatoric uncertainty cannot be explained away even with more data(the stochasticity of a dice roll cannot be reduced by observing more rolls)[27]. In this thesis, epistemic and aleatoric uncertainties can also be regarded as same expression as model uncertainty and data uncertainty.

The example in previous section is a simple case that the test data are out of the range of training distribution. Another typical and more practical example would be: given several pictures of dogs and cats as training data, then let the model to examine whether the uploaded picture from a user is a dog or a cat. In this case, it is easy to raise the question that what should happen if a user uploads a photo of a tiger? The model has been trained on pictures of dogs and cats, and is expected to distinguish between them. But the model has never seen a tiger before, thus a picture of a tiger would lie outside of the data distribution where the model was trained on. This illustrative example can be extended to more serious settings, such as scenes that an autonomous vehicle system has never been trained on, or operating conditions that never occur in a power system. A possible desired behaviour of a model in such cases would be to return a prediction, but also with extra information that the input data lies outside the training range, which is regarded, in this research, as capturing both epistemic uncertainty and aleatoric uncertainty.

In the literature, most of the existing Bayesian deep learning approaches can merely capture either the model uncertainty or the stochastic uncertainty alone [27]. However, Bayesian neural network can simultaneously capture the *model uncertainty* and the stochastic uncertainty. More specifically, the model uncertainty is captured by placing a prior distribution over the model's weights; then, the posterior can be approximated via an inference algorithm. Hence, the model uncertainty is represented by the shape of the distribution of the weights. In other words, the BDL attempts to capture how much those weights change based on the input data. For safety-critical applications, it is of significant importance to capture the epistemic uncertainty to understand examples that are different from the training data. Furthermore, *stochastic*

uncertainty is captured by placing a distribution with small variance (usually Gaussian random noise) over the output and, therefore, the model learns the variance in the noise as a function of different inputs [6].

2.2.3 Explainable under probability theory

Traditional deep neural networks use their neurons to memorize the information inside the training data, which implies that the parameters in traditional neural networks have no physical meaning, and thus, their values can be arbitrary. Nonetheless, Bayesian networks calculate their outputs with Bayesian theory to render the parameters explainable so that the network has the ability to 'feel' certain or uncertain about its result. In particular, BDL can calibrate the model and the prediction uncertainty to obtain smart systems that know exactly what they do not know. For example, in net load forecasting, when the predictor encounters input features with extremely different or unreasonable values than it has encountered before (i.e., out-of-distribution test data), the predictor can give an answer (e.g., the quantified model uncertainty) indicating that it does not know how to handle this new data-set, rather than giving a wrong forecast like the current deep learning models. Another typical example is the Q-value function approximation in RL. This is the function that estimates the quality of different actions an agent can take. In other words, Q-value directly determines the quality of the learning process of an RL agent. In the existing approaches, epsilon greedy search is often employed where the agent selects the action with maximum Q-value. But with uncertainty information, an agent can avoid greedy, short sighted, high-risky or unreasonable actions with high Q value, which means the agent's behaviour becomes reasonable and explainable, thus could be regarded as balancing when to exploit and when to explore.

2.2.4 Reliable performance with small data-sets

Many real-world tasks have limited amounts of data (small data) that conventional deep learning systems cannot address. These tasks include choosing what data to learn from, or exploring an

agent's environment efficiently. Data collection is sometimes expensive such as the labelling work of individual examples by an expert, or time consuming such as the time-domain simulations in a dynamic security assessment problem of multiple times. More importantly, the amount of labelled data required could increase significantly with the growing of the complexity level of the problem.

One possible solution to this task could be active learning [28]. For instance, model itself could choose what unlabelled data to be its next training set, which would be most informative. Then asking an external party (e.g. a time-domain simulator) for a label only for these selected data points. An acquisition function is usually required in order to select the correct data points to be labelled, which is based on the ranking of potential informativeness. Among various acquisition functions, many of them employ model uncertainty in order to decide on their potential informativeness[29]. Hence, it could be concluded that by using BDL, the amount of required data could be further reduced, while still maintaining good performance.

However, even without the aforementioned learning framework, BDL still shows its great potential in dealing with small data-sets. For example, in terms of the net load forecasting problem, although a large number of measurements can be collected through advanced smart metering systems, for classical deep learning, which usually requires millions of training samples, the performance may still be limited due to the lack of data. However, for BDL, less data are required to make accurate forecasting. By integrating prior knowledge into learning systems, BDL can effectively address the over-fitting problem by imposing a prior on hidden units or neural network parameters, even with small/insufficient data-sets. In other words, BDL enables the network to achieve automatic model complexity control and structure learning with the benefits of the built-in implicit regularization [30].

Chapter 3

Using Bayesian Deep Learning to Capture Uncertainty for Residential Net Load Forecasting

3.1 Introduction

Global decarbonization is expected to be achieved by increasing the penetration of renewable energy sources (RES) and by the electrification of the heating and transport sectors. Although uncertainty at the higher system level is more likely to be traded off, in future power systems, the predictability of aggregate loads still tends to be limited by the significant uncertainties arising from climate variability, electric vehicles, distributed renewable energy generation, energy efficiency, and demand response [31]. Accurate probabilistic net load forecasting is thus of great importance to capture these massive uncertainties, contributing to the operation and planning of future smart, low-carbon energy systems.

In the literature, conventional forecasting approaches focus on point/deterministic forecasting (e.g., [32, 33, 34, 35]). In particular, the pioneering work of deterministic short-term load forecasting in [34] and [35] effectively addresses the challenges of peak load estimation at

system level and bus load prediction, respectively. However, in view of capturing uncertainties injected from different resources, point forecasting is becoming obsolete because it can provide only a single output per time step for the decision-making process that heavily depends on expected values. In other words, the ideal forecasting models should be capable of representing uncertainty via quantiles, intervals or probability density functions for numerous applications such as probabilistic load flow analysis, reliability planning, and optimal bidding in electricity markets. In general, according to reference [31], the probabilistic forecasts can be obtained via (i) feeding multiple scenarios to a deterministic model [36, 37, 38, 39]; (ii) developing novel probabilistic forecasting models [40, 41, 42, 43, 44]; (iii) post-processing the point forecasts [38, 45]; or their combinations [46]. In particular, the novel hybrid probabilistic load forecasting model proposed in [46] was developed based on an improved wavelet neural network trained by a generalized extreme learning machine to provide the load forecast with a probabilistic interval while capturing the forecasting model and data noise uncertainties. A comprehensive review on probabilistic electric load forecasting challenges and modern probabilistic forecasting models is presented in [31].

Despite the rich literature focusing on electric load forecasting, very few studies aim to predict the net load (i.e., the load traded between the microgrid and the utility grid), which is important for smart grid management and operations as well as resource allocation and electricity market participation with respect to common coupling between interconnected grids [5]. Different from traditional load forecasting, net load refers to the total energy consumption partially supported by the distributed renewable energy, such as local PV generation, thus injecting additional uncertainty, especially when the PV generation is partially visible or completely invisible. Therefore, the researchers in [47] designed a novel method to address the invisible high PV penetration, where the net load profile is decomposed into PV output, actual load and residual, which are predicted in turn. Additionally, additive and integrated net load forecast models are compared in [5], and the results demonstrate that the forecasting errors of net load and solar are cointegrated with a common stochastic drift. Other works, such as [48], propose very short-term forecasting using a complex-valued neural network. A neural network (NN)

with a Levenberg-Marquardt training algorithm is used in [49] to generate the feeder net load forecast.

Beyond the aforementioned studies, which are mostly based on classical statistical or ANN methods, in recent years, deep learning, as one of the cutting-edge technologies, has received widespread attention in a range of research fields [50, 51]. Regarding energy-related time-series forecasting, researchers [52] have used deep learning methods to achieve a load forecasting task and compared the performance between a conditional restricted Boltzmann machine and a factored conditional restricted Boltzmann machine. Additionally, the authors of [53] propose a novel forecasting model for short-term power load and probability density forecasting based on deep learning, quantile regression and kernel density estimation. Furthermore, another type of network structure designed for handling sequence dependence (i.e., time-series data in this case) is recurrent neural networks. The long short-term memory (LSTM) network is one powerful type of RNN structure that includes a memory cell that can retain information for long periods of time and deal with the the problem of long-term dependencies [54]. As an example, the authors of [55] and [56] have used deep LSTM networks to tackle the challenges of high volatility and uncertainty in household-level loads, showing a verified superior performance. More recent works such as [57] proposed an improved quantile regression neural network by introducing Gaussian noise into the training process. In [58], a deep residual network is proposed based on Monte Carlo dropout to achieve probabilistic forecasting. Additionally, LSTM also has some other variations, such as dilated LSTM [59] and bidirectional LSTM [60]. When dealing with the challenges of insufficient data, the authors in [61] designed a transfer training model with shared layers to perform wind farm forecasting.

Although the existing research has successfully demonstrated the superior performance of deep learning on forecasting tasks, inherently, most of the studies are actually based on deterministic models, which lack the ability to capture uncertainty. As a new probabilistic deep learning model, the concept of Bayesian deep learning (BDL), which enables a deep learning framework to model uncertainty, is becoming increasingly prevalent in computer vision, natural language processing, medical diagnostics, and autonomous driving [27]. BDL exhibits the benefits of

uncertainty representation, understanding generalization, and reliable prediction, leading to a more interpretable deep neural network through the lens of probability theory. In this part of work, a novel probabilistic net load forecasting framework is proposed based on BDL, aimed at capturing both epistemic uncertainty and aleatoric uncertainty. Note that this work will focus on the net load prediction at the aggregated level, and the proposed framework implements a clustering technique to group the residential customers and employs PV outputs as parts of input features for network training. We design case studies based on real PV generation and load data from the Australian grid. Compared with other state-of-the-art methods, the proposed approach outperforms conventional approaches, and the results show the importance of clustering and high PV visibility. To summarize, this study makes the following original contributions:

(1) A clustering-forecasting-aggregation probabilistic day-ahead net load forecasting strategy is proposed to make full use of smart meter data and partially visible PV output data.

(2) Bayesian theory and deep LSTM networks are combined to generate aggregated level probabilistic net load forecasts with the target of capturing both epistemic uncertainty and aleatoric uncertainty. To the best of the authors' knowledge, this is the first work to exploit Bayesian deep learning for net load prediction.

(3) A comprehensive comparison with a series of state-of-the-art methods is conducted. The superior performance of the proposed scheme is demonstrated with respect to both the deterministic and probabilistic forecasting results. Additionally, it is shown that the forecasting performance can be effectively enhanced in the context of high PV visibility.

3.2 Bayesian Deep LSTM Network (BDLSTM)

The appeal of a special recurrent neural network architecture, long short-term memory networks (LSTMs) [62], has been demonstrated for short-term residential load forecasting to tackle the challenges of long-term dependencies in the literature (e.g., [55]). Beyond that, the deep

architecture of LSTMs can contribute to learning highly nonlinear relationships between the input explanatory features and the output residential load data through a series of linear or nonlinear functions.

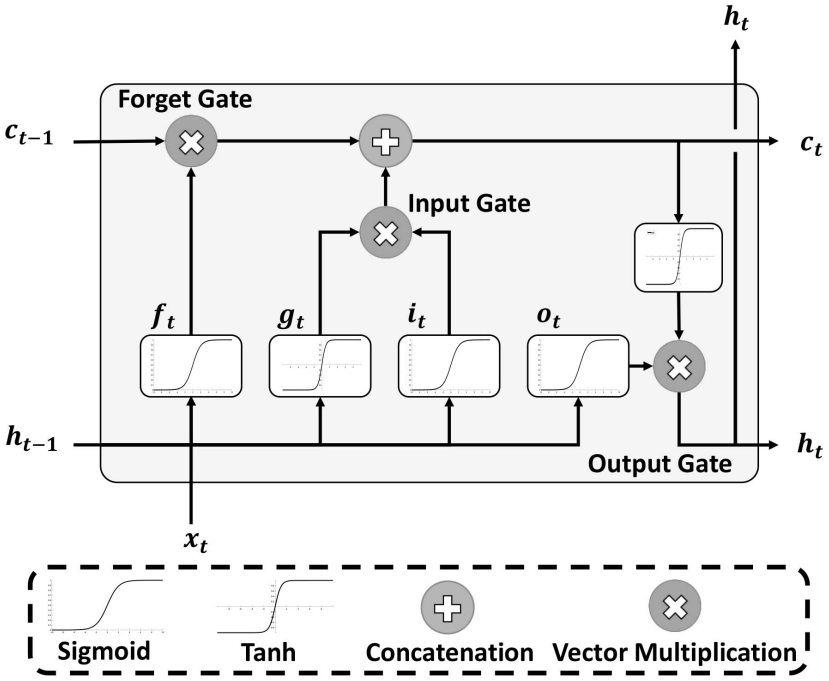


Figure 3.1: The structure of one LSTM cell.

To describe the basic architecture of the proposed Bayesian deep neural network, we first briefly introduce the structure of one LSTM cell, as shown in Fig.3.1. The inputs of the LSTM cell at one particular time step t are the previous state h_{t-1} and the current input x_t. Through four fully connected neurons f_t, g_t, i_t, and o_t, three gates are employed to fulfill the function of memory or forget information. In particular, the forget gate decides how much previous information will be transported forward, the input gate controls the aspects of new input information, and the output gate decides what will be output at this time step. In terms of the outputs, h_t is then fed into the next time step as input, which can be considered as a short-term state, while c_t decides the longer-term dependency. The overall computation is summarized in equations (3.1)-(3.4) as follows:

$$f_t = \eta\Big(W_{xf}^T \cdot x_t + W_{hf}^T \cdot h_{t-1} + b_f\Big) \tag{3.1}$$

$$i_t = \eta\left(W_{xi}^T \cdot x_t + W_{hi}^T \cdot h_{t-1} + b_i\right) \tag{3.2}$$

$$o_t = \eta\left(W_{xo}^T \cdot x_t + W_{ho}^T \cdot h_{t-1} + b_o\right) \tag{3.3}$$

$$g_t = \tanh\left(W_{xg}^T \cdot x_t + W_{hg}^T \cdot h_{t-1} + b_g\right) \tag{3.4}$$

Given the values of the three gates at next time step, the values of next state c_t and h_t are calculated by the equations $c_t = f_t \cdot c_{t-1} + i_t \cdot g_t$ and $y_t = h_t = o_t \cdot \tanh(c_t)$, respectively, where W_{xf}^T, W_{xi}^T, W_{xo}^T, and W_{xg}^T represent the weights of each input vector x_t, W_{hf}^T, W_{hi}^T, W_{ho}^T, W_{hg}^T are the weights of each previous short-term state h_{t-1}; b_f, b_i, b_o, and b_g are the biases for each of the four. It is notable that at the initial stage, b_f should be initialized with 1 instead of 0 to avoid forgetting everything from the beginning of training. Overall, through the above novel structure, LSTM handles time series by storing the important input information in the long-term state, preserving it for as long as required and retrieving it when necessary.

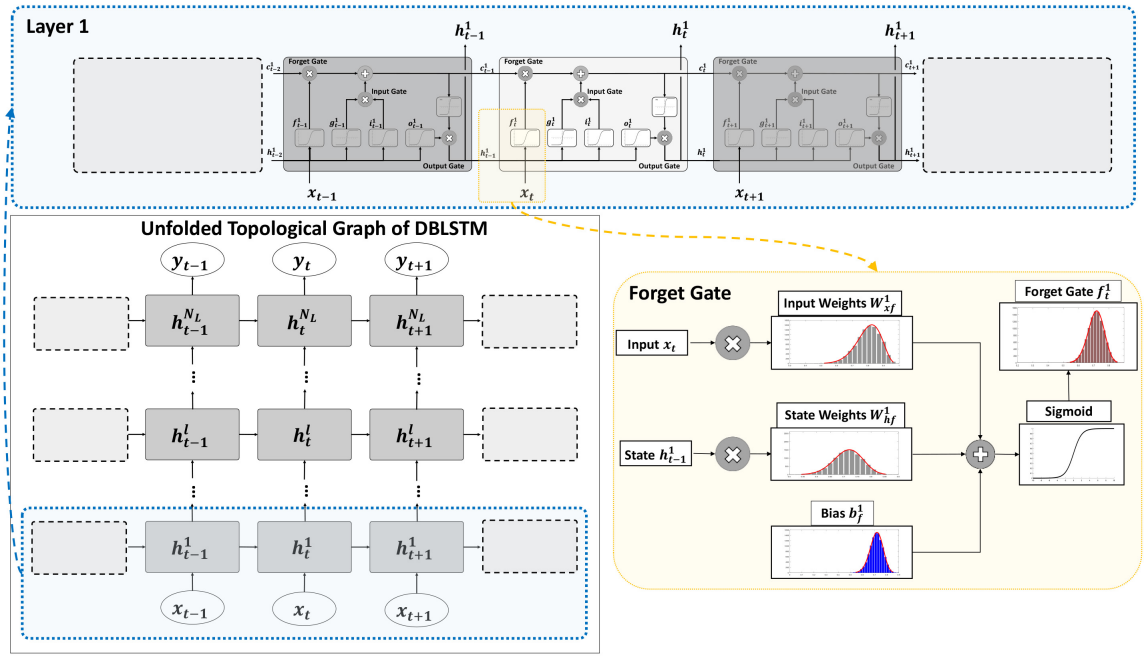


Figure 3.2: An example of the proposed BDLSTM network with a zoomed-in plot of the *forget gate* at time step t in the first layer.

To obtain uncertainty estimates in deep learning, most of existing Bayesian deep learning approaches can capture only the epistemic uncertain or the aleatoric uncertainty alone, which are usually formalized as probability distributions over the model parameters or the model

outputs [27]. To jointly capture the epistemic uncertain and the aleatoric uncertainty, a Bayesian deep LSTM network (BDLSTM), casting deep LSTMs as Bayesian models, is proposed, which retains the model architecture, placing a prior distribution upon the network weights and bias parameters of LSTMs and then inferring a posterior distribution over the given data.

Let $X_{train} = [x_1, \cdots, x_{T_{train}}]^T \in \mathbb{R}^{T_{train} \times d_x}$ and $Y_{train} = [y_1, \cdots, y_{T_{train}}] \in \mathbb{R}^{T_{train} \times d_y}$ denote the input data and output label, respectively, of the BDLSTM model that needs to be trained, where T_{train} is the total number of training data points, and d_x and d_y represent the dimensions of the input and the output, respectively. The primary target of a deep LSTM network can be formalized as identifying the optimal parameters W of a function $y = f^W(x)$ that are likely to have generated the outputs (i.e., the actual net load). In this case, $f^W(\cdot)$ represents the deep LSTM network with N_L layers and model parameters are denoted by $W = [W_1, ..., W_{N_L}]$, which is a set of random variables. An example of the Bayesian LSTM cell of the proposed BDLSTM network is given in Fig. 3.2 with a zoomed-in plot of the *forget gate* at time step t in the first layer. Detailed mathematical illustrations are given as follows.

The Epistemic Uncertainty

In general, the *epistemic uncertainty* (model uncertainty) comprises structure uncertainty and model parameter uncertainty. More specifically, structure uncertainty refers to the uncertainty in selecting the most appropriate model structure to extrapolate or interpolate the data well. Among a vast number of possible model parameters, which set of parameters should be selected to best explain the observations is uncertain, denoted by the model parameter uncertainty [27]. To capture the *epistemic uncertainty*, a prior distribution (e.g., $\mathcal{N}(0, I)$) is placed over W. In the literature, a series of studies have been carried out on prior selection (e.g., [63, 64]). In general, prior distributions can be classified into 1) non-informative prior distributions; 2) highly informative prior distributions; and 3) moderately informative hierarchical prior distributions [63].

For Bayesian deep neural networks, the prior distributions should represent the prior belief

about the distribution of the neural network parameters (weights and bias), which are difficult to be identified because the physical meaning of these parameters remains unclear. In other words, selecting the prior for the Bayesian deep learning is still an open question that needs to be further investigated by researchers. According to references [65, 27, 66], employing standard parametric distributions has been demonstrated as one of the most effective solutions when the prior belief is difficult to be identified. Therefore, in this case, we set the standard normal distribution as our prior whose zero mean can bring about the benefit of regularization [27]. It is important to note that after training the Bayesian deep neural network, the posterior distribution will be employed to generate the samples of forecasts rather than the prior distribution.

After determining the appropriate prior, the model likelihood $p(Y_{train}|f^W(X_{train}))$ is also defined as a normal distribution $\mathcal{N}(f^W(X_{train}), \sigma^2)$ with a constant noise level σ. Based on Bayes rule, the posterior $p(W|X_{train}, Y_{train})$ is calculated by

$$p(W|X_{train}, Y_{train}) = \frac{p(Y_{train}|X_{train}, W) \cdot p(W)}{p(Y_{train}|X_{train})} \tag{3.5}$$

where $p(Y_{train}|X_{train})$ is the marginal probability that cannot be estimated analytically. To this end, different inference techniques such as variational inference and Markov chain Monte Carlo (MCMC) [30] are proposed to approximate it. Note that $p(W|X_{train}, Y_{train})$ represents the posterior distribution over weights given the training data $\{X_{train}, Y_{train}\}$. Given a new input point x, the new output y, which is defined as a random variable, can be predicted by integrating

$$p(y|x, X_{train}, Y_{train}) = \int p(y|x, W)p(W|X_{train}, Y_{train})\mathrm{d}W \tag{3.6}$$

It is notable that the true posterior is usually intractable, especially for complex models (e.g., deep LSTM networks). Therefore, an approximating variational distribution $q_\theta(W)$, parameterized by θ, is employed to ensure that the optimal distribution $\tilde{q}_\theta(W)$ can well represent $p(W|X_{train}, Y_{train})$, by minimizing the Kullback-Leibler (KL) divergence between $q_\theta(W)$ and

$p(W|X_{train}, Y_{train})$ [67]:

$$KL(q_\theta(W)||p(W|X_{train}, Y_{train})) = \int q_\theta(W) \log \frac{q_\theta(W)}{p(W|X_{train}, Y_{train})} \mathrm{d}W, \tag{3.7}$$

using inference algorithms such as variational inference (VI), which is employed in this work. It is notable that it is intractable to analytically solve the optimization problem. Consequently, the objective is transformed from a KL divergence minimization problem to an Evidence Lower Bound (ELBO) maximization problem. More details regarding the employed VI algorithm can be found in the reference [27].

After obtaining the optimal distribution $\tilde{q}_\theta(W)$, the predictive distribution can be approximated by

$$p(y|x, X_{train}, Y_{train}) = \int p(y|x, W)\tilde{q}_\theta(W)\mathrm{d}W = \tilde{q}_\theta(y|x). \tag{3.8}$$

Let T_{sample} denote the number of sampled weights $\{\hat{W}_t\}_{t=1}^{T_{sample}}$, simulating the model based on the input x, the predictive mean and the predictive variance of y, which is a vector of size T_{sample}, can be approximated based on these samples. Mathematically, the predictive mean (the first raw moment) can be estimated with the unbiased estimator [27]

$$\widetilde{\mathbb{E}}[y] := \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} f^{\hat{W}_t}(x) \tag{3.9}$$

where $f^{\hat{W}_t}(x)$ represents stochastic forward passes through the model (i.e., samples). On the other hand, to obtain the predictive variance, the second raw moment needs to be estimated. Similar to the estimation of the first moment, given that $\hat{W}_t \sim \tilde{q}_\theta(W)$ and $p(y|f^W(x)) = \mathcal{N}(y; f^W(x), \sigma^2)$ for some $\sigma > 0$, we have the estimator

$$\widetilde{\mathbb{E}}[y^T y] := \frac{1}{T_{sample}} \sum_{t=1}^{T} f^{\hat{W}_t}(x)^T f^{\hat{W}_t}(x) + \sigma^2 \tag{3.10}$$

with T_{sample} samples. Finally, the predictive variance can be approximated by $\widetilde{\mathrm{Var}}[y]$ as follows:

$$\widetilde{\mathrm{Var}}[y] = \widetilde{\mathbb{E}}\big[y^T y\big] - \widetilde{\mathbb{E}}\big[y\big]^T \widetilde{\mathbb{E}}\big[y\big] := MU(x, y, W) + \sigma^2 \tag{3.11}$$

where

$$MU = \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} f^{\hat{W}_t}(x)^T f^{\hat{W}_t}(x) - \frac{1}{{T_{sample}}^2} \sum_{t=1}^{T_{sample}} f^{\hat{W}_t}(x)^T \sum_{t=1}^{T_{sample}} f^{\hat{W}_t}(x) \tag{3.12}$$

represents the epistemic uncertainty (model uncertainty), which measures how much the model is uncertain about its outputs. It is important to note that in equation (3.11), with the increasing number of observations, the term $MU(x, y, W)$ can be reduced whereas the inherent noise measured by σ^2 cannot be vanished.

The Aleatoric Uncertainty

According to the dependency between the uncertainty and the inputs, the *aleatoric uncertainty* can be further divided into *homoscedastic uncertainty* and *heteroscedastic uncertainty*[6]. For *homoscedastic uncertainty*, the observation noise parameter σ is fixed whereas in this case, we need to capture the *heteroscedastic aleatoric uncertainty* because the uncertainty varies over different periods of time when dealing with the net load. To this end, σ in equation (3.11) needs to be adapted as a function of the input x, which means it is data-dependent. Let T_{train} denote the number of training observations, and the minimization objective of the data-dependent heteroscedastic model can be expressed as follows:

$$\mathcal{L}(\theta) = \frac{1}{T_{train}} \sum_{i=1}^{T_{train}} \frac{1}{2\sigma(x_i)^2} ||y_i - f(x_i)||^2 + \frac{1}{2} \log\sigma(x_i)^2 \tag{3.13}$$

In this case, maximum a posteriori (MAP) inference is carried out to locate a single parameter, θ, rather than the distribution of the weights, leading to neglect of the model uncertainty.

The Combined Uncertainties

To combine the *epistemic uncertainty* and the *aleatoric uncertainty* in a single BDLSTM model, the most straightforward and effective way is to transform the heteroscedastic model into a Bayesian model by placing a distribution over the weights and the bias [6]. First, we need to set up a new expression for the model to split the top layers of a deep LSTM network between the predictive mean $f(x)$ and the model precision $g(x)$ to simultaneously output $\hat{y}$ and $\hat{\sigma}^2$:

$$[\hat{y}, \hat{\sigma}^2] = f^{\hat{W}}_{BDLSTM}(x) \tag{3.14}$$

where f_{BDLSTM} represents the proposed Bayesian deep LSTM network parameterized by $\hat{W} \sim q_\theta(W)$. Given that a normal likelihood is chosen to model the aleatoric uncertainty, the final loss function of the BDLSTM model can be formulated as:

$$\mathcal{L}_{BDLSTM}(\theta) = \frac{1}{T_{train}} \sum_{i=1}^{T_{train}} \frac{1}{2\hat{\sigma}_i^2} ||y_i - \hat{y}_i||^2 + \frac{1}{2} \log \hat{\sigma}_i^2 \tag{3.15}$$

Note that the loss function can consider both the model uncertainty through $\hat{y}$ and the heteroscedastic uncertainty through $\hat{\sigma}$. Finally, the predictive uncertainty $\mathrm{Var}[y]$ of the proposed BDLSTM model, consisting of both the *aleatoric uncertainty* and the *epistemic uncertainty* can be approximated by

$$\widetilde{\mathrm{Var}}[y] := \Big[\frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} {\hat{y}_t}^2 - \left(\frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} \hat{y}_t \right)^2 \Big] + \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} {\hat{\sigma}_t}^2. \tag{3.16}$$

It is important to note that, compared with equation (3.11) which has a fixed σ, the second term in equation (3.16) is data-dependent. Detailed explanations regarding the Bayesian deep learning are presented in references [27], [6].

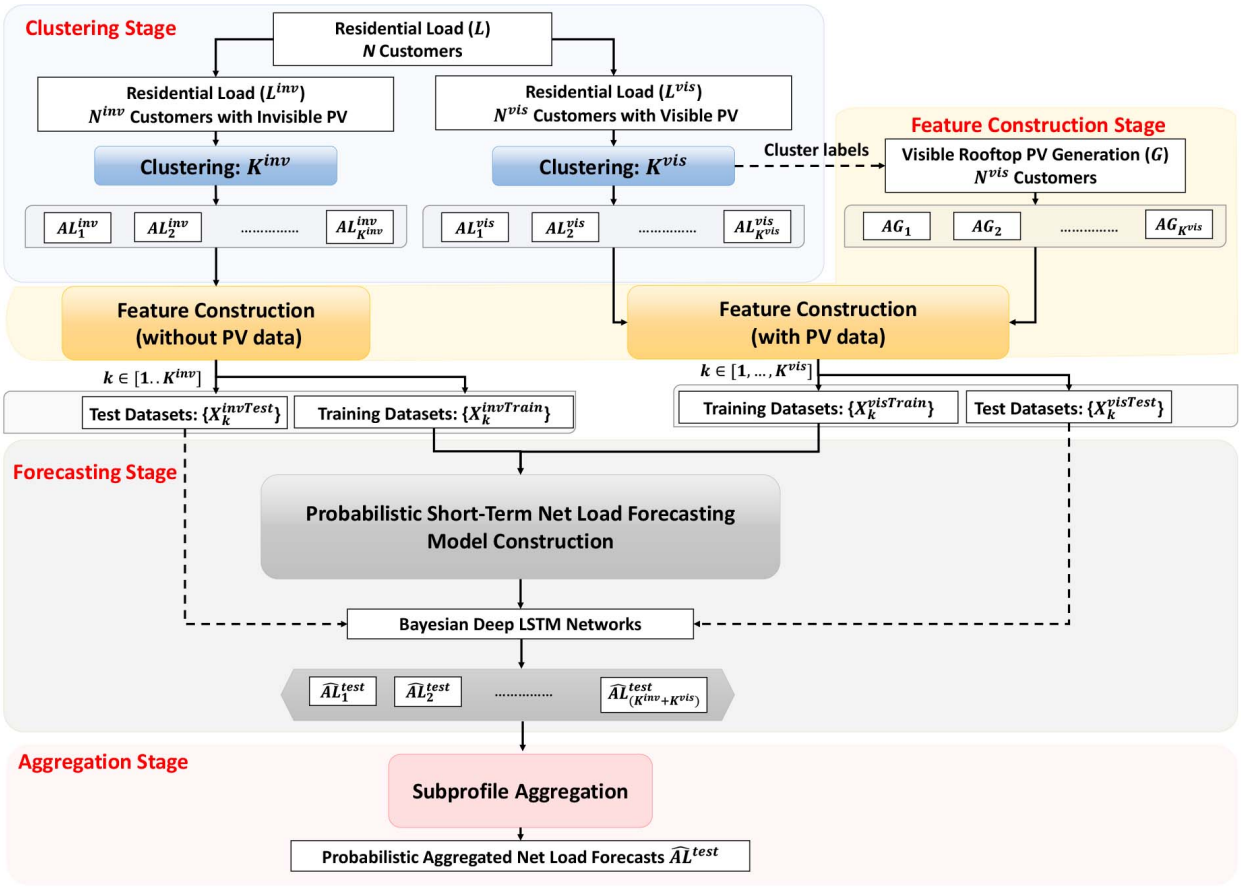


Figure 3.3: The overall structure of the proposed framework.

3.3 The Proposed Net Load Forecasting framework

Based on the above-introduced BDLSTM model, a novel probabilistic short-term net load forecasting scheme is proposed to fully utilize the subprofiles of residential customers and exploit the partially visible PV output data to enhance the forecasting performance. In particular, the proposed framework includes four main stages: i) a *Clustering Stage*; ii) a *Feature Construction Stage*; iii) a *Forecasting Stage*; and iv) an *Aggregation Stage*, as shown in Fig. 3.3.

In order to provide an illustrative and straightforward forecasting procedure, Fig. 3.4 is also employed. It is notable that since the work is about day-ahead forecasting, there is no so called 'real-time operation'. To clarify the ambiguity, the procedure is simplified and summarised as follows: When it comes to 0 o'clock at midnight, the hourly-resolution measurement of the last 24 hour is conducted and collected, which forms the latest section of historical database. If

model updating work is required, historical data including load and temperature etc. are used as input signal to establish the model, which is usually called training process. The testing work can also be implemented to examine the performance of the updated model. If no updating work is needed, then the model can be used directly to obtain new forecasting results. In order to obtain the formal forecasting, the up to date measurements are used as input signal. Since this is only a feed-forward process, the new formal 24-hour forecasting results would only cost seconds.

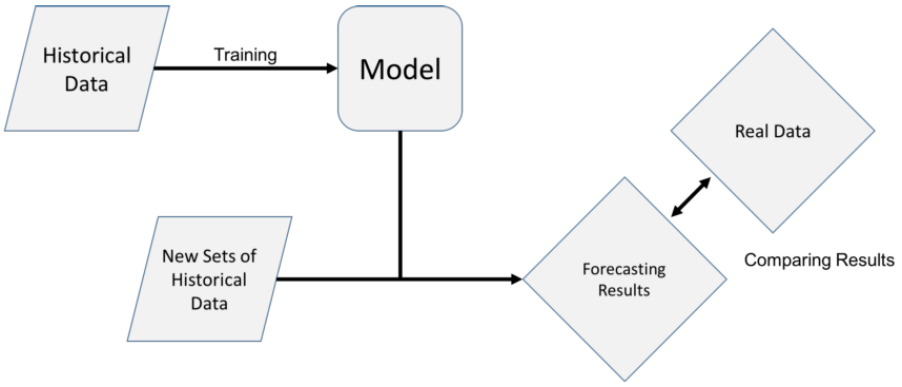


Figure 3.4: Typical diagram of load forecasting

3.3.1 Clustering Stage

In the proposed framework, the *Clustering Stage* aims to group the prosumers into different clusters based on their average daily net load patterns over the training days and to extract representative net load profiles from each cluster. This step is motivated by the fact that fine-grained subprofiles can reveal more information about the aggregated load and further assist in improving the forecasting accuracy [68]. However, it is impractical and inefficient to build a BDLSTM model for each individual customer and then aggregate them. The clustering procedure can also contribute to effectively reducing the computational complexity by balancing the number of models and the forecasting accuracy.

Let $L = [L_1, ..., L_N] \in \mathbb{R}^{T \times N}$ denote the historical load data of N residential customers where T is the total number of observations. The first step is to separate all the customers into two

groups: by invisible PV generation and visible PV generation, represented by $L^{inv} \in \mathbb{R}^{T \times N^{inv}}$ and $L^{vis} \in \mathbb{R}^{T \times N^{vis}}$, respectively. Given that the numbers of clusters for each of these groups are K^{inv} and K^{vis}, respectively, as one of the most widely used and powerful methods, a hierarchical clustering method with Ward's linkage [69],[70] is applied based on the average daily net load patterns of L^{inv} and L^{vis}, defined as $RLP^{inv} \in \mathbb{R}^{N^{inv} \times 48}$ and $RLP^{vis} \in \mathbb{R}^{N^{vis} \times 48}$, respectively, to obtain the cluster label for each individual customer. In particular, hierarchical clustering has the benefits of having a deterministic nature and terminating the agglomeration procedure at any number of clusters as required [71]. A detailed explanation of the hierarchical clustering method with Ward's linkage can be found in references [69, 70, 71]. Subsequently, we aggregate the subprofiles in each cluster for both the *invisible* and *visible* groups to obtain the net load at a higher level:

$$AL_k^{inv(vis)} = \sum_{i \in \Omega_k^{inv(vis)}} L_i, \forall k \in 1 \cdots K^{inv(vis)} \tag{3.17}$$

where AL_k^{inv} and AL_k^{vis} represent the higher-level net load of the k^{th} cluster in the groups L^{inv} and L^{vis}, respectively. It is notable that the weights of each cluster, $P^{inv} = [p_1 \cdots p_{K^{inv}}]$ and $P^{vis} = [p_1 \cdots p_{K^{vis}}]$, are also saved in this stage and then will be used in the *Aggregation Stage*.

3.3.2 Feature Construction Stage

The task of the *Feature Construction Stage* is to identify the most correlated explanatory variables that contribute to forecasting and construct the training and testing sets for the BDLSTM model. For short-term load forecasting, feature selection is a key procedure to obtain reliable prediction strategies by removing ineffective candidate features. Conventionally, feature selection is conducted based on either expert experience or trial-and-error procedures [72]. To automatically select the effective features, the relevance of the input features and the target variable as well as the redundancy among the candidate features are considered as the two critical information-theoretic criteria, which have been investigated in the power systems

literature (e.g., [73, 74, 75]). Beyond that, the concept of interaction (synergy) is proposed in [72] based on the mutual information (MI) and the interaction gain (IG) to measure the interaction among candidate features of a forecast process. The effectiveness of this novel feature selection technique has been well demonstrated based on real load and price data.

Although several advanced feature selection techniques have been proposed for forecasting tasks, in terms of the practical implementation, it is always the accessibility to the data that matters. Depending on the maturity level of the data management (e.g. Does the data has aliened resolution? Does the data storage has enough capability? How to deal with the privacy issue? Does it require the cooperation of multiple departments?), the data quality might also vary a lot. More importantly, the development of deep learning techniques renders it possible to effectively handle raw data without the significant requirements of extensive domain expertise and careful feature design [76]. Therefore, instead of implementing or proposing novel feature selection methods, the investigation in this work focuses on the novel Bayesian deep learning technique, which has the benefit of automatically identifying the representative features based on the raw features while considering uncertainty. In order to give a benchmark comparison and also to demonstrate the significance of the training data features, the critical feature components are discussed next, which is widely agreed and given by [32]. The integration of feature selection methods will be studied in our future work to further improve the forecasting performance.

As illustrated in [32], feature selection should reflect the temperature relations, the seasonal effects and the effects of other interactions.

Temperature

Temperature is reported to have relationships with loads. During the previous decades, the relationship between these two has been intensively studied e.g. piece-wise linear function in [77]; piece-wise quadratic function in [78] and the 3rd ordered polynomials in [79].

In order to investigate the relationship, a scatter diagram of load and temperature is employed in Fig. 3.5 from the 2004 to 2008. Both positive and negative correlations can be identified from

observation. For instance, above approximately 55°F(13°C) a positive correlation between load and temperature can be observed. While temperature below 55°F leads to a negative correlation between two variables. A good example for understanding is that at winter time, people tend to switch on their electric heaters when the weather gets colder, resulting in increasing demand. It is also notable that the characteristics of the figure might vary in different service territories, since the comfortable temperature level might be different for people living in different areas.

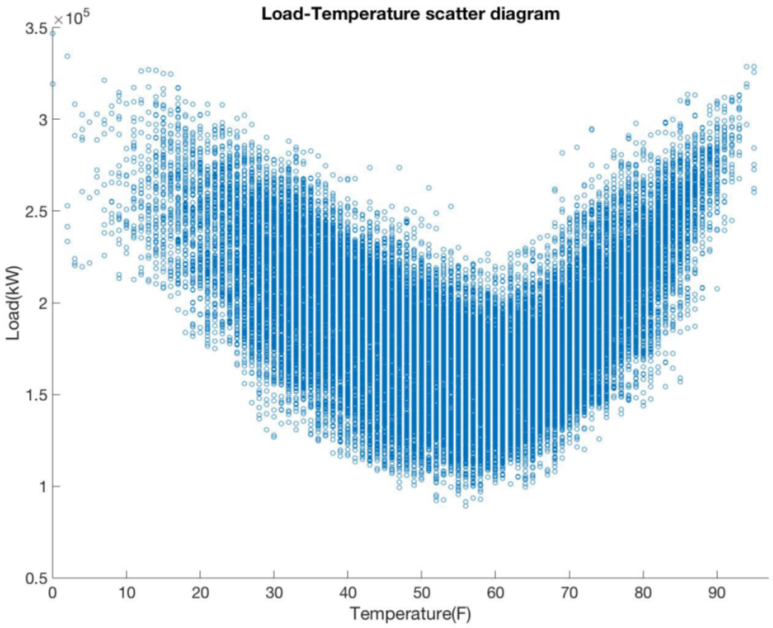


Figure 3.5: Load-Temperature scatter diagram from load forecasting competition database[2].

Calendar Variables

It is obvious that there are three fundamental periodic blocks in the load series: day, week, and year[32]. Depending on the exact methodology to be used or the human activity behavior of the particular territory, there could be various treatments to each block when constructing the database. For example, the 7 days of a week can be denoted by a binary variable with 2 classes (i.e. 1 for weekdays and 0 for weekends), qualitative variables of 3 classes (weekdays, Saturday, Sunday), or 7 classes for each day, etc. Although the definition of weekdays and weekends might be different in some of the places in the world, we focus on the most general regulation. Hence, the calendar variables (hour of the day, day of the week, and month of the year) with 24, 7, and

12 classes, are used to denote the 24 hours of a day, 7 days of a week, and 12 months of a year respectively.

Interaction Effects

Apart from the aforementioned aspects, there are also cross effects from the interaction of the variables. For example, temperature is obviously seasonal-dependent and normally noon has a higher temperature than in the midnight. Therefore, the selected features should also consider the correlation hidden in the interaction of the variables. In this case, the effect between temperature and the calendar variables exists in the form of 3rd ordered polynomial and the calendar variables Hour and Month should be in the model[2]. In addition, there are hidden effects within the calendar variables, for example, the same hour in either weekday or weekend might result in different load consumption due to human activities. For instance, people tend to get up early during weekdays. Typical activities might include taking a shower, boiling a cup of coffee etc. In contrast, there might be less load consumption in the morning of the weekends since people do not have to go to work. Hence, the interaction effect between Hour and Day should also be explored when constructing the training database.

As discussed above, the significant achievements in Deep Learning area gives researcher and electrical engineers the opportunity to develop data mining work upon limited data resources. To this end, we manually select two sets of features for the *visible* and *invisible* groups. Given that the target is to forecast the net load at time t for cluster k in the *invisible* group, the key selected training features for $AL^{inv}_{k,t}$ include the following: 1) the net load historical data at the same time step on the previous day $AL^{inv}_{k,t-24}$; 2) $AL^{inv}_{k,t-24.5}$; 3) $AL^{inv}_{k,t-25}$; 4) the net load historical data at the same time step on the previous two days $AL^{inv}_{k,t-48}$; 5) $AL^{inv}_{k,t-48.5}$; $AL^{inv}_{k,t-49}$; 6) the hour of the day h_t; 7) the day of the week d_t; and 8) the month of the year m_t. In addition to the aforementioned features, we consider historical aggregated rooftop PV generation data $AG_{k,t-24}$, $AG_{k,t-48}$, $AG_{k,t-72}$, and $AG_{k,t-week}$ as additional features to predict the net load $AL^{vis}_{k,t}$ for cluster a k at time t in *visible* group. Afterwards, the training sets of cluster k for both the *invisible* and *visible* groups are constructed as follows:

$$X_{k,t}^{invTrain} = [AL_{k,t-24}^{invTrain}, AL_{k,t-24.5}^{invTrain}, AL_{k,t-25}^{invTrain}, AL_{k,t-48}^{invTrain}, AL_{k,t-48.5}^{invTrain}, AL_{k,t-49}^{invTrain}, h_t, d_t, m_t] \quad (3.18)$$

$$X_{k,t}^{visTrain} = [AL_{k,t-24}^{visTrain}, AL_{k,t-24.5}^{visTrain}, AL_{k,t-25}^{visTrain}, AL_{k,t-48}^{visTrain}, AL_{k,t-48.5}^{visTrain}, AL_{k,t-49}^{visTrain}, h_t, d_t, m_t, AG_{k,t-24}, AG_{k,t-48}, AG_{k,t-72}, AG_{k,t-week}] \quad (3.19)$$

where the test sets $X_k^{invTest}$ and $X_k^{visTest}$ are defined similarly to those of the training sets. In addition, the training labels, the actual net load, for a cluster k are defined as $Y_k^{invTrain}$ and $Y_k^{visTrain}$ for the invisible and visible groups, respectively.

3.3.3 Forecasting Stage

The *Forecasting Stage* is the fundamental core of the entire scheme in which a novel Bayesian deep learning method is proposed. As illustrated in previous chapter, BDLSTM integrates the Bayesian method with a deep LSTM network to capture both aleatoric uncertainty and epistemic uncertainty.

For each k, either in the visible group or the invisible group, the proposed BDLSTM network is trained based on the constructed features $X_k^{invTrain}$ (or $X_k^{visTrain}$) and the target labels $Y_k^{invTrain}$ (or $Y_k^{visTrain}$). When initializing the Bayesian LSTM network, the network parameters including their weights and bias values are constructed by setting up a standard normal distribution as the prior. Additionally, the hyperparameters of the deep LSTM network are optimized in this stage via grid search and cross-validation. Note that we need to construct a total of $K = K^{inv} + K^{vis}$ BDLSTM networks for each cluster. Applying the test datasets $X_k^{invTest}$ and $X_k^{visTest}$ to their

corresponding models, the final outputs of this stage are the predicted aggregated net loads for each cluster $[\hat{AL}_1^{test} \cdots \hat{AL}_K^{test}]$ with a predetermined number of samples n_s for each time step.

3.3.4 Aggregation Stage

In the *Aggregation Stage*, all the individual probabilistic forecasts are aggregated through convolution with the previously saved weights to obtain the final probabilistic net load at the aggregated level, defined as $\hat{AL}^{test}$. Let $f(t)$ and $g(t)$ denote the probability density functions (PDFs) for two independent variables A and B, respectively; a convolution defined as the product of functions f and g over an infinite range, which is the probability distribution of the sum $A + B$, can be expressed as:

$$h(t) = f(t) * g(t) \triangleq \int_{-\infty}^{+\infty} f(\tau)g(t-\tau)d\tau \tag{3.20}$$

If A and B follow their respective Gaussian distributions

$$A \sim \mathcal{N}(\mu_A, \sigma_A^2), B \sim \mathcal{N}(\mu_B, \sigma_B^2), \tag{3.21}$$

then the convolution of two Gaussian distributions is another Gaussian distribution

$$C = A + B \ \sim \mathcal{N}(\mu_A + \mu_B, \sigma_A^2 + \sigma_B^2) \tag{3.22}$$

A detailed explanation and proof of the above equations can be found in reference [80]. In this case, the probabilistic forecast of each cluster is assumed to be independent of each other because the clustering procedure aims to differentiate the customers according to their net load patterns. Additionally, as illustrated in Section III, each individual probabilistic forecast (uncertainty component) obtained via the proposed Bayesian deep learning method follows a Gaussian distribution. Therefore, the distribution of the final aggregated net load can be

directly estimated through the above convolution process, which is expressed as follows:

$$\hat{AL}^{test} \sim \mathcal{N}(\mu_1 + ... + \mu_{(K^{inv}+K^{vis})}, \sigma_1^2 + ... + \sigma^2_{(K^{inv}+K^{vis})}) \tag{3.23}$$

where $\hat{AL}_k^{test} \sim \mathcal{N}(\mu_k, \sigma_k^2)$ represents the sub-aggregated level net load of cluster $k \in \{1, ..., K^{inv}+K^{vis}\}$.

3.4 Case Study

3.4.1 Data Descriptions

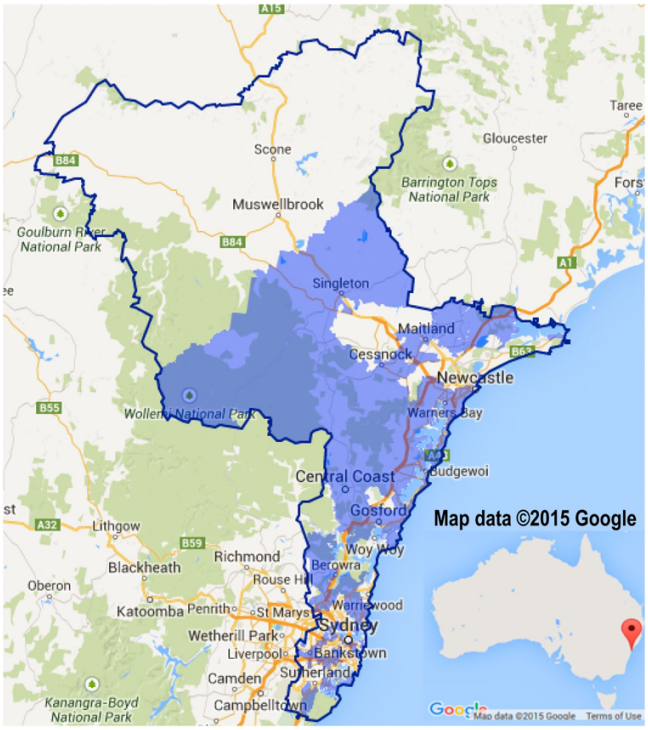


Figure 3.6: Outline of the Ausgrid distribution network[3].

The numerical experiments conducted in this study are based on real smart meter data collected from the Ausgrid distribution network, including load centers in Sydney and regional areas in NSW as shown in Fig. 3.6 [3]. The Ausgrid datasets are composed of separately reported measurements of rooftop PV generation and loads at half-hour time intervals over a three-year

period from 1st July 2010 to 30th June 2013.

A subset of 300 customers, which are spread and covered in the shaded area in Fig. 3.6, are chosen following the rationale as below[3]:

1. Ausgrid identifies the target group of residential customers with the ability to record PV generation directly from the PV panel over the period 1 July 2010–30 June 2013. At this step, approximately 15000 are selected.
2. Within this 15000 customers, Ausgrid removes customers the top and bottom 10% of annual household energy consumption or PV generation.
3. From the remaining customers, 300 are selected randomly by Ausgrid.

In this case, we have the training and test datasets of 21,024 observations and 480 observations, respectively, for both the load and PV generation data for all 300 customers. The target aggregated net load is directly obtained by summing the difference between customer power consumption and the PV outputs for each household. More detailed information about the Ausgrid dataset is given in the literature [3].

3.4.2 Experimental Setup

To demonstrate the superior performance of the proposed approach, a series of state-of-the-art load forecasting methods that have been widely used and firmly demonstrated with reliable performance in the literature are used for comparison. More specifically, M1 (*multiple linear regression*) [32] and M2 (*long short-term memory*) [55] are point forecasting techniques, and the rest are probabilistic models, i.e., M3 (*quantile regression*) [32], M4 (*support vector quantile regression*) [81], M5 (*gradient boosting quantile regression*) [43] and M6 (*quantile random forests*) [43]. The proposed method M7 (*BDLSTM*) is the only method that captures both the epistemic uncertainty and the aleatoric uncertainty in a single model. More specifically, the hyperparameters of the proposed BDLSTM model determined by grid searching and cross

validation are given in Table 3.1. All the tested algorithms were implemented in Python with the main packages of scikit-learn [82], Keras [83] (M1-M6) and Edward [84] (M7) and were run on an Intel Xeon PC with an NVIDIA Titan-V GPU.

Table 3.1: Hyperparameters of the Proposed BDLSTM

Parameter	Value
Layer type	LSTM
Number of hidden layers	2
Number of neurons	10-20
Batch size	720
Number of epochs	150
Number of samples (T_{sample})	100
Dropout rate	0.02
Optimizer	Adam
Learning rate	0.001

3.4.3 Evaluation Metrics

Typical evaluation metrics are used to assess the forecasting performance of the examined methods (M1-M7), including the root mean square error (RMSE), the mean absolute error (MAE), the normalized root mean square deviation (NRMSD), and the mean absolute percentage error (MAPE) for point forecasting, the pinball loss function (Pinball) and the Winkler score (Winkler) for probabilistic forecasting [31], [85]. Given the actual net load AL^{test} and the predicted net load $\hat{AL}^{test}$, the aforementioned metrics are defined and formulated as below.

Metrics for deterministic forecasting

The RMSE measures the square root of the mean of the squares of the errors between the actual and the predicted values, which can be formulated as follows:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T}(AL_t^{test} - \hat{AL}_{p=50,t}^{test})^2}{T}} \tag{3.24}$$

where AL_t^{test} and $\hat{AL}_{p=50,t}^{test}$ are the actual net load and the 50th percentile value of the predicted net load, respectively, at time step t. Then, the NRMSD can be calculated as:

$$NRMSD = \frac{RMSE}{(AL_{max}^{test} - AL_{min}^{test})} \tag{3.25}$$

The MAE and the MAPE are calculated to quantify the absolute difference between the actual and the predicted net load in kW and percent %, respectively, and are expressed as follows:

$$MAE = \frac{1}{T}\sum_{t=1}^{T}\left|AL_t^{test} - \hat{AL}_{p=50,t}^{test}\right| \tag{3.26}$$

$$MAPE = \frac{100\%}{T}\sum_{t=1}^{T}\left|\frac{AL_t^{test} - \hat{AL}_{p=50,t}^{test}}{AL_t^{test}}\right| \tag{3.27}$$

Metrics for probabilistic forecasting

To evaluate the performance of the probabilistic forecasting methods, the calibration, reliability, and sharpness are three main factors that indicate the consistency, the variation, and the tightness of the estimated distribution, respectively [31]. As one of the most comprehensive metrics to measure the above factors, Pinball is used in this work that can be expressed as follows:

$$Pinball = \begin{cases} (AL_t^{test} - \hat{AL}_{q,t}^{test})q & \hat{AL}_{q,t}^{test} < AL_t^{test} \\ (\hat{AL}_{q,t}^{test} - AL_t^{test})(1-q) & \hat{AL}_{q,t}^{test} > AL_t^{test} \end{cases} \tag{3.28}$$

Note that the average of all the Pinball values is calculated to evaluate the overall performance of the probabilistic forecasts for $q = 0.01, 0.02, ..., 0.99$, and a lower value indicates better performance.

Additionally, the Winkler score is another type of comprehensive metric for probabilistic forecasting to simultaneously measure the unconditional coverage and interval width, which can

be expressed as follows:

$$Winkler = \begin{cases} 2(min_t - AL_t^{test})/\alpha + \delta, & AL_t^{test} < min_t \\ 2(AL_t^{test} - max_t)/\alpha + \delta, & AL_t^{test} > max_t \\ \delta, & \text{otherwise} \end{cases} \tag{3.29}$$

where min_t and max_t represent the lower and upper bounds of the probabilistic forecasts at time t (i.e., $\hat{AL}_t^{test}$), respectively, and $\alpha = 0.1$ in this case. A lower score implies better probabilistic estimation results regarding the estimation interval.

3.4.4 Deterministic and Probabilistic Forecasting Results

In this test, we aim to compare the forecasting performance of the proposed BDLSTM method with other popular methods in terms of both the point and probabilistic forecasting results. Note that we use the 50th percentile values for M3-M7 to evaluate their deterministic forecasting results. First, for all the considered methods, we assume that all customers belong to one cluster (i.e., K=1) and that PV data are 100% available for each individual customer. Fig. 3.7 presents

	RMSE	MAE	MAPE	NRMSD
M1(MLR)	43.5783	36.6224	0.2365	0.2278
M3(QR)	41.0675	32.7143	0.2001	0.1861
M4(SVQR)	29.9686	21.7598	0.1346	0.1356
M2(DLSTM)	23.7103	18.8784	0.1194	0.1074
M5(GBQR)	22.6757	17.6198	0.1081	0.1031
M6(QRF)	20.1121	15.4505	0.0937	0.0914
M7(BDLSTM)	17.1698	13.8607	0.0892	0.0775

Figure 3.7: Point forecasting results for different methods.

the point forecasting results of the RMSE and the MAE in kW and the MAPE and the NRMSD in PU. The length of the bar represents the value of the evaluation metric (i.e., a higher value corresponds to a longer bar). The results show that the BDLSTM model M7 dominates with respect to the point forecasting performance, as indicated by the approximately 60.60%, 62.15%,

62.28%, and 65.98% lower RMSE, MAE, MAPE, and NRMSD, respectively, when compared with the benchmark method of multiple linear regression (M1). Moreover, the performance of the BDLSTM model also dominates when compared with the best of the state-of-the-art methods, quantile random forests (M6), with approximately 14.63 %, 10.29%, 4.8%, and 15.21% improvements in the four evaluation metrics.

	Pinball	Winkler
M3(QR)	13.7448	189.6120
M4(SVQR)	8.1972	133.6392
M5(GBQR)	6.8724	99.5256
M6(QRF)	6.2100	94.6404
M7(BDLSTM)	4.8852	74.7684

Figure 3.8: Probabilistic forecasting results for different methods.

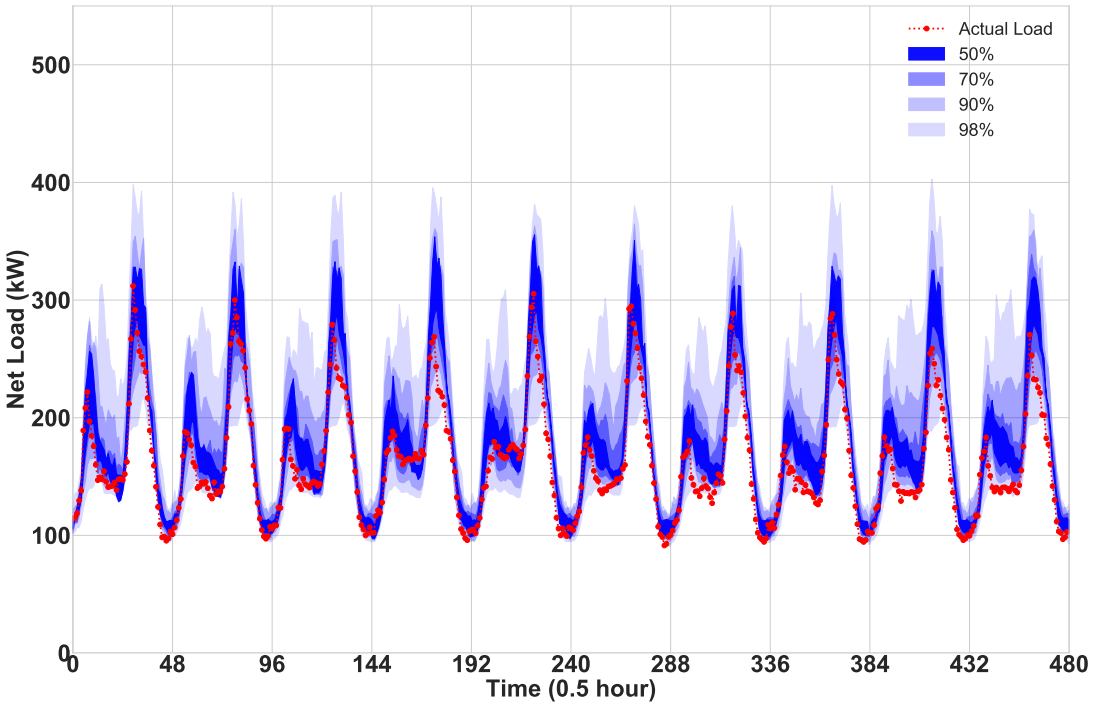


Figure 3.9: Probabilistic net load forecasting results: M5 (GBQR)

To illustrate the effectiveness of the proposed BDLSTM method and its capability to capture uncertainty, the overall probabilistic evaluation metric values of different probabilistic methods

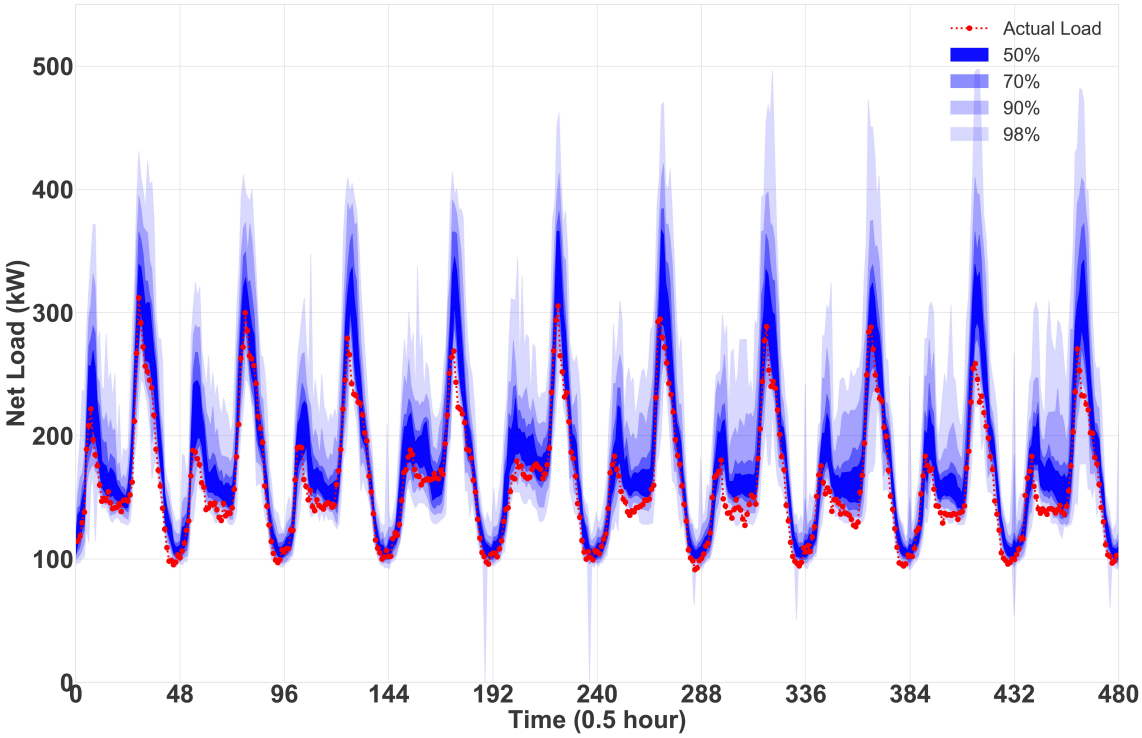


Figure 3.10: Probabilistic net load forecasting results: M6 (QRF)

(i.e., M3-M7) are listed in Fig. 3.8. The data show that the forecasting results obtained via the proposed Bayesian deep LSTM network with VI has the highest accuracy followed by the quantile random forests method (M6). The fact that M7 presents the best predictive capability indicates the significance of capturing both epistemic uncertainty and aleatoric uncertainty. As shown in Fig. 3.8, other methods, such as M3 and M4, perform poorly in this respect because they focus only on the uncertainty in the net load data (i.e., the aleatoric uncertainty). Another important finding is that the performance order across the different probabilistic forecasting methods is consistent with the results of point forecasting, shown in Fig. 3.7. For example, M7 (BDLSTM) outperforms the other tested methods, showing approximately 64.46% and 60.57% performance enhancements for the pinball loss and the Winkler score, respectively, compared with M3. Furthermore, M6 exhibits better performance than the other conventional approaches.

Additionally, Figs. 3.9 3.10 and 3.11 show the forecasting results of the 10 test days obtained via the proposed BDLSTM model, the second-best model M6 (QRF), and M5 (GBQR). Note that the actual net load during the tested periods is represented by the red curve with dots.

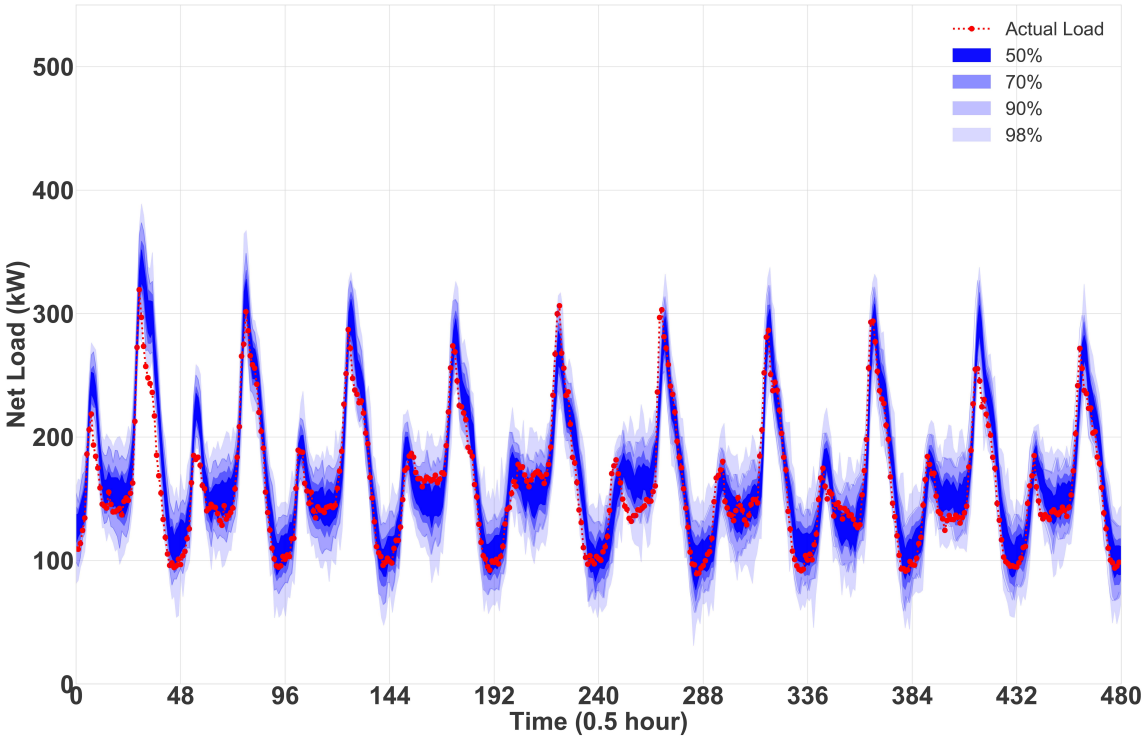
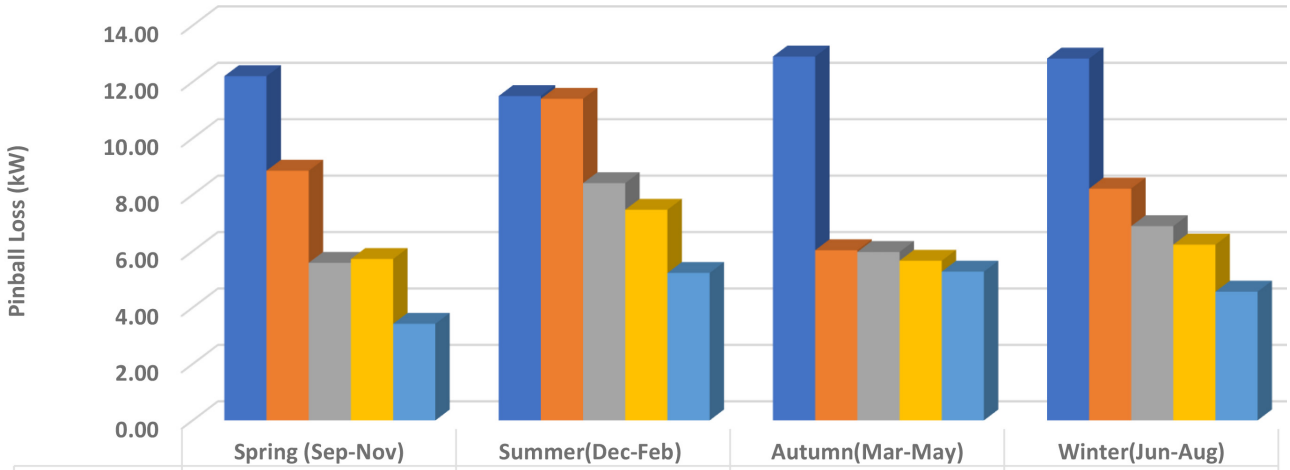


Figure 3.11: Probabilistic net load forecasting results: M7 (BDLSTM)

The 98%, 90%, 70%, and 50% confidence intervals are indicated by an increasing color depth of the blues. In general, the probabilistic forecasting performance is evaluated in terms of three primary aspects: reliability, sharpness, and resolution [31], which have been quantified by the comprehensive evaluation criteria: the pinball loss and the Winkler score. Visually inspecting the results of M5 (GBQR), M6 (QRF) and M7 (BDLSTM), the probabilistic forecasts generated using the constructed BDLSTM model present the benefits of a tighter prediction coverage interval, a lower prediction interval that varies over time, and higher unconditional coverage, corresponding to sharpness, resolution, and reliability, respectively [31]. It is constructive to highlight that the net loads during the peak hours of each day, which are crucial factors for system operation, can be well predicted with reasonable magnitudes using the proposed Bayesian deep learning method. On the other hand, it can be seen that M5 and M6 overestimates the peak demand with a misleading trend across the 10 test days.

Additionally, we expand the test datasets from 10 days to four seasons to investigate the probabilistic forecasting performance across the different seasons. Fig. 3.12 presents the average



	Spring (Sep-Nov)	Summer(Dec-Feb)	Autumn(Mar-May)	Winter(Jun-Aug)
M3(QR)	12.2026	11.4978	12.8997	12.8279
M4(SVQR)	8.8518	11.4015	6.0322	8.2177
M5(GBQR)	5.5870	8.4105	5.9700	6.8871
M6(QRF)	5.7217	7.4669	5.6613	6.2317
M7(BDLSTM)	3.4249	5.2272	5.2742	4.5634

Figure 3.12: The average pinball loss across different seasons.

pinball loss values and the bar plots for all the probabilistic forecasting approaches (M3-M7). As shown, although the amount of relative improvement varies across different seasons, the proposed BDBL method (M7) consistently outperforms the other benchmark approaches, especially in spring, which exhibits a 40.14% lower average pinball loss value than that of M6 (QRF). Furthermore, compared with QRF, 29.99%, 6.83% and 26.77% improvements are obtained by using Bayesian deep learning to conduct the probabilistic net load forecasting during the periods of summer, autumn and winter, respectively.

Table 3.2: Deterministic and Probabilistic Forecasting Results for BDLSTM and QLSTM

	Pinball	**Winkler**	**RMSE**	**MAE**	**MAPE**	**NRMSD**
QLSTM	6.1073	91.8621	21.9014	17.7268	0.1155	0.0993
BDLSTM	4.8852	74.7684	17.1698	13.8607	0.0892	0.0775
Relative Improvements (%)	20.01%	18.61%	21.60%	21.81%	22.77%	21.95%

Regarding the computational cost, the CPU times of the training process for all the examined methods are presented in Table 3.3. The proposed BDLSTM method takes longer to train than most of the other benchmark approaches. However, it is notable that model training is an offline procedure. Given the input features, using the constructed model to conduct day-ahead forecasting only takes a few seconds in practice. Therefore, the main target in this case is to

Table 3.3: Computational Time For Model Training

	CPU Time (s)
M1(MLR)	1.57
M2(DLSTM)	885.25
M3(QR)	53.60
M4(SVQR)	12420,67
M5(GBQR)	198.74
M6(QRF)	441.59
M7(BDLSTM)	2495.13

obtain an accurate forecasting result.

3.4.5 Bayesian LSTM vs Pinball Loss Guided LSTM

Recently, a series of novel deterministic deep learning-based probabilistic models have been proposed in the literature (e.g., [44, 86, 53]) to exploit deep learning to achieve state-of-the-art performance in probabilistic load forecasting. In particular, an improved wavelet neural network, a multilayer perceptron (MLP) and a deep LSTM network are considered the main networks in [86], [53] and [44], respectively. To make the comparisons based on the same type of network considered in this work (i.e., LSTM), we implement a pinball loss guided LSTM (QLSTM) algorithm proposed in [44] in this case. More specifically, instead of using the mean square error (MSE), QLSTM employs the pinball loss as the loss function to guide the training of the parameters and thus extends traditional LSTM-based point forecasting to probabilistic forecasting in the form of quantiles. The deterministic and probabilistic forecasting results of QLSTM and BDLSTM are given in Table 3.2. As can be seen, with the same network architecture, Bayesian deep learning exhibits a superior performance to the deterministic deep learning-based probabilistic model with approximately 20% relative improvements regarding the evaluation metric values and thus further highlighting the importance and benefit of capturing the model uncertainty.

3.4.6 Different Numbers of Clusters

After demonstrating the prominent probabilistic forecasting capability of the Bayesian deep LSTM network, this part aims to verity the effectiveness of the *Clustering Stage* in the proposed framework. In this case, we assume that all the PV data are still visible and that the number of clusters is set to $K = [1, 2, 3, 4, 5, 6]$. The point and probabilistic evaluation metrics across different Ks are shown in Fig. 3.13.

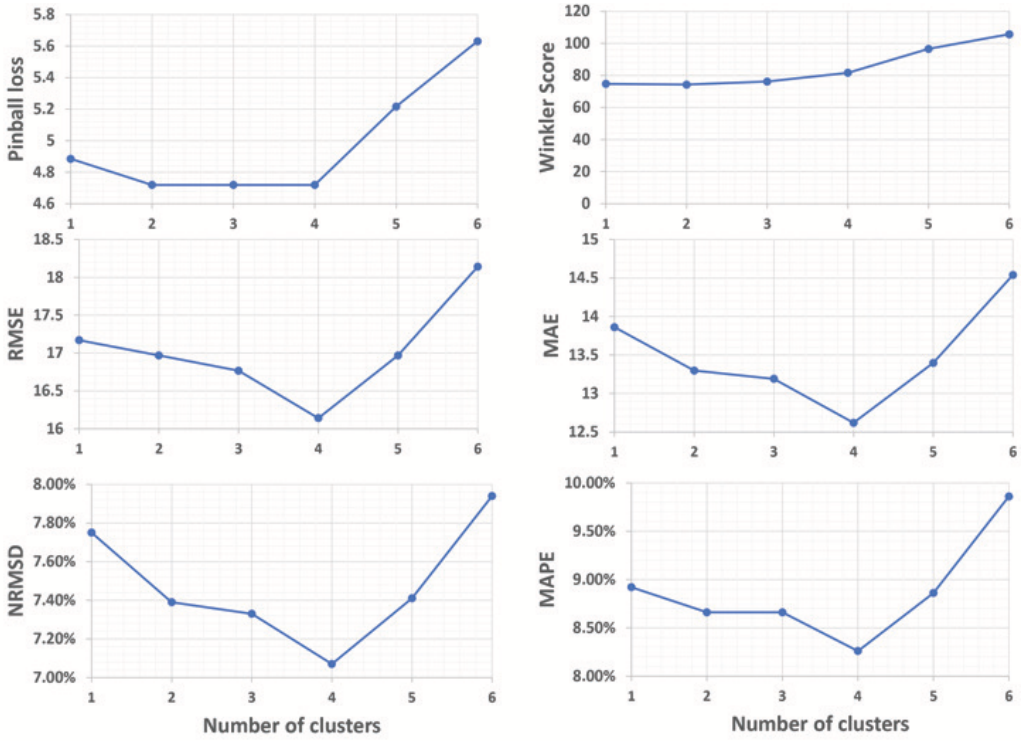


Figure 3.13: Net load forecasting performance across different K.

Most of the criteria decrease from $K = 1$ and achieve the best performance at $K = 4$ with further improvements of 3.39%, 5.99%, 8.96%, 8.77%, and 7.40% for the pinball loss, RMSE, MAE, NRMSD, and MAPE, respectively, demonstrating the importance and effectiveness of performing clustering based on the subprofiles and then aggregating to the higher-level net load. In addition, these separate clusters are all trained by the same network structure, i.e., two layers of 10 and 20 neurons in each layer, which is the best network structure for the one-cluster, 100% PV-visibility case. Therefore, further adjustment of the hyperparameters for each individual cluster may improve the forecasting performance at the aggregated level.

Furthermore, to investigate how categorizing the costumers by the invisibility of their solar power affects the prediction model, an additional case study is carried out to evaluate the probabilistic forecasting performance across different combinations of K^{vis} and K^{inv} in the context of *visibility*= 50%. Table 3.4 presents the calculated average pinball losses for the proposed BDLSTM method across different numbers of $K = K^{vis} + K^{inv}$, where $K^{inv} = 1, 2, 3$ and $K^{vis} = 1, 2, 3$. The results show that the optimal combination is $K^{inv} = 1, K^{vis} = 3$, which results in an approximately 31.14% improvement regarding the average pinball loss when compared with the no-clustering case (i.e., $K^{inv} = 1, K^{vis} = 1$). In addition, increasing the number of clusters either for the visible group or for the invisible group both lead to lower pinball losses than that of the no-clustering case, which demonstrates the effectiveness of the *Clustering Stage* in the proposed framework. Note that if the number of clusters increases to a relatively large value (e.g., $K^{inv} = 4, K^{vis} = 4$), the calculated pinball loss may become larger than that of the no-clustering case, and thus, it is imperative to select an appropriate range for K^{inv}/K^{vis} to determine the optimal combinations.

Table 3.4: The average pinball loss across different numbers of clusters (M7-BDLSTM, Visibility= 50%)

	$K^{inv} = 1$	$K^{inv} = 2$	$K^{inv} = 3$
$K^{vis} = 1$	6.2100	4.5021	4.9202
$K^{vis} = 2$	5.1478	5.4826	4.8550
$K^{vis} = 3$	4.2759	4.6060	5.8326

3.4.7 Different Levels of PV Visibility

In this part, the case study lies in investigating how and to what extent the visibility of distributed PV generation can contribute to a more accurate net load forecasting at the aggregated level. This experiment is carried out based on the assumption that $K = 1$ across various levels of PV visibility, defined by $vis = [0, 0.2, 0.4, 0.5, 0.6, 0.8, 1]$. In this case, $vis = 0$ and $vis = 1$ represent the contexts of invisible and visible PV, respectively, whereas other values indicate that PV data are partially visible. For example, $vis = 0.5$ means that 50% of the 300 households have separate meters for rooftop PV generation, and the rest of the PV outputs are not measured.

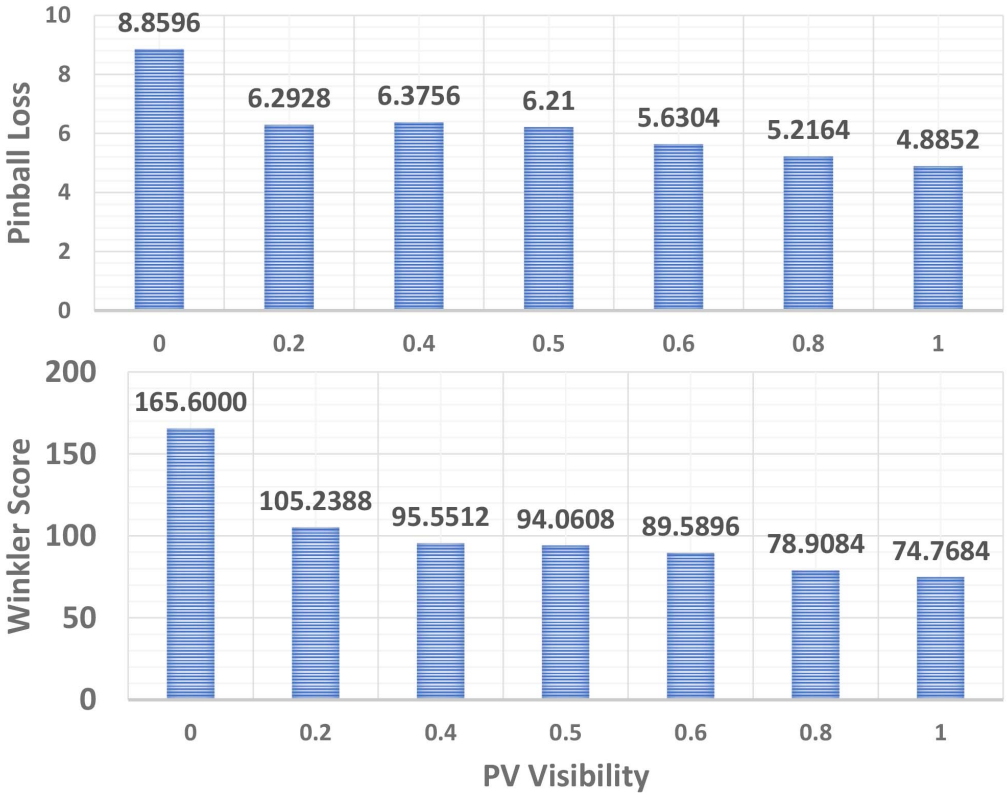


Figure 3.14: Net load forecasting performance across various PV visibilities.

Fig. 3.14 contains the bar plots of the pinball loss and the Winkler score across different visibility levels. The primary conclusions stemming from the results are depicted as follows: i) exploiting the available PV output data from visible PV generation can enhance the forecasting performance of the net load at the aggregated level, and ii) in terms of the costs of installing meters for measuring the PV outputs separately, a trade-off between the forecasting accuracy and the PV visibility can be made based on the operator's requirements. For example, if the system operator can accept an approximately 13% lower pinball loss value ($vis = 0.6$ vs $vis = 1$), only 60% of the households need to install separate meters for rooftop PV generation, thus leading to a significant reduction in terms of the costs of the devices and their installation.

Finally, to demonstrate the superior performance of the proposed method under different levels of PV visibility, the pinball losses for all the tested probabilistic forecasting methods are calculated and presented in Fig 3.15. It can be seen that with the increasing visibility of the PV output, the probabilistic net load forecasting results of all the tested methods improve and are indicated by

	Visibility=0%	Visibility=50%	Visibility=100%
M3(QR)	21.3624	20.9860	13.7448
M4(SVQR)	15.6668	12.3741	8.1972
M5(GBQR)	10.0577	9.5713	6.8724
M6(QRF)	9.0216	8.9629	6.2100
M7(BDLSTM)	8.8596	6.2100	4.8852

Figure 3.15: The average pinball loss under different levels of PV visibility.

the reduced pinball loss values. To further enhance the performance of the proposed framework, our future work will increase the PV 'visibility' by estimating the invisible PV generation using some novel invisible solar power generation estimation approaches (e.g., [87]). In addition, the results demonstrate the superiority and effectiveness of the proposed BDLSTM method across different levels of visibility.

3.5 Conclusion

This work proposes a novel probabilistic net load forecasting framework using a Bayesian deep LSTM neural network to capture epistemic uncertainty and aleatoric uncertainty simultaneously. In the proposed scheme, the *Clustering Stage* aims to enhance the forecasting performance by building a deep learning model for each individual cluster and aggregating the probabilistic forecasts of each cluster at the end to obtain the final predicted net load at the aggregated level. The effectiveness and importance of considering visible or partially visible PV output data as an input feature is investigated across different PV visibility levels. The overall performance of the proposed method is analyzed and compared with a series of state-of-the-art probabilistic forecasting models. The evaluation results demonstrate the superior performance of the proposed Bayesian deep learning-based method and highlight the improvements contributed by the *Clustering Stage* and the PV visibility.

Chapter 4

A Confidence-Aware Machine Learning Framework for Dynamic Security Assessment

4.1 Introduction

The world is expecting a securer and cleaner power system in the future. To achieve this, continued attention is drawn on the integration of RES. However, due to its intermittent nature, massive uncertainties and corresponding corrective devices are brought to the power system[88]. Suffering from the difficulty of accurately predict the sources (location and levels of power injections), the operation mode of power flows thus is strongly diversified[89]. Consequently, the category of traditional disruptive dynamic phenomena might need to be replaced and expanded with more unforeseen critical conditions, which implies frequent system topology changes. Based on the connectivity status among power system components such as generators, transformers, lines, and loads[90], topology changes can be classified as: 1) scheduled system topology changes (e.g. line maintenance); 2) structure changes caused by the on/off status of circuit breakers. Traditional DSA doctrine hence is challenged by these potential changes of system operation

mode, and it is urgently important for TSOs to develop a robust and accurate DSA tool to deal with the challenges of system topology change.

In general, system security issues could be categorized into either static (e.g. line overloading or voltage limits exceeding) or dynamic (e.g. rotor angle stability) security problems. The former is relatively simple since power system parameters in the post-disturbance steady state directly indicate whether system limits are violated or not. The latter, on the other hand, requires more advanced modelling techniques, which can be either data-driven or analytical. In the literature, there are different approaches to predict the transient stability status of a power system: 1) time-domain simulations (TDS), 2) transient-energy-function (TEF) methods, 3) curve-fitting techniques, and 4) machine learning-based methods[91][92]. In particular, TDS provides the most straightforward analytical approach[93]. However, the simulation task is usually highly computing-intensive since detailed information of network configuration during and after a fault is required[91]. In order to solve this issue, researchers have investigated the feasibility of carrying out part of the computation offline. In terms of the TEF methods, Lyapunov function, which includes the kinetic energy and potential energy of a system, is employed to establish a critical energy level first. Then the system assessment is achieved by comparing the target value with this threshold value under a given disturbance[94]. However, one practical issue is that the determination of the level of kinetic and potential energy is almost intractable, especially under certain disturbances[91]. To tackle this issue, data-driven approaches are developed since they do not require the physical information of the network. Examples such as [95] predict the post-fault rotor angle behaviour using grey Verhulst model. Curve-fitting method (e.g., [96][97]) is another approach which aims at avoiding using the network configuration information. However, the prediction performance is poor as it suffers from the start-up time of prediction and the sampling period[91].

Recently, with the superior development of phasor measurement units (PMUs), the post-disturbance dynamic response of a power system can be directly measured. This cutting-edge technique encourages researchers to construct more reliable models through machine learning methods instead of using conventional rules[8]. In particular, a machine learning model can be

established and trained offline by using the TDS results (training labels) collected in advance. The established operating conditions (OCs) provide a region that the system can operate within and likely to occur in the near foreseeable future. Hence the system operator can conduct some analysis before real-time decision-making. As it is released from the constraints of real-time process, classifier thus can be trained on a significantly larger database in order to obtain better performance. In the literature, most of the works are focusing on Decision Trees (DTs) (e.g.[98][99][100][7] [101][102]) since it shows advantage on computational speed. Works such as [103] also uses Decision trees to provide interpretability. Other techniques such as Support Vector Machines (SVMs)[104], long short-term memory (LSTM) networks[105], and ensemble approaches [106][107] have also been widely verified. In addition, works such as [108] and [109] employ hybrid ensemble models, including extreme learning machine (ELM) and random vector functional link networks (RVFL). The former uses the idea of transfer learning in order to implement one model on other faults, so that time cost of training a large number of models can be alleviated. The latter uses generative adversarial network (GAN) in order to complete the missing data so that the original feature characters can be reconstructed. With such, the DSA accuracy can be maintained.

Although machine learning models have shown promising performance in terms of the security assessment task, most of the existing methods are facing the fundamental limitation regarding their capability of confidence awareness. In this work, confidence awareness refers to the ability that a machine learning model obtains model uncertainty through an epistemic learning process. The model with such an ability thus could quantify how confident the model is about its outputs upon the given data set. From the perspective of TSOs, the significance of confidence awareness thus lies in assigning a high level of uncertainty to the erroneous predictions so that the decision-making process could be assisted. Existing works such as [110] and [111] propose probabilistic modelling since it could generate probabilistic intervals. However, these generated intervals cannot be treated as the confidence indicator since the ability of confidence awareness comes from the internal model uncertainty (epistemic uncertainty). This is the property that reflects how much model parameter would change with more knowledge obtained by the model.

Therefore, the value of those works is restricted only to proposing more advanced models so that the performance is enhanced.

With the new challenge in confidence representation, in recent years, Bayesian Deep Learning (BDL) has received widespread attention in a range of research fields such as renewable energy forecasting[112], energy price forecasting [113], semantic segmentation[114], and health-care [115] etc. Through the angle of probability theory, BDL reveals the advantages in terms of uncertainty representation, generalization, and prediction reliability, which makes the neural network more explainable[27]. Currently, there are two different directions to realize Bayesian Deep Learning. The work in [116] uses direct inference with Kullback-Leibler (KL) divergence as minimization target. On the other hand, the authors in [6] employ dropout technique as Bayesian approximation, where the aleatoric part of the uncertainties are used as part of the minimization objective under an unsupervised process.

In this part of work, a Bayesian deep auto-encoder based methodological framework is proposed, which is able to solve multi-contingency issue and provide confidence information. Key contributions of this section can be summarized as follows:

(1) A confidence-aware machine learning framework for DSA of the large-scale electrical system is proposed. To the best of the authors' knowledge, this is the first attemption that achieves confidence awareness by exploiting Bayesian deep learning in the DSA problem.

(2) The concept of conditional training is introduced. The proposed framework thus enhances the performance when facing multi-contingency issue within a single model.

(3) A confidence-oriented model updating strategy is proposed. The proposed strategy only requires small sample data to update the model.

(4) A series of comprehensive case studies are conducted. The superior and robustness performance of the proposed method is demonstrated and compared with other state-of-the-art approaches, which is based on different system topology.

4.2 The Proposed DSA with Model Updating Strategy Framework

4.2.1 The Concept of Security:

The definition of power system security is 'the ability of the bulk power electric system to withstand sudden disturbances such as electric circuits or unanticipated loss of system components', which is widely accepted and given from the North American Reliability Council (NERC) [117].

There are fundamental difference between the concept of security and stability. For instance, stability is the ability that system could recover from a disturbance regardless of post-fault operating limits. In other words, after a disturbance, power system returns to its initial position or enters into a new stable position. Hence, stability reflects the actual response of the system and it is a feature of the system itself.

On the other hand, security relates to a pre-defined list of contingencies and operating limits of the system assets. For instance, the occurrence of any contingency in the system might cause violation of operating limits such as overloading of lines, bus-bar voltage out of nominal range, generator rotor angels, or system frequency etc. Hence, the list of contingencies should include those might lead to high impact disruptions in the system operation and are also highly likely to occur in the foreseeable future. The operating limits of the assets are typically defined by their nominal operating range. Together, they are associated with different phenomena in the power system and might eventually affect the power delivery. The specification of security depends on the definition and criteria that are used to test the system, which provides the fundamental obligation for the system operator. Currently, the system operator of most countries follows the 'N-1' security principle. 'N-1' security means that the system operator must ensure that the operating condition remains stable after the failure of every single asset of the system. Typically, the analysis focuses on failures of major transmission lines and the largest generators in the system.

4.2.2 The Concept of Security Assessment:

To guarantee the system security, it is of significant importance to evaluate the system via all likely contingencies. The security assessment thus is the process of providing security information to the operator so that the system operator is able to be aware of the current and future level of security. For the cases which violate the system's security rules, appropriate corrective/preventive actions should be implemented to ensure the system's security. The planning and operation of the power system requires information on whether the system will survive the direct transient disturbance and results in an acceptable steady-state condition with all assets remain in their nominal operating limits[117]. The system's security assessment thus is classified into two categories, static and transient/dynamic security.

Static Security Assessment (SSA):

The target of static security assessment is to evaluate the system voltage and the equipment's thermal limits violations after any contingency to ensure that they remain in the acceptable range upon a pre-fault operating condition. Based on the N-1 principle, the power system operators must ensure that the operation of the system is maintained for every single failure of equipment in the system. Hence, operators must assess the contingencies of single faults of the equipment. For instance, the SSA includes computations for whether the power balance is met in the post-fault operating condition, which could be obtained from a steady-state simulator. Typical functions of the SSA include performing load flow calculations and sensitivity analysis etc. In addition, in order to obtain the limits and constraints for the pre-fault operating condition, optimal power flow (OPF) based methods are widely used, which are designed with respect to the network constraints including line capacities, the balance of the network nodes, and other physical feasibility such as Kirchhoff's laws. Optimisation techniques are often used to solve an OPF problem. On the other hand, when extra constrains of static security are involved in the OPF problem, the common OPF problem can be expanded to a Security Constraint OPF (SCOPF). Hence, in the SCOPF problem, a set of contingencies are considered and

the optimisation is conducted so that the post-fault steady-state security remains for these contingencies.

Dynamic Security Assessment (DSA):

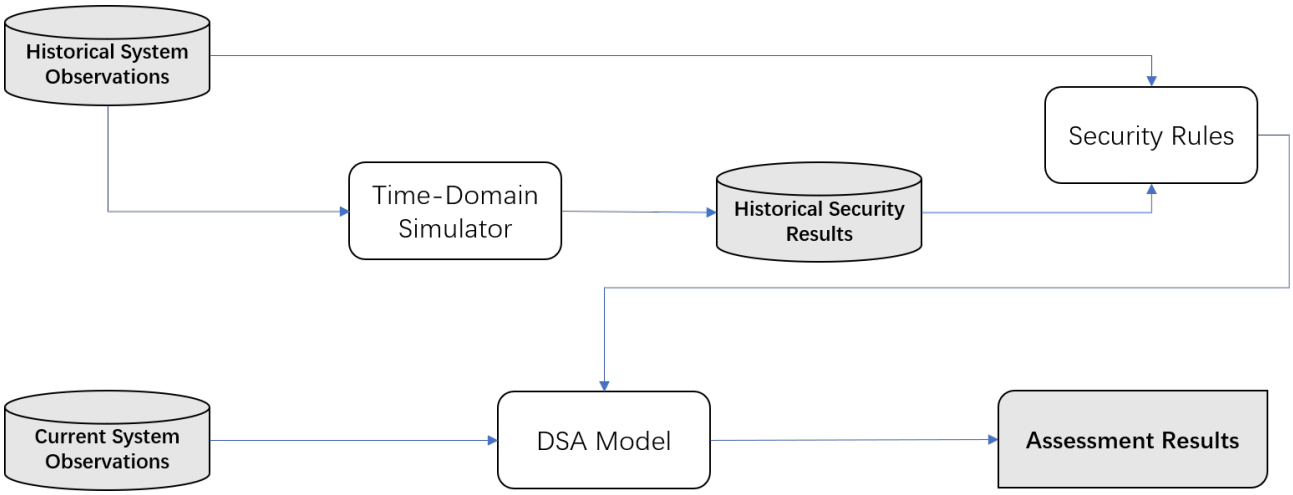


Figure 4.1: The illustrative input-output diagram of DSA.

The target of DSA is to assess the dynamic/transient response for a set of pre-defined contingencies on a pre-fault operating condition. A typical example of DSA workflow is demonstrated in Fig. 4.1, which shows the input-output signals of both offline training and online assessing. For instance, the first step in the offline workflow is to collect historical observations of system operating conditions. These pre-fault operating conditions could be described by bus-bar voltage, the phase-angles, power generation or load at each bus-bar and power flows of the each line. From these pre-fault operating conditions, time-domain simulations are conducted so that binary labels are generated to indicate secure or insecure (1 for a secure response to a contingency and 0 for an insecure response to a contingency). This simulation requires significant computational resources and more importantly, the computational cost grows with the size of the power system since the number of studied contingencies and the corresponding operating conditions also increases significantly. Hence, instead of the real-time online assessment of dynamic security that is currently not practical, system operators tend to set up large safety margins on the operating equipment in order to ensure that the dynamic response is secure[118]. However, this

redundancy setting often results in lower utilise level of the system components, which further implies less economic profits. Hence, in order to push the system operation approaching to its limits and also enhancing economic efficiency, it is reasonable to consider more contingency scenarios. For example, apart from single faults that could happen in the system, sometimes there could also be multiple assets failure at the same time or in a cascade fashion. Despite less possibility, once happened it would lead to a much more severe result. In general, higher security level could be obtained if more contingencies are considered in the assessment. But in order to stick to the 'N-1' principle, the author would like to emphasise that in this work, only single element failure is considered and explored.

To tackle the challenges and address the issues proposed in previous sections, we propose a novel Conditional Bayesian Deep Auto-Encoder (CBDAC) based DSA framework, as shown in Fig.4.2. The framework includes the following stages: off-line training, model updating and online assessment. In particular, the input database of training stage is constructed with the input features and their corresponding labels, which are represented by the pre-fault OCs and the post-fault TDS results, respectively. Furthermore, a validation set is established under different system topologies to identify when to update the model indicated by the model uncertainty. After that, the online part can be conducted by feeding in the real-time measurements. Specifically, the detailed step by step explanation of the framework is given as follows:

4.2.3 Database Construction Stage:

The first step is to construct the database, which includes pre-fault OCs data and the corresponding post-fault labels indicating whether safe or not. The pre-fault OCs include active and reactive power (either generation $G^{active}_{original} \in \mathbb{R}^{n\times g}, G^{reactive}_{original} \in \mathbb{R}^{n\times g}$ or load $L^{active}_{original} \in \mathbb{R}^{n\times l}, L^{reactive}_{original} \in \mathbb{R}^{n\times l}$), power flows $F^{active}_{original} \in \mathbb{R}^{n\times f}, F^{reactive}_{original} \in \mathbb{R}^{n\times f}$, voltages $V_{original} \in \mathbb{R}^{n\times v}$, and phase angles $\Theta_{original} \in \mathbb{R}^{n\times\theta}$ of each bus. These simulations together construct the m dimension original training features $X_{original} \in \mathbb{R}^{n\times m}$, where n represents the size of entire data set from one topology and $m = 2 \times (g + l + f) + v + \theta$. The corresponding post-fault labels,

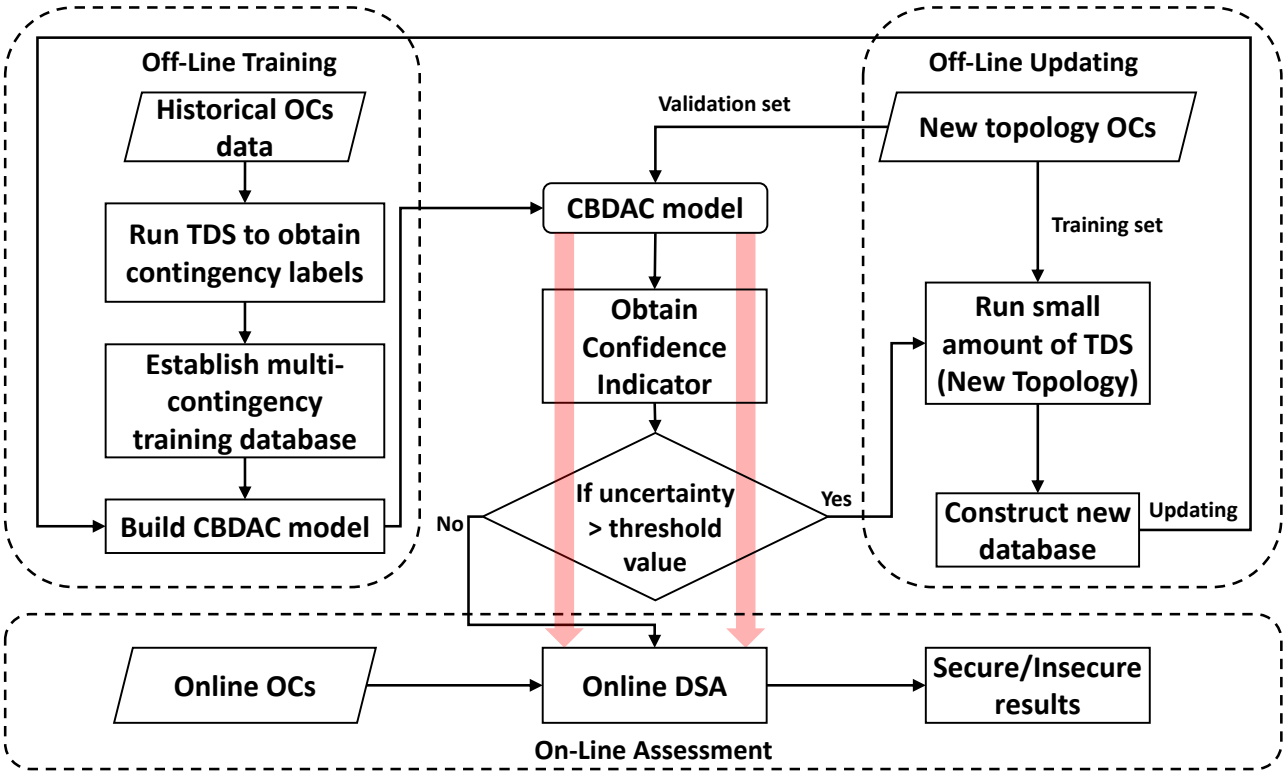


Figure 4.2: The proposed CBDAC based DSA framework.

denoted as $Y_{original} \in \mathbb{R}^n$, where for each element y_i

$$y_i = \begin{cases} 1, & safe \\ 0, & unsafe \end{cases} \tag{4.1}$$

are from off-line computed TDS. It is notable here that we use $T_0, T_1, ..., T_K$ to represent the data from various system topology. In addition, within one system topology, various contingencies data are generated in order to expand the OCs domain. Given the total contingency number C, data sets $X_{original}, Y_{original}$ are stacked up, normalised and shuffled properly. Eventually, the dimension of features becomes $X \in \mathbb{R}^{(C\times n)\times m}$ and the labels become $Y \in \mathbb{R}^{(C\times n)\times 2}$ since the labels are transferred into one-hot code. The data is then separated into training and testing parts. We use $X^{T_0}_{train}$, $Y^{T_0}_{train}$, $X^{T_0}_{test}$, $Y^{T_0}_{test}$ to denote the data from T_0.

4.2.4 Training and Evaluating Stage:

The constructed database $X_{train}^{T_0}$, $Y_{train}^{T_0}$, $X_{test}^{T_0}$, $Y_{test}^{T_0}$ is then used to train and evaluate the model. The model is based on auto-encoder with modified Bayesian approximation, instead of logistic regression, from the hidden layer. In order to enhance the network performance under multiple contingencies, we set up a conditional mask at each layer. A detailed introduction is given in the following subsections.

Modified Deep Auto-Encoder

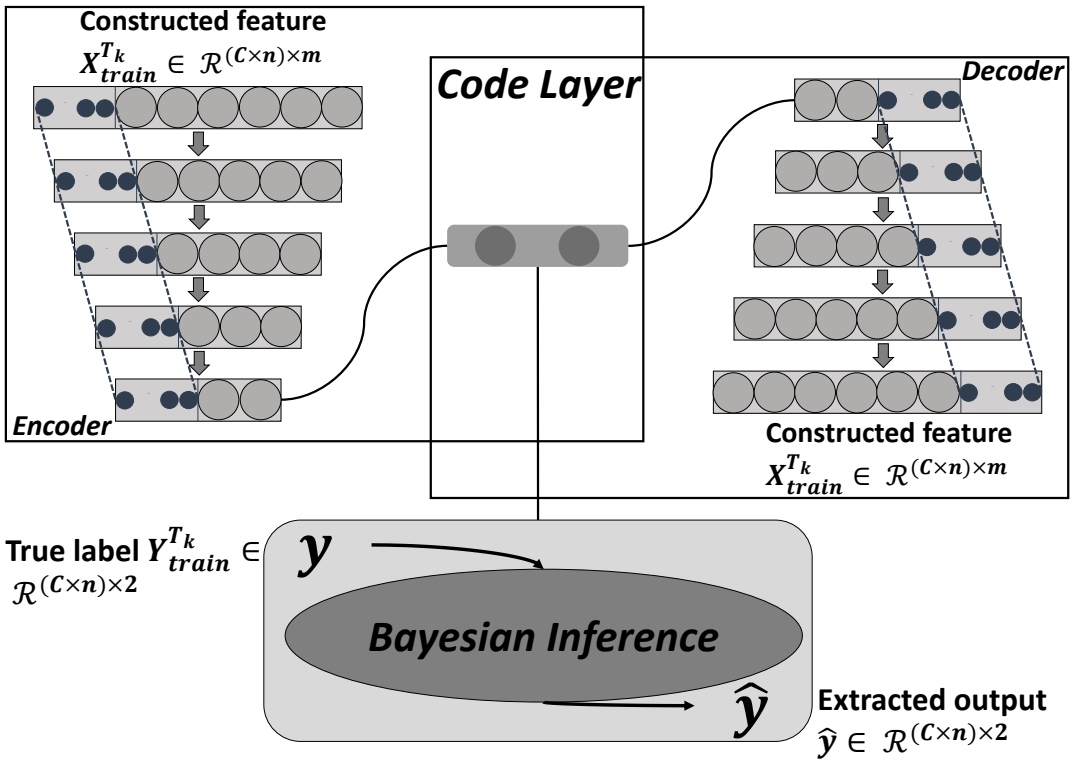


Figure 4.3: The structure of Conditional Bayesian Deep Auto-Encoder.

In DSA problem, system OCs are evaluated as secure or insecure, which makes DSA essentially a classification question. To this end, researchers have investigated the feasibility of various machine learning approaches as the classifier. Among those state-of-the-art methodologies, auto-encoder is one of the most famous examples. Traditional auto-encoder is unsupervised, including an input layer, a hidden layer, and an output layer. Auto-encoder is usually used for

feature extraction tasks, which minimizes the difference between the input data (coding) and output data sets (decoding). In the area of the power system, auto-encoder based applications are also widely verified such as abnormal state detection[119], system state reconstruction[120], and fault diagnose[121] etc. In [8], a deep auto-encoder with greedy layer-wise pre-training and logistic regression at the hidden layer is demonstrated to have excellent performance in terms of DSA problem. As a further exploration of this work, we continue our work based on this deep auto-encoder structure, which is illustrated in Fig.4.3.

Monte Carlo dropout as approximated Bayesian inference

Although auto-encoder is proved to be effective, as stated in chapter II, the limitation of lacking confidence restricts the practical implementation of this approach in the power system. Hence, we use dropout, a commonly used regularisation technique, to transfer a deterministic auto-encoder into a probabilistic Bayesian model.

A traditional neural network is trained by optimizing its parameters directly. To obtain a probabilistic model, a prior distribution is placed over the weights, usually Gaussian distribution: $\mathcal{N}(0,\, I)$. With such, a common neural network is transformed into a Bayesian network. How do we train this type of network? We replace the optimising process by minimising KL divergence between the true posterior $p(\omega|X_{train}^{T_0}, Y_{train}^{T_0})$ and the approximating variational distribution $q_\theta(\omega)$, which is usually referred to as variational inference. Due to its intractable nature, it is extremely difficult to analytically solve this optimization problem. Consequently, this optimization problem is transformed from a KL divergence minimization problem to an Evidence Lower Bound (ELBO) maximization problem. The optimisation function using MC estimator is give by[27]:

$$\hat{\mathcal{L}}_{MC}[\theta] = -\frac{N}{M}\sum_{i\in S}\log p(\hat{y}_i|f^{g(\theta,\epsilon)}(x_i)) + KL(q_\theta(\omega)||p(\omega)) \tag{4.2}$$

with N, M indicating the sub-sampling process and $g(\theta,\epsilon),\omega$ represents the corresponding parameters.

Although the above optimization approach is straightforward, it has the limitation of high computational burden in terms of the practical implementation[116]. Considering this fact, one simplified way is to use dropout as Bayesian approximation. Dropout can be interpreted as equivalent to variational inference. It generates noise into the feature space, from which we can transform it into the network parameter space as illustrated in equation (4.3)[27].

$$\begin{aligned}
\hat{y} &= \hat{h}\boldsymbol{M_2} \\
&= (h \odot \hat{\epsilon_2})\boldsymbol{M_2} \\
&= (h \cdot diag(\hat{\epsilon_2}))\boldsymbol{M_2} \\
&= h(diag(\hat{\epsilon_2})\boldsymbol{M_2}) \\
&= \eta(\hat{x}\boldsymbol{M_1} + \boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2}) \\
&= \eta((x \odot \hat{\epsilon_1})\boldsymbol{M_1} + \boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2}) \\
&= \eta(x(diag(\hat{\epsilon_1})\boldsymbol{M_1}) + \boldsymbol{B})(diag(\hat{\epsilon_2})\boldsymbol{M_2})
\end{aligned} \tag{4.3}$$

In the above equation, we assume an example of a two-layer network with weights $\boldsymbol{M_1}$ and $\boldsymbol{M_2}$ (deterministic matrix), non-linear activation function η, and x, h as input into each layer. We use $\hat{x}$, $\hat{h}$ in order to denote that the input x, h have been through a dropout layer, represented as $\hat{\epsilon_i}$. We write $\hat{W}_1 = diag(\hat{\epsilon_1})\boldsymbol{M_1}$ and $\hat{W}_2 = diag(\hat{\epsilon_2})\boldsymbol{M_2}$, so that it indicates that the network parameters are going through the dropout mask, thus we have:

$$\hat{y} = \eta(x\hat{W}_1 + \boldsymbol{B})\hat{W}_2 = f^{\hat{W}_1, \hat{W}_2, \boldsymbol{B}}(x) \tag{4.4}$$

The minimisation function of a neural network thus can be rewritten from:

$$\mathcal{L}[\boldsymbol{M_1}, \boldsymbol{M_2}, \boldsymbol{B}] = E^{\boldsymbol{M_1}, \boldsymbol{M_2}, \boldsymbol{B}}(x, y) + \lambda_1 ||\boldsymbol{M_1}||^2 + \lambda_2 ||\boldsymbol{M_2}||^2 + \lambda_3 ||\boldsymbol{B}||^2 \tag{4.5}$$

to:

$$\hat{\mathcal{L}}_{dropout}[\boldsymbol{M_1}, \boldsymbol{M_2}, \boldsymbol{B}] = \frac{1}{M} \sum_{i \in S} E^{\hat{W}_1^i, \hat{W}_2^i, \boldsymbol{B}}(x_i, \hat{y_i}) + \lambda_1 ||\boldsymbol{M_1}||^2 + \lambda_2 ||\boldsymbol{M_2}||^2 + \lambda_3 ||\boldsymbol{B}||^2 \tag{4.6}$$

where i indicates each data point from data sub-sampling with a random set S of size M. According to [122], the first term $E^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x,y)$ in neural network optimisation objective function can be rewritten as negative log-likelihood scaled by a constant, as shown in equation (4.7), where τ indicates the observation noise.

$$E^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x,y) = \frac{1}{2}||y - f^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x)||^2 = -\frac{1}{\tau}\log p(y|f^{\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}}(x)) + constant \tag{4.7}$$

Hence we can rewrite equation (4.6) into (4.8)

$$\hat{\mathcal{L}}_{dropout}[\boldsymbol{M_1},\boldsymbol{M_2},\boldsymbol{B}] = -\frac{1}{M\tau}\sum_{i\in S}\log p(\hat{y_i}|f^{g(\theta,\hat{\epsilon_i})}(x_i)) + \lambda_1||\boldsymbol{M_1}||^2 + \lambda_2||\boldsymbol{M_2}||^2 + \lambda_3||\boldsymbol{B}||^2 \tag{4.8}$$

with $\{\hat{W}_1^i, \hat{W}_2^i, \boldsymbol{B}\} = \{diag(\hat{\epsilon_1^i})\boldsymbol{M_1}, diag(\hat{\epsilon_2^i})\boldsymbol{M_2}, \boldsymbol{B}\} = g(\theta, \hat{\epsilon_i})$ represents the parameters.

Comparing equation (4.2) and (4.8), it has been proved that for a specific choice of prior distribution $p(\omega)$, such that:

$$\frac{\partial}{\partial\theta}KL(q_\theta(\omega)||p(\omega)) = \frac{\partial}{\partial\theta}N\tau(\lambda_1||\boldsymbol{M_1}||^2 + \lambda_2||\boldsymbol{M_2}||^2 + \lambda_3||\boldsymbol{B}||^2) \tag{4.9}$$

which is referred to as KL condition, the dropout neural network would have identical optimisation procedure as variational inference[27]. In summary, the minimization function of a dropout approximating network is given as follows[6]:

$$\mathcal{L}[\theta,p] = -\frac{1}{N}\sum_{i=1}^{N}\log p(\hat{y_i}|f^{g(\theta,\hat{\epsilon_i})}(x_i)) + \frac{1-p}{2N}||\theta||^2 \tag{4.10}$$

where N refers to the size of data, p represents the dropout probability and θ is the parameter of the tractable distribution. More details regarding the dropout approximation can be found in the reference [6] and [27].

In terms of the practical implementation, this approximated inference is made by keeping dropout on at both training and testing stage, which is easy to implement. In other words, dropout is done at test stage to sample from the trained network, which can also be treated as

a stochastic feed-forward process.

Combined epistemic and aleatoric uncertainties in one model in classification tasks

There are two types of uncertainties in Bayesian modelling[123]. The *epistemic uncertainty*, as introduced previously, represents the uncertainty in the model parameters. It is of great significance to capture this type of uncertainty since it could reflect the system topology changes. The *aleatoric uncertainty* on the other hand represents noise inherently in the observations and thus can be further categorized into *homoscedastic uncertainty* and *heteroscedastic uncertainty*[6]. For *homoscedastic uncertainty*, the observation noise parameter σ is fixed while the *heteroscedastic aleatoric uncertainty* varies over different periods of time depending on the data itself. Hence it is obvious that *heteroscedastic aleatoric uncertainty* is a more general and realistic situation. In the prior literature, most of the existing BDL approaches can merely capture *epistemic uncertainty* or *aleatoric uncertainty* alone [27]. Hence, it is important to model these two uncertainties together in one model.

However, it is notable that in our work, the main concentration is on using *epistemic uncertainty* to indicate the proper model updating time. In other words, though discussion of the source of inherent noise (*aleatoric uncertainty*) such as sensor noise, missing data points or any kinds of manually added Gaussian noise etc. are important as well, we assume that noise test and missing data test are out of the scope of this part of work and will not be discussed in the following chapters. Due to the page limit, we would like to refer the readers to other valuable works such as [109][124][125].

To combine the *epistemic uncertainty* and the *aleatoric uncertainty* in a single model, we need to split the top layers of a deep auto-encoder network into predictive mean $\hat{y}$ as well as predictive variance $\hat{\sigma}^2$, as follows:

$$[\hat{y}, \hat{\sigma}^2] = f_{BDAC}^{\hat{W}}(x) \tag{4.11}$$

where f_{BDAC} represents the proposed Bayesian deep auto-encoder network. For classification tasks, the output probability is then computed from approximated Monte Carlo integration,

which is as follows:

$$p(\hat{y} = c|x, X, Y) \approx \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} Softmax(f^{\hat{W}}(x)) \tag{4.12}$$

The epistemic uncertainty mu of the trained model can then be calculated using the entropy:

$$H(p) = -\sum_{j=1}^{J} p_j \log p_j \tag{4.13}$$

where J represents the total number of classes.

The overall predictive uncertainty Var$[y]$, consisting of both the *aleatoric uncertainty* and the *epistemic uncertainty* thus can be approximated as

$$\text{Var}[\hat{y}] := H(p) + \frac{1}{T_{sample}} \sum_{t=1}^{T_{sample}} \hat{\sigma}^2. \tag{4.14}$$

Given that a normal likelihood is chosen to model the aleatoric uncertainty, the final loss function of the BDAC can be formulated as:

$$\mathcal{L}_{classification} = \frac{1}{T_{train}} \sum_{t=1}^{T_{train}} (-\hat{y}_{t,j'} + \log \sum_{j} \exp \hat{y}_{t,j}) \tag{4.15}$$

$$\hat{y}_t = f^{\hat{W}} + \epsilon_t, \epsilon_t \sim N(0, (\hat{\sigma})^2) \tag{4.16}$$

with $\hat{y}_{t,j}$ the j element in the logits vector $\hat{y}_t$. Note that the loss function can consider both the model uncertainty through $\hat{y}$ and the heteroscedastic uncertainty through $\hat{\sigma}$.

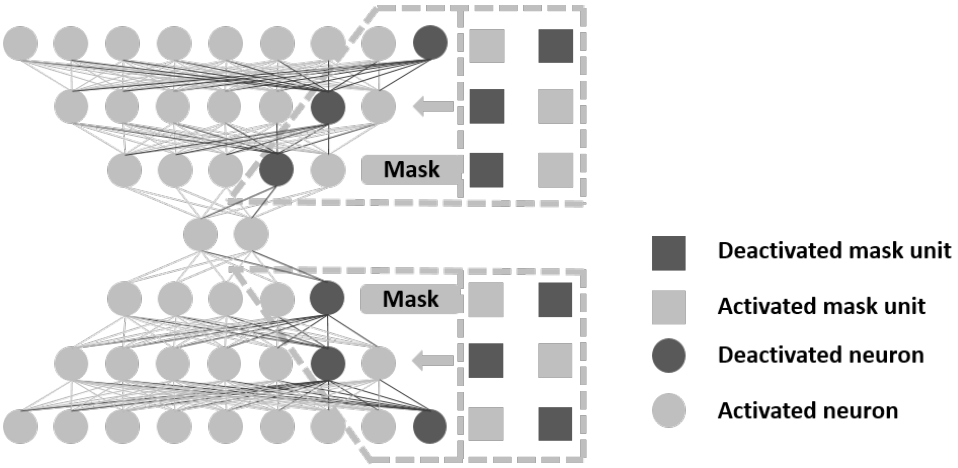


Figure 4.4: Auto-encoder with conditional mask

Conditional mask

As illustrated in chapter II, the corresponding expansion of OCs domain caused by the N-1 criterion will result in a significant increase in training burden when using deep learning approaches. Hence, it is reasonable to explore the feasibility of training one model to learn both the unique contingency information and the common operating information. In Fig. 4.4, we assume a network of L layers where at each layer $l \in \{1, ..., L\}$, the input and the output of layer are x_l and $\hat{y}_l$ respectively. Given a non-linear activation function $\sigma()$, the feed-forward process within one layer can be written as:

$$\hat{y}_l = \sigma(W_l \cdot x_l + \boldsymbol{B}_l) \tag{4.17}$$

where W_l and $\boldsymbol{B_l}$ indicate the weights and bias at layer each layer l

We then define the mask layer h, which is consist of one-hot coded 1's and 0's indicating the lines connection or disconnection respectively[126]. Specifically, the width of the mask layer should be equal to the number of contingency. Hence, neurons with mask layer in front are used specifically to learn contingency information. By doing this, the network is separated into two regions. The region without mask will learn and store the common information of various contingencies, while the neurons with mask covered are not activated unless the labeled

contingency data comes in. As a result, these neurons with mask do not participate in the training process since they have fixed values and thus zero gradient. Therefore, the model can obtain the ability of dealing with multi-contingency data by exploring the inner structure of the network. The equation for one layer of mask becomes:

$$\hat{y}_{lc} = \begin{cases} h_{lc} \cdot \sigma(W_{lc} \cdot x_{lc} + \boldsymbol{B}_{lc}), & c \in [1, C] \\ \sigma(W_{lc} \cdot x_{lc} + \boldsymbol{B}_{lc}), & c \in [C, D] \end{cases} \tag{4.18}$$

In the above equation, $\hat{y}_{lc}$ indicates one particular vector output c (column) of layer l. Similarly, W_{lc} and $\boldsymbol{B_{lc}}$ denote the c_{th} vector of the matrix of weights and bias respectively. h_{lc} represents the mask (activation rules) at layer l, colomn c. C denotes the contingency number and D represents the width of the network.

4.2.5 Model Updating Stage:

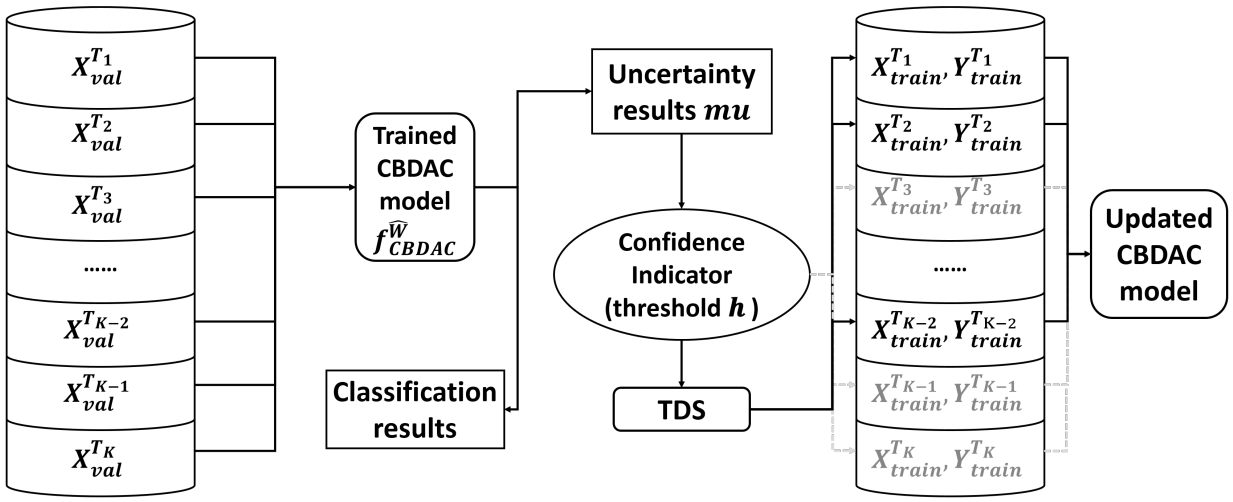


Figure 4.5: Model updating stage.

When system topology changes, the decision of whether updating the model becomes vital. The practical industrial procedure is to update the model parameters following an experience-based timely basis. Although the current strategy has the advantage of simplicity, it does not participate in or even has a negative contribution to the whole system operation. Hence by

employing the property of confidence awareness, the core task of this stage is to provide an indicator of updating, so that redundancy work can be avoided. In addition, when the model is indicated to be updated in a practical situation, TSO is always required to finish the updating work in limited time. Given that the TDS process of simulating a large training database is a practical bottleneck, the amount of data that required to update model will play a key role.

As can be seen from Fig.4.5, validation sets $X_{val}^{T_k}$ from different system topology $T_0, T_1, ..., T_K$ are imported into the model firstly. By doing this, an initial classification result $\hat{y}_i$ together with confidence information (model uncertainty) mu, which is defined in equation (4.13), will be generated and calculated. The confidence information indicates how well the model 'recognizes' the input data. In other words, if the system topology changes, due to the change of data distribution characteristics, mu will increase, which implies that the model does not 'recognize' the data, or feel 'not confident' about its prediction results. It is important to emphasize here that since the TDS are not yet processed for the new topology, the initial uncertainty results can be obtained almost immediately as it only requires stochastic feed-forward calculations based on the CBDAC trained by the original training data. This is of significant importance in terms of rapid, frequent topology changes in the future power system since the reaction time of such an event would be limited. Confidence indicator with a threshold value h is then used to determine whether to update the model. A small number of training labels $Y_{train}^{T_k}$ would be simulated through TDS only when the threshold value h is violated. Therefore, ($X_{train}^{T_k}$, $Y_{train}^{T_k}$) from new topology will be employed in order to update the model, where k represents the number of topologies. Under this circumstance, only essential updating work will be implemented, and the updating task can be guaranteed with reasonable computational burden and accuracy. To summarize, algorithm 1 demonstrates how the proposed framework works in detail.

Algorithm 1 Confidence-Aware DSA Framework

Require: $X_{train}^{T_0}$, $Y_{train}^{T_0}$, $X_{val}^{T_k}$, $Y_{val}^{T_k}$, $X_{train}^{T_k}$, $Y_{train}^{T_k}$
Ensure: $X_{train}^{T_0}$, $Y_{train}^{T_0}$, are from topology T_0; $X_{val}^{T_k}$, $Y_{val}^{T_k}$, $X_{train}^{T_k}$, $Y_{train}^{T_k}$, are from topology T_1 to T_K
Define learning rate λ, dropout p, data size N, batch size M, optimizer *Adam* etc.
Initialize all parameters
for CBDAC($X_{train}^{T_0}$, $Y_{train}^{T_0}$) **do**
  repeat
    Mini-batch M optimisation through $\hat{\epsilon} \sim p(\epsilon)$
    Calculate derivation w.r.t. θ
    $\hat{\delta\theta} \leftarrow -\frac{1}{M\tau} \sum_{i \in S} \frac{\partial}{\partial \theta} \log p(\hat{y}_i | f^{g(\theta, \hat{\epsilon}_i)}(x_i)) + \frac{\partial}{\partial \theta} N\tau(\lambda_1 ||\boldsymbol{M_1}||^2 + \lambda_2 ||\boldsymbol{M_2}||^2 + \lambda_3 ||\boldsymbol{B}||^2)$
    Update θ: $\theta = \theta + \lambda\delta\theta$
  until θ has been optimised

  return $f_{CBDAC}^{\hat{W}}$
end for
Define threshold value h
Proceed stochastic feed-forward through $f_{CBDAC}^{\hat{W}}$ using $X_{val}^{T_k}$, $Y_{val}^{T_k}$
for Confidence Indicator(mu, h) **do**
  if $mu > h$ **then**
    Proceed CBDAC($X_{train}^{T_k}$, $Y_{train}^{T_k}$)
  else
    Model can be kept
  end if

  return Updated model
end for

4.3 Case Study

4.3.1 Data Descriptions

The numerical experiments conducted in this study are based on the example 68-bus system [4], which is illustrated in Fig 4.6. To simplify our work, we assume that PMUs devices and other measurement devices are deployed to conduct real-time measurement. For instance, voltage magnitudes, phases angles of all buses and the active and reactive outputs of generators can be directly measured and transmitted. Power flow data, on the other hand, can be calculated from the solver. Also, to enrich the OCs domain, more OCs are simulated from a pre-defined range of distribution.

From the system, a set of 12000 observations are sampled, where each observation represents a pre-fault OC. These observations are created by drawing the active load power from a multivariate Gaussian distribution and using a Pearson's correlation coefficient c between all power pairs. These active load power are then converted to a marginal Kumaraswamy distribution with the probability density function:

$$f(x) = abx^{a-1}(1-x^a)^{b-1} \tag{4.19}$$

where $a = 1.6$, $b = 2.9$ and $x \in [0, 1]$. The active load power is scaled to be within $\pm 50\%$ of the nominal values, while the reactive load power is scaled by assuming constant impedance of buses. Considering the fact that the resulting OCs might be infeasible and also to restrict the sampled power factor of generators within the range of [0.95, 1], an optimization is solved accounting for the full AC network model. The optimization is carried on in Python 3.5. with Pyomo package and the IPOPT 3.12.4 solver. In order to obtain more general test conditions, the transient stability of three-phase faults is simulated over 22-line contingencies, where the work in [103] has proved the effectiveness of the selected contingencies. These 22-contingencies are selected based on the rules in [127]. For instance, in terms of the fault location, only those close to generator buses are considered, and the fault clearing is coupled with line tripping. The reason is that these are the cases of rotor angle stability interest. As a comparison, cases, where faults are close to loads, are from the angle of voltage stability[128]. The first 14 contingencies are shown in table III in [127], which are within the NETS part of the IEEE-68 bus system. The rest contingencies are selected based on the same rationale, but for the NYPS part. An OC is considered stable if the differences between each two-phase angles of the generators are within the corresponding limits during 10s simulation time, otherwise unstable. The fault clearance time is assumed to be 0.1s. The simulation is performed in Matlab R2016b Simulink, and the model used is described in [4].

To establish the database, data from each contingency are finely shuffled firstly in order to provide randomness and generalization. 80% of the entire database (i.e. 80%×12000×22) is used as the training set to train the model, and the rest 20% of the database is used as the testing set.

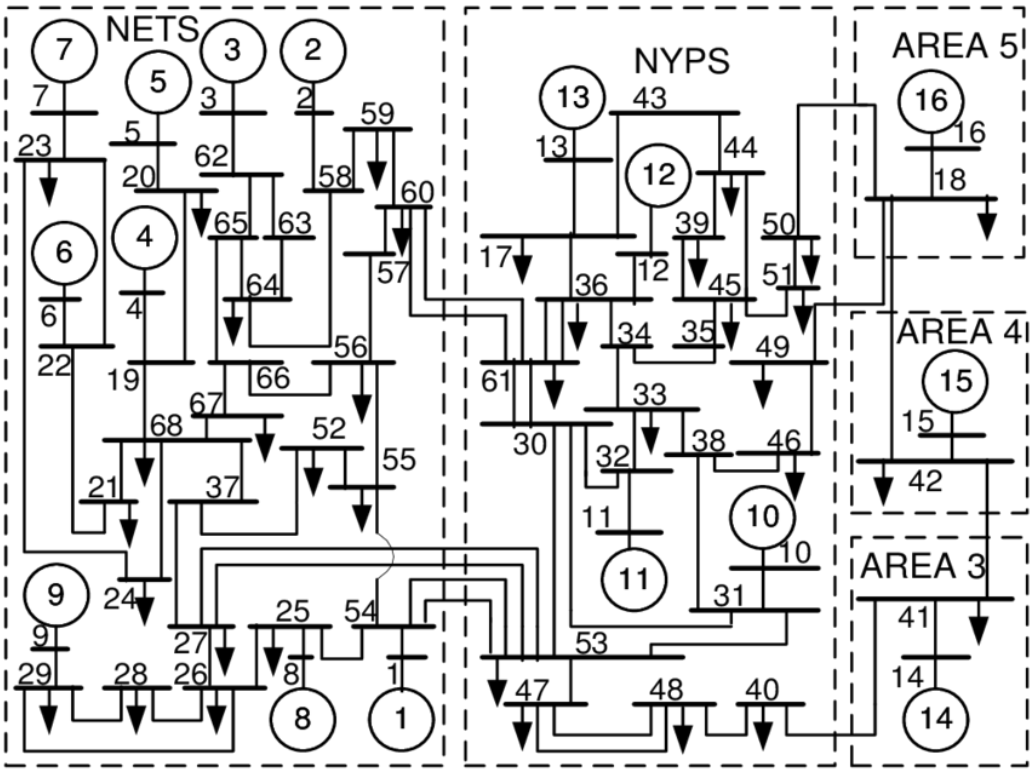


Figure 4.6: IEEE-68 bus system[4].

4.3.2 Experimental Setup

To demonstrate the superior performance of the proposed approach, a series of state-of-the-art methods that have been widely used and firmly demonstrated with reliable performance are used for comparison. For the rest of this section, the following notation will be used. For instance, DT (*Decision Tree*) SVM (*Support Vector Machine*) RF (*Random Forest*) DAC (*Deep Auto-Encoder Classifier*) BDAC (*Bayesian Deep Auto-Encoder Classifier*) and CBDAC (*Conditional Bayesian Deep Auto-Encoder Classifier*). All the methodologies mentioned above are implemented in Python with the main packages of Scikit-learn [82], Keras [129], TensorFlow [130] and run on an Intel Xeon PC with NIVIDIA Titan-V GPU. The hyper-parameters of the proposed CBDAC model is determined by grid search and cross-validation, which are given in Table 4.1. The employed evaluation metrics are precision (PRE), specificity (SPE), F1-Score and accuracy (ACC) respectively, where detail introduction is given in the next section.

Table 4.1: Hyperparameters of the Proposed CBDAC

Parameter	Value
Layer type	dense
Number of hidden layers	13
Encoder structure	450-350-250-150-50-10
Decoder structure	10-50-150-250-350-450
Logistic regression layer	2
Batch size	15000
Number of epochs	160
Number of samples (T_{sample})	200
Dropout rate	0.001
Optimizer	Adam
Normalisation	[0,1]normalisation
Learning rate	0.0001

4.3.3 Evaluation Metrics

In this section, the concept of the confusion matrix and four evaluation metrics are introduced to evaluate the performance of security assessment. Given a set of input data, four different types of results can be obtained, which are denoted by True Positive(TP), False Positive(FP), False Negative(FN), and True Negative(TN). For instance, TP represents that unsafe OCs are correctly predicted as unsafe, TN represents when safe OCs are correctly predicted as safe. The wrong results are further grouped into FP, which represents unsafe OCs incorrectly predicted as safe, and FN, when safe OCs are incorrectly predicted as unsafe. The proposed four evaluation metrics are calculated based on these four variables. For instance:

(1)The precision: the proportion of the correctly predicted unsafe OCs in all the actual unsafe OCs.

$$precision = TP/(TP + FP) \tag{4.20}$$

(2)The specificity: the proportion of the correctly predicted safe OCs in all the predicted safe OCs.

$$specificity = TN/(TN + FP) \tag{4.21}$$

(3)The F1-Score: the comprehensive evaluation of the the precision and the recall. (Recall =

TP / (TP + FN))

$$F1 - Score = 2 \times PRE \times REC/(PRE + REC) \tag{4.22}$$

(4)The accuracy: the proportion of correct classification results over all output results.

$$accuracy = (TP + TN)/(TP + FP + FN + TN) \tag{4.23}$$

4.3.4 Case Study 1: CBDAC Classification Performance without Topology Changes

In this test, we aim to compare the classification performance of the proposed CBDAC method with other popular methods. As introduced in section IV A, 80% of the database is used to do the training work. Fig. 4.7 presents the testing results of the four evaluation metrics, where the testing is based on the data size of 20%×12000×22. The length of the blue bar represents the value of the evaluation metric (i.e., a higher value corresponds to a longer bar). CA at the last column indicates the ability of confidence awareness. In terms of the classification performance, the results show that the CBDAC model dominates when compared with the most popular method DT with 17.03%, 21.98%, 15.56%, 20.55% improvements, respectively. In addition, the performance of CBDAC also dominates when comparing with the best of the state-of-the-art methods, RF, especially when considering the fact that the performance is already approaching the limit, with approximately 6.52%, 11.40%, 7.91%, and 8.04% improvements for the four evaluation metrics. Moreover, only Bayesian models have confidence awareness capability. The fact that Bayesian methods provide the best performance indicates the significance of capturing uncertainties.

Regarding the computational time, which is presented in Table 4.2, the experiment shows that DT and RF methods have the shortest training time, 31 and 59 seconds, respectively. In contrast, SVM has the heaviest computational burden, approximately 18,345 seconds. It can

	ACC	PRE	SPE	F1-Score	CA
DT	81.86%	78.89%	84.02%	78.96%	×
SVM	89.30%	88.77%	91.33%	87.73%	×
RF	89.94%	86.38%	89.97%	88.11%	×
DAC	88.58%	84.51%	88.64%	86.45%	×
BDAC	95.12%	94.65%	95.92%	94.36%	✓
CBDAC	95.80%	96.23%	97.09%	95.19%	✓

Figure 4.7: Classification results for different methods.

be seen that deep learning methods require longer training time than most of the benchmark approaches, where DAC BDAC and CBDAC consume 763, 1127 and 831 seconds, respectively. However, it is notable that model training is an offline procedure. Given the input dataset, the security assessment task can be finished within seconds in practical use. Therefore, the main target in this case is to obtain an accurate classification result.

Table 4.2: Computation Time For Model Training

	CPU Time (s)
DT	31
SVM	18,345
RF	59
DAC	763
BDAC	1,127
CBDAC	831

4.3.5 Case Study 2: CBDAC VS. Other Methods under 44 Different System Topology

In this test, we aim to evaluate the performance of the proposed CBDAC method and other methods when facing system topology changes. The trained models shown in case study 1 are directly used in this test without re-training. OCs data from new topology cases are sampled following the same rationale as introduced in section IV A. We assume system topology changes by switching off lines between buses. We have generated 44 different topology cases with various similarity to the original system topology. For instance, by observing the system structure from [4], we can find that there is a double line scheme between bus NO.27 and NO.53, hence

the disconnection between these two buses might have a slight influence on the rest area of the network. On the other hand, bus NO.17 has the maximum load within the system, which implies that disconnection occurred here could cause severe power flow pattern changes. It is notable that due to the extreme time cost, it is unreasonable to keep considering the SVM method.

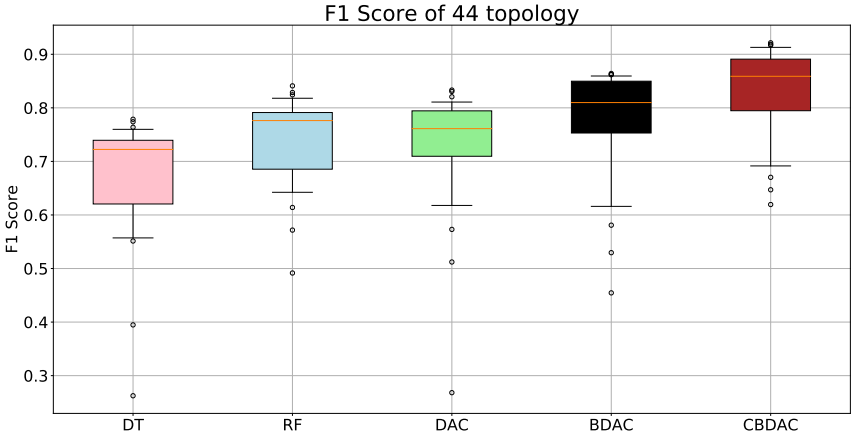


Figure 4.8: Box plots of F1 Score for 44 different topology.

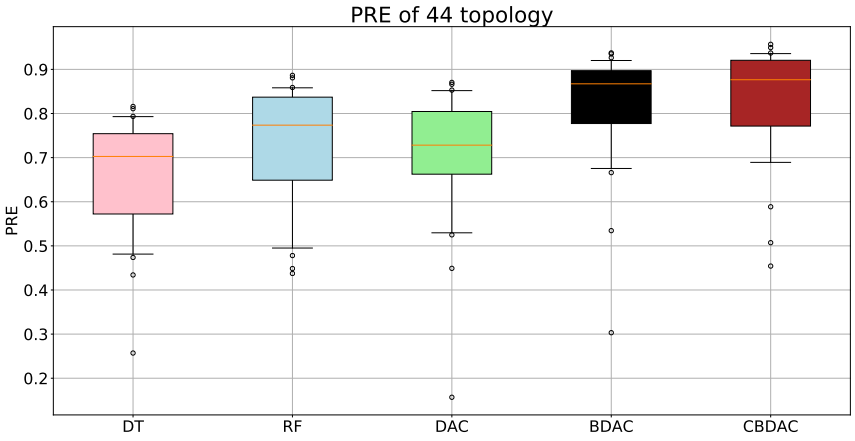


Figure 4.9: Box plots of PRE for 44 different topology.

The general experiments results are shown in Fig 4.8-4.11, where Bayesian methods show superior performance than the DT, RF and DAC models. For instance, as illustrated in Fig 4.8(a), in terms of the comprehensive evaluation F1-Score, DT has the lowest score, an average

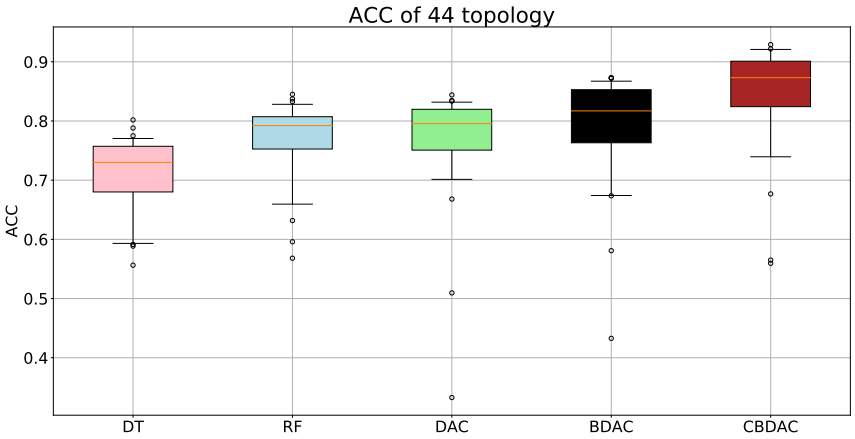


Figure 4.10: Box plots of ACC for 44 different topology.

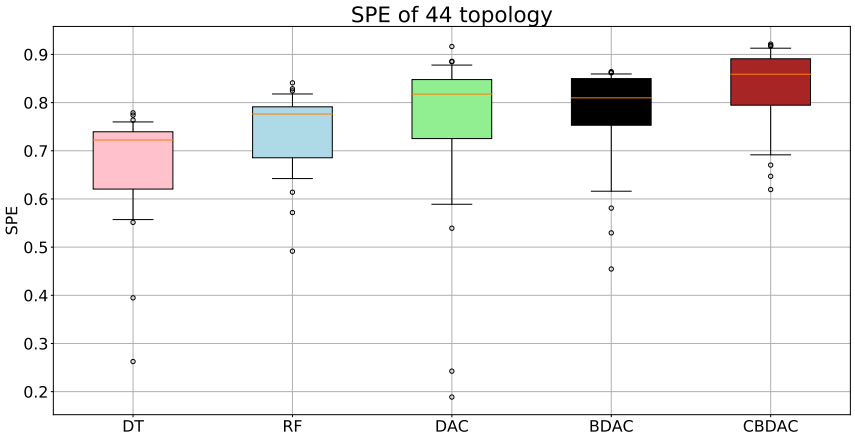


Figure 4.11: Box plots of SPE for 44 different topology.

of 67.80% while RF has a slightly higher performance of 74.45%. Deep Learning methods show significantly better results such as DAC at 77.51% and BDAC at 78.06%. Our improved CBDAC method has the best average performance of 83.31%. CBDAC also outperforms in terms of other metrics, which is demonstrated in Fig 4.8(b)(c)(d) respectively. Table 4.3 also shows several individual results of CBDAC model. More importantly, if the acceptable accuracy level is set to be 0.8 F1-Score, we can find that none result from DT method locates above it. RF has a proportion of 20.45% (9 of 44) that goes over the threshold. DAC and BDAC methods have 45.45% (20 of 44) and 54.55% (24 of 44) results that go beyond the threshold value respectively.

CBDAC has 70.45% (31 of 44) above the threshold.

The result proves that the Bayesian model with improvement still has superior performance when facing system topology changes, which further implies that resources can be saved if unnecessary updating work can be avoided.

Table 4.3: Different Topology Cases for CBDAC

CBDAC				
Case	**F1-Score**	**ACC**	**PRE**	**SPE**
27-53	92.15%	92.91%	92.52%	93.85%
65-64	91.71%	91.91%	93.71%	94.02%
63-64	91.86%	92.18%	92.03%	92.64%
17-36	64.70%	80.41%	69.30%	88.70%
48-40	79.92%	85.41%	82.66%	90.25%

4.3.6 Case Study 3: Epistemic Uncertainty as Model Updating Indicator

So far we have proved that the Bayesian deep learning method with improvement is advance and also robust when faced with system topology changes. However, the essence of forecasting (classification) itself determines that we will never know whether the next prediction is true until we know the results. Hence, a reliable auxiliary indicator, which has the ability to show the 'confidence' of the forecasting result becomes vital. Bayesian method thus shows its second advantage: the ability to represent the prediction confidence using model uncertainty as the indicator. The model uncertainty mu, as previously illustrated in equation (4.13), represents the domain knowledge learned by the model. In other words, the uncertainty of the Bayesian model can reflect the similarity of the original system topology and the new topology, i.e. a big change of topology means a significantly higher uncertainty.

In this case study, BDAC and CBDAC are used since they are Bayesian-based approaches. It is notable that in this work, we only consider the topology changes that are pre-defined by the TSO. As can be seen in Fig.4.5, when system topology changes, a small number of OCs are feed into the trained CBDAC model firstly, and the uncertainty results can be generated and

collected. During this process, we find that the generated uncertainty values are around a certain level. Compared with the uncertainty level of a finely trained model at the original topology, it is found that the uncertainty level is correlated with the model performance at each topology case. It is also implied that the level of uncertainty is determined by the characteristics of the database (system topology characteristics). In other words, when it comes to another network (e.g. a 108-bus system), there will be another uncertainty level, which could also be decided by the validation data set $X_{val}^{T_k}$ from the corresponding topology cases. Therefore, considering the trade-off between the model accuracy and the updating work burden, we found that 0.8 F1-Score is a reasonable setting.

In this test, if 0.8 F1-Score is used to distinguish safe and unsafe, 20 out of 44 topology cases will be identified as unsafe when using BDAC model. Similarly, in terms of CBDAC model, 13 topology cases have F1-Score lower than 0.8. The performance of the proposed two models under 44 different topology cases and the effectiveness of uncertainty indicator is demonstrated in Fig 4.12-4.15. However, it is notable that the uncertainty might not be sensitive enough when it comes to each individual cases. Instead, it is more reasonable to evaluate various set of topology cases and calculate the average performance. For instance, though fluctuation could be observed, as a general calculation, the first 20 topology cases have an average uncertainty of 0.171, which is larger than the average of the last 24 cases: 0.149, in the BDAC figure Fig 4.13. In terms of CBDAC Fig 4.15, 0.172 VS 0.154 are the average uncertainty values for the first 13 cases and the rest.

Therefore, we can conclude that if 0.8 F1-Score and 0.16 of uncertainty are chosen to be the threshold value, we can observe that most of the topology with poor performance can be detected. This 'detection' property has significant importance since it is generated simultaneously with the classification results. In other words, in reality, the system operator can have confidence information on the model's prediction, which allows him to decide whether to trust or update the model.

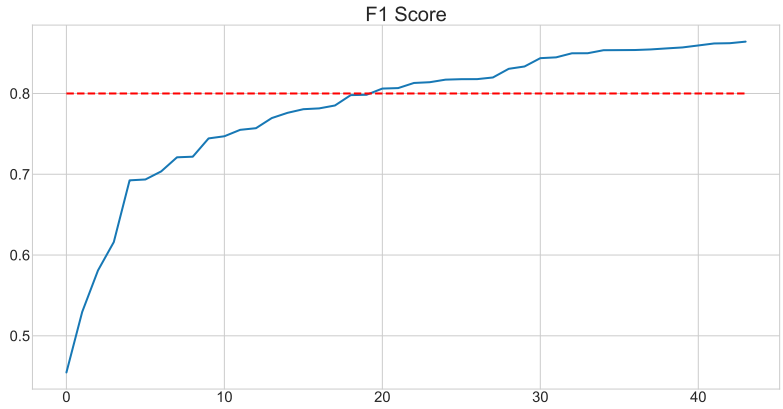


Figure 4.12: F1 Score of BDAC under 44 different topologies.

Figure 4.13: Uncertainty of BDAC under 44 different topologies.

4.3.7 Case Study 4: Model Updating Using Small Data

In this test, we explore the situation when only limited data is available to update the model. In terms of new topology cases, 800 OCs are randomly sampled for each contingency (i.e. a total of 800×22=17600). Following the rationale before, the new database is finely shuffled in order to provide good randomness and generalization. The first 100 OCs of each contingency are used to generate the uncertainty information (i.e. 100×22=2200). The last 400 OCs of each contingency (i.e. 400×22=8800) are used as the testing set. The rest 300 OCs thus are used as the updating data set. Therefore, we design four different updating scenarios, where 30 OCs, 100 OCs, 200 OCs, and 300 OCs are used to update the model, which generates data sample

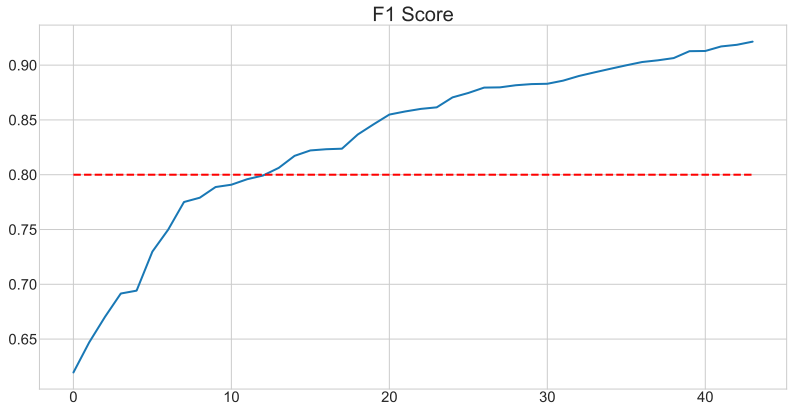


Figure 4.14: F1 Score of CBDAC under 44 different topologies.

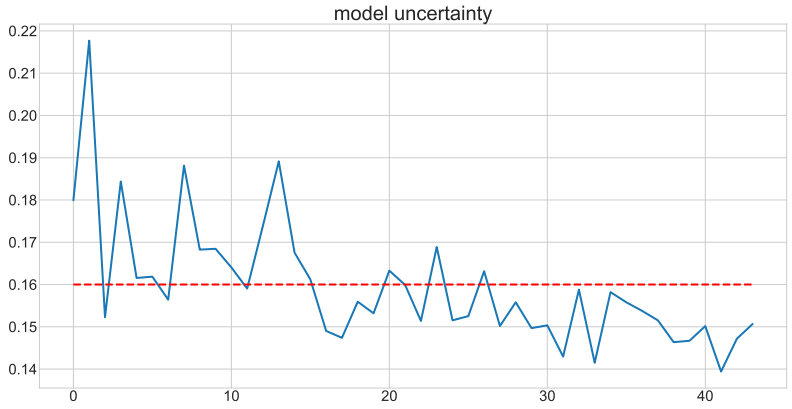


Figure 4.15: Uncertainty of CBDAC under 44 different topologies.

size of 660, 2200, 4400, 6600 respectively. The OCs from each scenario are sampled uniformly except the 300 OCs case so that the test is convincing.

We choose DAC, BDAC, CBDAC in the experiments as a comparison. The evaluation results are shown in Fig 4.16 to 4.19. Firstly, we can observe that the classification performance can be improved by updating the model with a few data points. The performance is enhanced as the number of data increases, which is in line with our intuitive speculation. However, one interesting phenomenon here is that when using extremely small size data (e.g. 30 OCs), the performance becomes unstable. For example, the original F1-Score performance of topology NO.10 and NO.11 are 0.9215 and 0.8615. After updating with 30 new OCs, their new performance is 0.8412

and 0.7896 respectively. Considering this phenomenon, it is reasonable to think about a more efficient updating strategy, with the ability to distinguish the necessity of updating the model. The Bayesian model thus is able to fulfil our requirement.

Table 4.4: Performance of Different Updating Strategies: 100 OCs

100 OCs Updating				
	F1-Score	**ACC**	**PRE**	**SPE**
Original	83.31%	84.93%	82.93%	84.64%
All	86.30%	87.88%	83.23%	85.03%
Stochastic	84.08%	85.72%	83.10%	84.54%
Selective	86.10%	87.18%	86.03%	86.80%

Table 4.5: Performance of Different Updating Strategies: 30 OCs

30 OCs Updating				
	F1-Score	**ACC**	**PRE**	**SPE**
Original	83.31%	84.93%	82.93%	84.64%
All	82.40%	84.90%	78.11%	81.16%
Stochastic	82.33%	84.31%	80.45%	83.15%
Selective	84.79%	86.08%	84.77%	85.71%

Based on our previous experience from case study 3, it has been found that the uncertainty level is mainly decided by the characteristics of the network. In terms of the example IEEE 68-bus system, we find that 0.16 uncertainty could be a proper separation limit, which identifies 16 cases to be updated. It is notable that this time we only focus on the extremely small size data, i.e. 30 and 100 OCs. As a comparison, we select 16 cases stochastically in order to prove the effectiveness. Table 4.4 and 4.5 demonstrate the average performance of four evaluation metrics in terms of different updating strategies. For instance, traditional updating strategy (i.e. update all) works only if there is sufficient data. However, considering the corresponding time consuming of 2200 data points during the TDS, the practical value of this strategy remains questionable. On the other hand, updating the model selectively is proved to have not only equally robust performance but also significantly fewer time consumption. For example, in our test, only 16 out of 44 topology cases are required to do the TDS, which means the time consumption is only 36% of traditional updating strategy, not to mention the part of re-training the model. Furthermore, comparing to the traditional strategy, the selective updating strategy still has reliable performance, even with significantly fewer data points.

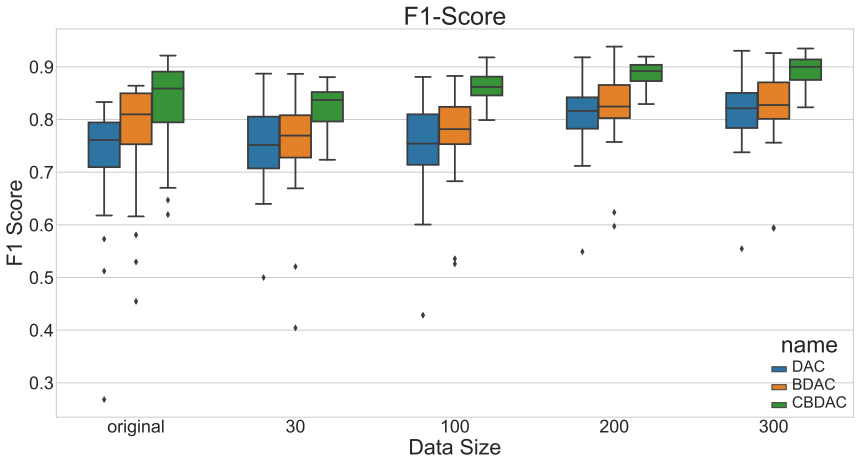


Figure 4.16: Model updating with different data size: Box plots show the evaluation of F1-Score

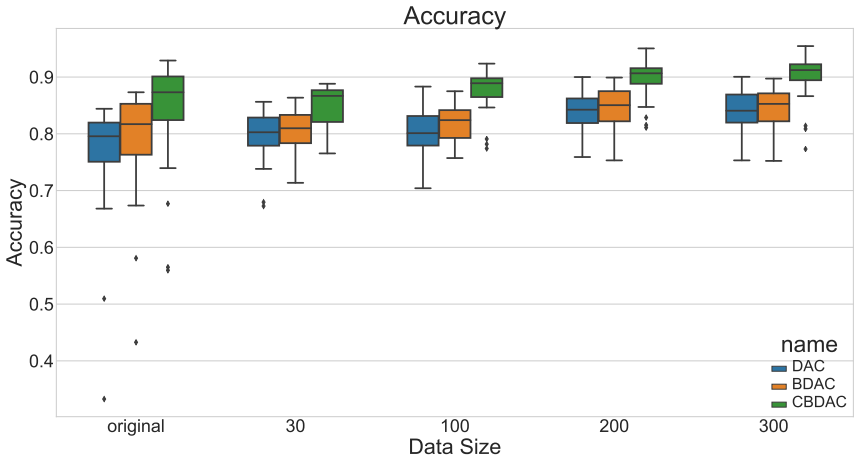


Figure 4.17: Model updating with different data size: Box plots show the evaluation of ACC

To verify the effectiveness of computational cost reduction of the proposed updating strategy, we illustrate the simple calculation process which is based on several assumptions. Assume the model training time is T for one topology and the TDS time for calculating the label is S for one OC. Comparing to T and S, the model initialization time and the feed-forward time can be neglected. The traditional updating strategy requires the model to be trained every time the system topology changes based on the full-sized database, which means the full-sized database TDS is also needed. In our experiments, the total time consumption thus should be $44 \times (T + 300 \times 22 \times S)$, where 300×22 OCs are used for 22 contingencies. This

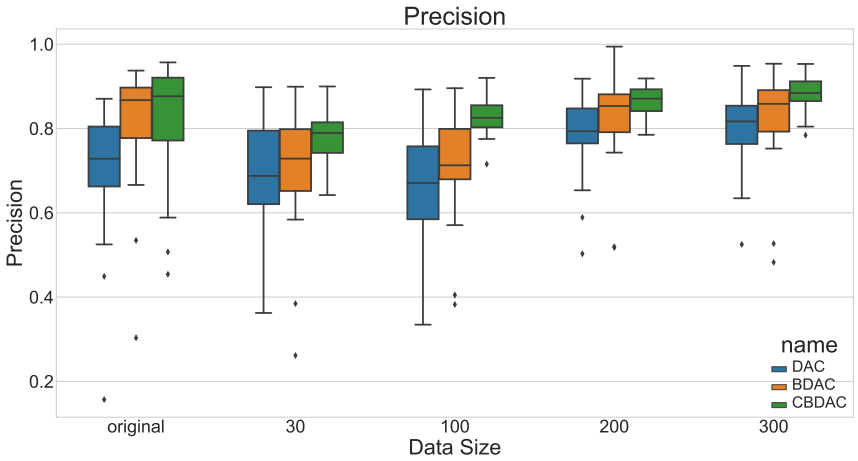


Figure 4.18: Model updating with different data size: Box plots show the evaluation of PRE

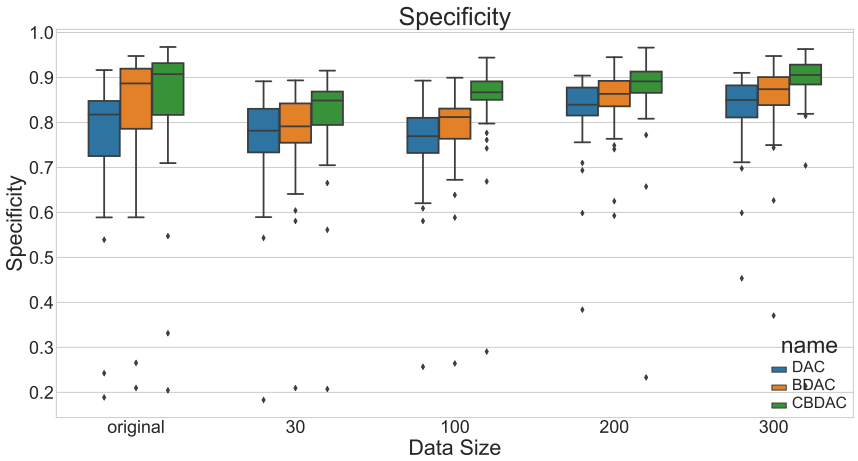


Figure 4.19: Model updating with different data size: Box plots show the evaluation of SPE

results in a significantly high computational cost, thus will not be considered. Instead, by using small size database, the computational cost can be reduced to $44 \times (T_{100} + 100 \times 22 \times S)$ or $44 \times (T_{30} + 30 \times 22 \times S)$, where T_{100} and T_{30} indicate the training time using 100 or 30 OCs from each contingency. The training time is also reduced as the data size is reduced, which means $T_{30} < T_{100} < T$. By using the proposed updating strategy, only 16 out of 44 topology need to be updated which further reduces the computational cost to $16 \times (T_{100} + 100 \times 22 \times S)$, or $16 \times (T_{30} + 30 \times 22 \times S)$. Comparing the proposed 'Selective' updating strategy and the 'All'

strategy, the approximated computational time saving can be calculated as follows

$$\begin{aligned}&(1-\frac{16\times(T_{30}+30\times22\times S)}{44\times(T_{100}+100\times22\times S)})\times100\%\approx\\&(1-\frac{16\times(30\times22\times S)}{44\times(100\times22\times S)})\times100\%=89.09\%\end{aligned}\tag{4.24}$$

where we find in our experiments that $T_{30}\ll30\times22\times S$, and $T_{100}\ll100\times22\times S$.

4.3.8 Discussion

The proposed Conditional Bayesian Deep Auto-Encoder based DSA classifier has shown promising performance. Specifically, in this work, one practical problem we would like to solve is how to avoid unnecessary cost when system topology changes? We solve the problem by using the confidence as updating indicator and small size data.

Given the complexity of a real power grid, it is usually infeasible for the traditional approaches to scale and adapt to a larger power network. For a medium-sized system such as the IEEE-68, the use of a pre-trained classifier can be much faster than using optimization approach since it solely involves the evaluation of a small number of inequality statements (i.e. security rules). This advantage persists to much larger systems where the computational burden of optimization problems may scale in a non-linear fashion. It can be particularly important when operating in a real-time fashion where the available computational time budget is limited, and the list of contingencies to be checked might include hundreds or thousands of potential faults. However, the most fundamental benefit of the proposed workflow is that it can be readily extended to other types of stability indicators that cannot be determined via optimization but only via TDS (e.g. angle stability, small-signal stability, transient stability etc.). As presented in [101], performing such simulations in real-time is prohibitively slow, which is why an offline analysis must have been carried out beforehand.

The limitation of the work lies in: (1) The database to be used in a ML task is usually collected from various scenarios in advance, and the training works are also done offline. In other words, in

terms of the basic DSA task, there will be enough time for the TSO to do the database collecting and updating works. In addition, to enrich the OCs domain, more OCs are simulated from a pre-defined range of distribution. These OCs are sampled in order to cover the domain that is potentially to occur in the near future, thus can make the database more effective. However, TSO might only have limited reacting time when there is a topology change. Considering the fact that different topology cases might lead to various reacting time, it is possible that practical updating work could be challenged. (2) The uncertainty threshold value is selected based on experience. Therefore, it is imperative to investigate an analytical method to identify the appropriate threshold for the proposed CBDAC method. (3) More comprehensive evaluation metrics should be proposed or employed to deal with the imbalanced problem of DSA.

4.4 Conclusion

Machine learning approaches have been proved to be promising in terms of forecasting and classification tasks. Predictably, it will play an important role in the future power system. However, traditional machine learning techniques are lack of capability in confidence awareness, which is of great importance for TSOs to understand whether the data-driven model is certain about its prediction. As a response, this work proposes a confidence-aware machine learning framework for DSA based on the Conditional Bayesian Deep Auto-Encoder network. The proposed CBDAC model uses dropout to achieve Bayesian approximation with further improvement of conditional training. The superiority and robustness of the proposed methodology are demonstrated with comparison to a series of state-of-the-art methods. We have shown that comparing with the best of state-of-the-art methods, our proposed model still has 6.52%, 11.40%, 7.91% and 8.03% improvement in terms of four evaluation metrics. Furthermore, we explore the feasibility of using limited data in order to update the model and thus propose a selective updating strategy. Indicated by the model confidence, the proposed strategy significantly alleviates the unnecessary time consumption, approximately 89%, under frequent system topology changes, which is of great practical value.

Chapter 5

A Bayesian Deep Reinforcement Learning-based Resilient Control for Multi-Energy Micro-gird

5.1 Introduction

Secure and reliable power system operation is crucial to the modern world as the society depends on electricity increasingly. In recent years, due to the climate change, people have witness of more often extreme weather events, which is expected to occur with wider range in the foreseeable future. Although power systems are designed with a certain extend of ability to resist potential component outage(usually refer to as the N-1 security principle), many natural disasters or malicious attacks could result in more serious damages to the power systems, which is beyond the capability of the power system self-healing. Many blackouts are due to these natural disasters such as the 2005 Hurricane Katrina, 2011 Japan Earthquake and the most recent 2021 Texas power crisis. In the future, the increasing penetration of multi-energy sectors will bring major flexibility challenges but also provide the opportunity for the researchers to consider enhancing the system resilience in different strategies. To this end, an integrated

optimal resilient control scheme of multi-energy systems will play a vital role contributing to a safe, cost-effective and low-carbon energy future.

Researchers have contributed significant efforts to this field and several meaningful works have been highlighted in this section. In [11], the authors give a comprehensive review of the existing works with the aim of clarifying ambiguity. In general, enhancing resilience can be classified into two aspects: 1) the system hardening and 2) the operational resilience strategies. System hardening is defined as increasing the construction standards and system protection level so as to make the system infrastructure less vulnerable to extreme events [131]. However, the trade-off between the system resilience level and the cost is always challenging since one hardening strategy could be effective only to one particular case [11]. Instead, smart grid technologies that is regarded as operational resilience strategies have attracted lots of attention, such as topology reconfiguration, micro-grids etc.

For instance, in the existing literature, [132] and [133] use mixed-integer linear programming (MILP) method to enhance power system resilience by changing network topology, generation re-dispatching etc.. The latter specifically focuses on the situation against ice storms. Micro-grids are able to support their own critical power demand. Under some conditions, micro-grids can even support the main grid or other micro-grids during an extreme event [134]. Research works such as [135] and [136] illustrate good examples. In particular, [137] analyzes the effectiveness of networked micro-grids towards large scale power system resilience enhancement. [14] and [138] propose to study the strategies for service restoration in the distribution system by using micro-grids. [15] and [139] investigate the situation when micro-grid is operated in island mode due to a scheduled interruption from the main grid. In this case, micro-grid is required to serve critical loads, until back to connected mode. The work focuses on the day-ahead scheduling, thus the real-time emergency reaction stays unclear. In addition, distributed generator is able to serve as backup capacities, which is usually in the form of controllable fossil fuel generators[134]. Energy storage of electricity, heating and gas could also provide the same function, such as the discussion in [140]. There are also important works discussing about preventive strategies. For instance, the authors in [141] propose a linear programming approach aiming at determining

the capacity and location of the BESS. The work [142] focuses on the cyber side when there are bad data injections attacks. In [143], the authors propose a framework of three stages preventive control, emergency control, and restoration, considering the typhoon path uncertainty. Moreover, [144] discusses a worse scenario when extreme events such as hurricane happens in a sequential manner.

Although the aforementioned works have shown effectiveness to enhance the resilience of power systems, these methods require accurate system knowledge in order to build the model. In addition, optimisation-based approaches could be less computationally efficient especially for large-scale, multi-sector power systems. Therefore, data-driven methods that do not depend on accurate system parameters have attracted increasing attention in recent years.

Reinforcement learning is a model-free and data-driven approach thus provides a solution that alternative to the model-based approaches. Recent works have shown that reinforcement learning has great potential in terms of real-time system control and decision making. For instance, [145] and [146] report to use reinforcement learning to solve voltage regulation task. This is due to the fact that reinforcement learning can avoid exploiting the exact system models thus could be computationally efficient. There are applications of reinforcement learning in other aspects as well. Literature [24][147] and [148] contribute to the smart building energy management problems. In particular, [24] improves the existing DDPG method in order to obtain better policy quality and accelerates the training process by using prioritized sampling. Works such as [149] have shown the potential of reinforcement learning in the field of electricity market. In [150], it is proposed to use reinforcement learning as a control scheme for the electric vehicles charging. Moreover, reinforcement learning can also be used to enhance resilience. The authors in [151] use deep reinforcement learning to manage the planing strategy of shunts to enhance resilience against multiple line failures. Besides preventive planning, reinforcement learning can also be used as resilient operation controller. Examples such as [152] develops a real-time service restoration controller that is able to manage generation and storage units after power outages. Other works such as [153] investigates the multi-agent resilient control under the situation when micro-grid is operated in island mode.

The existing research has successfully demonstrated the superior performance of reinforcement learning on various tasks. However, most of the studies are based on deterministic models, which is lack of the ability to capture uncertainty. Bayesian Reinforcement Learning (BRL) has exhibited the superior performance in dealing with uncertainties in sequential decision-making, understanding generalization and inferring the proper distribution function between action and state spaces, which is always regarded as leading to a more interpretable deep neural network through the lens of probability theory.

Currently, there are a few important works of Bayesian reinforcement learning. For instance, [154] proposes Bayesian Deep Q-Network (BDQN), where the authors address the issue of inefficient sampling of the posterior. [155] focuses on the model-free reinforcement learning algorithms. A good industrial example of autonomous vehicle safety is discussed in [156]. Despite the aforementioned works, there is only few contribution of Bayesian learning in the area of power system including [116] and [157].

To fill the gap of the knowledge, in this part of work, a data-driven Bayesian Deep Reinforcement Learning algorithm is proposed to provide the energy management and resilient control in a multi-energy micro-grid system. To summarize, this study makes the following original contributions:

1) A Bayesian DRL-based real-time decision making scheme has been proposed, which is designed to deal with the resilient operation of multi-energy micro-grid system. During the resilience mode, the target is to keep only essential loads served. During normal condition, the proposed RL approach is able to help the TSO to achieve a near optimal real-time control with minimum system operation costs.

2) Bayesian Deep Learning theory and Reinforcement Learning are integrated to generate real-time system control strategy with the aim of capturing uncertainties and avoiding value function estimation during the training process in the multi-energy micro-grid system. To the best of the authors' knowledge, this is the first attemption to exploit Bayesian Deep Reinforcement Learning in the area of system resilient control.

3) A series of comprehensive case studies are conducted, which consider the uncertainties in the extreme events. For example, the extreme events are assumed to have longer duration period and their own time-changing profiles. Compared with the state-of-the-art methods, the superior performance and the robustness of the proposed approach are studied and analysed through various operating scenarios.

The rest of this part of work is organized as follows. Section 5.2 reviews the dropout-based Bayesian Deep Learning method and the proposed Bayesian Deep Deterministic Policy Gradient(BDDPG) is introduced. Section 5.3 illustrates the experiment setting including the description of the test network and data, the details of the formulation of the reinforcement learning and other related parameter setting. Comprehensive numerical experiments are then conducted in order to demonstrate the superior performance of the proposed method. Finally, section 5.4 draws the conclusions.

5.2 The proposed Bayesian Deep Deterministic Policy Gradient real-time control scheme

5.2.1 Problem Setting

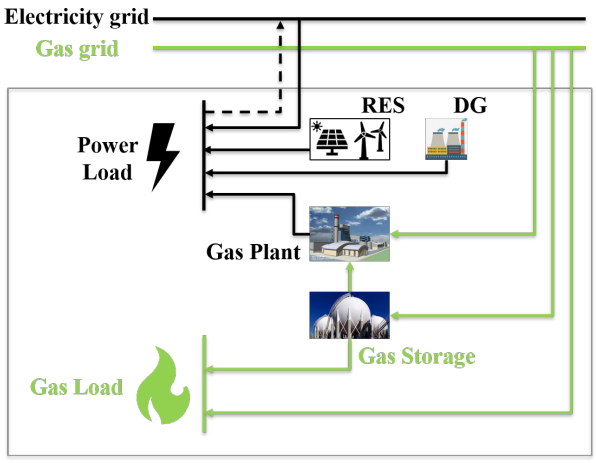


Figure 5.1: The illustrative structure of the studied multi-energy micro-grid

We focus on a multi-energy micro-grid, which consists of transmission lines, distributed generators(DG), RES, gas plants, gas storage, gas well(i.e gas grid), power load, and gas load, as depicted in Fig. 5.1. In addition, we assume that the connection lines between power system and gas system are designed so that the gas flow is flexible enough to supply the gas loads, gas generators, gas storage unit simultaneously.

5.2.2 Markov Decision Process

To tackle the aforementioned challenges, we propose to integrate the Bayesian Deep Learning theory with the current RL technique. Before illustrating the methodology, we would like to firstly give background introduction of RL and try to reformulate the proposed problem as a Markov Decision Process (MDP).

In the context of RL, an agent takes actions sequentially to interact with the environment following a pre-defined rule, which is designed to maximize the cumulative reward (or minimize the pre-designed cumulative cost). In general, RL is described as a MDP which includes: 1) a state space S; 2) an action space A; 3) a state transition probability $p(s_{t+1}|s_t, a_t)$, which satisfies the Markov property, i.e., $p(s_{t+1}|s_t, a_t) = p(s_{t+1}|s_1, a_1, ..., s_t, a_t)$; 4) a reward function r: $(S, A) \rightarrow R$ and 5) a policy function $\pi(s_t) = a_t$, which is used to govern the agent when choosing the action a_t at a certain state s_t. More specifically, the components of a MDP for this problem are detailed as:

(1) ***Agent:*** RL involves an agent, the *micro-grid central controller* (MGCC) in this work, learning by itself which actions to take in an uncertain and complex environment [158].

At each time step t, MGCC obtains the local information of PV generation, power loads, gas loads from the installed sensors and measures the state-of-charge (SoC) of gas storage unit. When the agent interacting with the environment, a series of states, actions and rewards are generated: $s_1, a_1, r_1, s_2, a_2, r_2, ..., s_t, a_t, r_t, ...$ over T time steps. The cumulative return $R = \sum_{t=0}^{T} \gamma^t r_{t+1}$ is the sum of discounted reward where $\gamma \in [0, 1]$ is the discount factor and T is the experiment

horizon (e.g., 24 hours). It is used to represent how the agent is going to balance the effect from current and future states. The Q-value function $Q_\pi(s_t, a_t) = E[R|s_t, a_t, \pi]$ represents the estimation of the cumulative return given an action a_t, at state s_t, and following the policy π from the selected states on-wards. An optimal policy can be obtained from the optimal Q value $Q^*(s_t, a_t) = \max Q_\pi(s_t, a_t)$ by selecting the action that contributes the highest Q-value at each state. Afterwards, the determined optimal control policy will be implemented by the MGCC upon the controllable units (e.g., traditional generator, gas well, and gas storage), aiming at minimizing the load shedding cost under the constrains of AC-OPF. This is then returned to the agent and is used to update the control action at next time step. In order to obtain the maximum cumulative reward (i.e. optimal policy), the agent must learn this mapping by continuously interacting with the environment.

(2) ***State:*** The state vector s_t works as providing a feedback signal towards the RL agent, which represents the results after implementing action a_{t-1} at time step $t-1$ on the micro-grid environment. In this work, the state vector is defined as:

$$s_t = [t, PL_t, GL_t, PV_t, GS_t] \in \mathcal{S} \tag{5.1}$$

where t denotes the time step in the training period, PL_t represents the electricity load of the micro-grid system, GL_t represents gas load of the system, and PV_t represents the PV generation. These features are independent to the agent's actions. On the other hand, GS_t denotes the energy contents of the gas storage unit, which can be affected by agent's actions.

(3) ***Action:*** Given the state s_t, the corresponding actions a_t of the RL agent at time step t is defined as:

$$a_t = [a_t^{GW}, a_t^{DG}, a_t^{GS}] \in \mathcal{A} \tag{5.2}$$

where $a_t^{GW} \in [0, 1]$ represents the output percentage of the gas well. $a_t^{DG} \in [0, 1]$ represents the output percentage of the distributed generator. $a_t^{GS} \in [-1, 1]$ represents the percentage of

charging (positive) and discharging (negative) gas of the gas storage unit.

(4) ***Environment and State Transition:*** When action a_t is determined, it interacts with the micro-grid environment in order to obtain the next state s_{t+1} and the reward r_t. This interacting process can be denoted as a mapping function $[s_{t+1}, r_t] = f_{MG}(s_t, a_t)$, which is constrained and characterized by the operational rules and corresponding physical rules so that the feasibility of the selected actions at each time step can be ensured.

In this research, the constrains within the transition process is mainly about the energy contents of the storage/generation units so that both gas demand and electricity demand can be supplied in both normal mode and resilience mode. As illustrated in previous sections, the state transition from s_t to s_{t+1} is not only affected by the action selection policy but also by the uncertainties in state features. Hence, in this study, the values of PL_{t+1}, GL_{t+1} and PV_{t+1} are assumed to be taken directly from the forecasting scheme, where we assume the forecasting has a certain level of error, represented by a Gaussian noise. On the other hand, the values of GS_{t+1} will be determined by the actions selected at time step t. The details of the environment f_{MG} is illustrated as follows:

$$G_t^{GW} = a_t^{GW} \times \overline{G}^{GW}, \quad \forall\, t \tag{5.3}$$

$$P_t^{DG} = a_t^{DG} \times \overline{P}^{DG}, \quad \forall\, t \tag{5.4}$$

$$GSC_t = \begin{cases} a_t^{GS} \times \overline{GS} & \text{if } a_t^{GS} \geq 0 \\ 0 & \text{else} \end{cases}, \quad \forall\, t \tag{5.5}$$

$$GSD_t = \begin{cases} a_t^{GS} \times \overline{GS} & \text{if } a_t^{GS} < 0 \\ 0 & \text{else} \end{cases}, \quad \forall\, t \tag{5.6}$$

$$GS_{t+1} = GS_t + GSC_t + GSD_t, \quad \forall\, t \tag{5.7}$$

$$P_t^{GG} = \max\big(\frac{1}{\eta^{GP}}(GSD_t + G_t^{GW} - GL_t - GSC_t), 0\big) \tag{5.8}$$

$$P_t^{GG} = \min\big(P_t^{GG}, \overline{P}_t^{GG}\big) \tag{5.9}$$

where equations (5.3) and (5.4) represent the gas output of gas well and the power output of distributed generator, respectively. Equations (5.5) and (5.6) indicate the mutually charging and discharging gas quantities GSC_t, GSD_t of gas storage unit, and the energy dynamics in (5.7) is calculated given GSC_t, GSD_t. Finally, the system still needs to purchase a certain level of gas quantity from the gas grid, converting into the power quantity P_t^{GG} with a certain conversion ratio η^{GP}, as presented in (5.8).

(5) ***Reward:*** At the end of time step t, agent obtains its reward r_t. The objective of the agent is to optimally manage the energy schedules of each controllable component to minimize the system energy cost and to avoid load shedding when an extreme event occurs. Thus, the reward function can be designed as the following aspects: 1) the penalty for violating power balance r_t^{bal}, where δ represents a tolerable slack margin that ensures the stability of the system frequency and ξ works as a bias constant that helps balancing the ratio of reward and penalty during the training process; 2) the negative cost for providing fuels to the generators and gas wells as well as the operating cost for gas storage unit r_t^{cost}; 3) the penalty when system fails to support critical electric load during extreme event period r_t^{ctl}, where PNS_t represents the proportion of power load not served at time step t and c^{PL} indicates the cost with the unit of \$/MW; 4) the reward when gas load is properly served and the penalty when it fails to support the gas load, represented as r_t^{gas}, where c^{GL} indicates the cost with the unit of $\$/Sm^3$; 5) the penalty when gas storage unit violates its physical capacity limit r_t^{gs}; 6) the reward when power load is

properly balanced r_t^{pow}

$$\begin{aligned} r_t^{bal} = & -|\xi_1 + P_t^{GG} + P_t^{DG} + PV_t - PL_t - \delta|, \\ & \text{if } P_t^{GG} + P_t^{DG} + PV_t > PL_t + \delta \end{aligned} \tag{5.10}$$

$$\begin{aligned} r_t^{cost} = & -b^{GS} \times GS_t - b^{GW} \times G_t^{GW} \\ & -(a^{DG} + b^{DG} \times P_t^{DG} + c^{DG} \times P_t^{DG} \times P_t^{DG}) \end{aligned} \tag{5.11}$$

$$r_t^{ctl} = -c^{PL} \times PNS_t \times PL_t \tag{5.12}$$

$$r_t^{gas} = \begin{cases} 20 & \text{if } GP_t - GL_t >= 0 \\ -c^{GL} \times GNS_t \times GL_t & \text{else} \end{cases} \tag{5.13}$$

$$r_t^{gs} = -100, \quad \text{if } GS_t > \overline{GS} \text{ or } GS_t < 0 \tag{5.14}$$

$$r_t^{pow} = 20, \quad \forall\ PL_t - \delta <= P_t <= PL_t + \delta \tag{5.15}$$

Finally, the overall reward function can be written as:

$$\begin{aligned} r_t = & r_t^{pow} + r_t^{gas} + \kappa^1 \times r_t^{cost} + \\ & \kappa^2 \times r_t^{ctl} + \kappa^3 \times r_t^{gs} + \kappa^4 \times r_t^{bal} \end{aligned} \tag{5.16}$$

where κ works as a bias constant that helps balancing the ratio of each reward component and penalty component during the training process. This reward function equation 5.16 indicates the direction that RL agent is trained towards, i.e. positive value for 'what to do' and negative value for 'what not to do'. In the test, the values of the reward have no physical meaning since they are normalised during the training process but the selection of the values follows the

principle that the reward should has similar level of magnitude compared to the total penalty and cost.

5.2.3 Bayesian Deep Deterministic Policy Gradient (BDDPG)

Policy gradient based approaches have shown great success in solving continuous action-state control tasks. However, with the growing stochasticity of such problems, the estimation of the deterministic value function becomes difficult. Due to the superior performance of Bayesian deep learning in probabilistic modelling, it is reasonable to investigate the feasibility of integrating Bayesian deep learning theory with RL.

In previous existing works such as [116] and [157], it has been illustrated that a deterministic neural network can be transformed into an equivalent Bayesian neural network by applying stochastic regularisation techniques in the model, such as dropout. To avoid redundancy, only key parts are kept and re-introduced here:

Dropout Approximated Inference

To combine the *epistemic uncertainty* and the *aleatoric uncertainty* in a single model, a distribution is firstly placed over the weights and the bias. The structure of a deep neural network also needs to be slightly changed, where the top layers are splited into two parts in order to simultaneously generate $\hat{y}$ and $\hat{\sigma}^2$:

$$[\hat{y}, \hat{\sigma}^2] = f_{BNN}^{\hat{W}}(x) \tag{5.17}$$

where f_{BNN} represents the Bayesian deep neural network parameterized by $\hat{W} \sim q_\theta(W)$. Given that a normal likelihood is chosen to model the aleatoric uncertainty, the final loss function of the Bayesian neural network can be formulated as:

$$\mathcal{L}_{BNN}(\theta) = \frac{1}{T_{train}} \sum_{i=1}^{T_{train}} \frac{1}{2\hat{\sigma}_i^2} ||y_i - \hat{y}_i||^2 + \frac{1}{2} \log \hat{\sigma}_i^2 \tag{5.18}$$

Note that the loss function can consider both the model uncertainty through $\hat{y}$ and the aleatoric uncertainty through $\hat{\sigma}$.

Dropout variational inference is a good example of inference approximation, which has great practical value of being simple to be implemented. However, there are also limitations such as underestimating the model uncertainty[27]. This is due to the unbalanced penalty setting when minimising the KL divergence between the approximating posterior $q(w)$ and true posterior $p(w|X, Y)$. For instance, penalty only occurs when $q(w)$ has probability mass where $p(w|X, Y)$ has no mass, but there is no penalty for $q(w)$ when not placing probability mass at locations where $p(w|X, Y)$ does have mass [159]. To avoid this underestimation, α-divergences could be regarded as an alternative divergences to the KL objective of variational inference.

Dropout Approximated α-Divergence Inference

The distribution to be approximated usually has a general form shown as below:

$$p(w) = \frac{1}{Z} p_0(w) \prod_n f_n(w) \tag{5.19}$$

where in the contexts of Bayesian neural networks, $f_n(w)$ represent the likelihood terms $p(y_n|x_n, w)$, $Z = p(Y|X)$, and the target of approximation is the posterior $p(w|X, Y)$ [159]. Several approximating inference techniques such as variational inference (VI) or expectation propagation (EP) are widely used and have already shown their effectiveness. But these two techniques are special cases of power-EP with α-divergences as its minimisation objective[160].

Various accepted definitions of α-divergences could be found in the existing literature. It is notable that these forms can be converted to each other, thus in this work, Amari's definition is used and expressed as follows[161]:

$$D_\alpha[p||q] = \frac{1}{\alpha(1-\alpha)}(1 - \int p(w)^\alpha q(w)^{1-\alpha} dw) \tag{5.20}$$

Due to the fact that power-EP might have issues with space complexity when facing larger size data, since the approximating factors are attached to every likelihood term $f_n(w)$, researchers have proposed alternative inference methods such as BB-α in [162], with the aim of reducing space complexity[159].

A general form of BB-α energy function given $\alpha \neq 0$ is firstly shown as below[163]:

$$\mathcal{L}_\alpha(q) = -\frac{1}{\alpha}\sum_n \log E_q[(\frac{f_n(w)p_0(w)^{\frac{1}{N}}}{q(w)^{\frac{1}{N}}})^\alpha] \tag{5.21}$$

An MC approximation form is given in equation 5.22 since the expectation might be intractable:

$$\mathcal{L}_\alpha^{MC}(q) = -\frac{1}{\alpha}\sum_n \log \frac{1}{K}\sum_k [(\frac{f_n(\hat{w}_k)p_0(\hat{w}_k)^{\frac{1}{N}}}{q(\hat{w}_k)^{\frac{1}{N}}})^\alpha] \tag{5.22}$$

where $\hat{w}_k \sim q(w)$. Although this MC approximation is a biased approximation since the expectation $E_q[\dots]$ in equation 5.22 is computed before logarithm, the work in [164] has shown that the bias can be overcome properly by the variance of the samples, resulting in negligible effect. When $\alpha \to 0$ it becomes the variational free energy (the VI objective), which is expressed in equation 5.23[159].

$$\mathcal{L}_0(q) = \mathcal{L}_{VFE}(q) = KL[q||p_0] - \sum_n E_q[\log f_n(w)] \tag{5.23}$$

This time, the corresponding MC approximation $\mathcal{L}_{VFE}^{MC}$ can be regarded as an unbiased approximation of $\mathcal{L}_{VFE}$. In [164], the authors directly evaluates the MC estimation equation 5.22 with samples $\hat{w}_k \sim q(w)$. However, when applied with dropout approximation, there could be prohibitive cost if the network has more complicated structure. This is because sampling $\hat{w}$

from dropout masked $q(w)$ are implemented to each data point, which will then be formed into a mini-batch. On the other hand, the minimisation of the variational free energy ($\alpha = 0$) with the dropout approximation can be computed very efficiently. As discussed before, the main computational burden comes from the sampling process of $q(w)$, which means if the additional part $KL[q||p_0]$ can be approximated properly, there will be no need to evaluate $q(w)$, thus the whole minimisation process could be relieved[159]. Hence, the next step is to give an improved form of $BB-\alpha$ energy to allow applications with dropout.

First, rewrite the approximate posterior $q(w)$ with respect to $\tilde{q}(w)$, which is a free-form cavity distribution[159]:

$$q(w) = \frac{1}{Z_q}\tilde{q}(w)\left(\frac{\tilde{q}(w)}{p_0(w)}\right)^{\frac{\alpha}{N-\alpha}} \tag{5.24}$$

In order to ensure $q(w)$ a valid distribution, $Z_q < +\infty$ is assumed as a normalising constant[159]. When $\alpha/N \to 0$ (e.g. when $\alpha \to 0$, $N \to +\infty$ or when α increases sub-linearly to N), the un-normalised density term in equation 5.24 converges to $\tilde{q}(w)$ for every w, and $Z_q \to 1$ by the assumption of $Z_q < +\infty$[165]. Hence $q \to \tilde{q}$ when $\alpha/N \to 0$. Next, rewrite the $BB-\alpha$ energy in terms of $\tilde{q}$[159]:

$$\begin{aligned}
\mathcal{L}_\alpha(q) &= -\frac{1}{\alpha}\sum_n \log \int \left(\frac{1}{Z_q}\tilde{q}(w)\left(\frac{\tilde{q}(w)}{p_0(w)}\right)^{\frac{\alpha}{N-\alpha}}\right)^{1-\frac{\alpha}{N}} p_0(w)^{\frac{\alpha}{N}} f_n(w)^\alpha dw \\
&= \frac{N}{\alpha}(1-\frac{\alpha}{N})\log \int \tilde{q}(w)\left(\frac{\tilde{q}(w)}{p_0(w)}\right)^{\frac{\alpha}{N-\alpha}} dw - \frac{1}{\alpha}\sum_n \log E_{\tilde{q}}[f_n(w)^\alpha] \\
&= R_\beta[\tilde{q}||p_0] - \frac{1}{\alpha}\sum_n \log E_{\tilde{q}}[f_n(w)^\alpha], \quad \beta = \frac{N}{N-\alpha}
\end{aligned} \tag{5.25}$$

where $R_\beta[\tilde{q}||p_0]$ represents the Renyi divergence of order β and $R_\beta[\tilde{q}||p_0] \to KL[\tilde{q}||p_0] = KL[q||p_0]$[166]. This means that the BB-α energy can be further approximated as expressed in equation 5.26, if a constant α scales sub-linearly with N[159]:

$$\mathcal{L}_\alpha(q) \approx \mathcal{L}_\alpha(\tilde{q}) = KL[q||p_0] - \frac{1}{\alpha}\sum_n \log E_q[f_n(w)^\alpha] \tag{5.26}$$

Given a loss function $\mathcal{L}(\dots)$ (e.g. l_2 loss in regression or cross entropy in classification), the likelihood term $f_n(w)$ can be found to be $f_n(w) \propto p(y_n|x_n, w) \propto exp[-l(y_n, f^w(x_n))]$[167]. Hence, replacing $f_n(w)$ with the loss function and using Monte Carlo sampling as a unbiased approximation of the expectation over $q(w)$, the following minimisation objective can be obtained and shown as below[159]:

$$\tilde{\mathcal{L}}_\alpha^{MC}(q) = KL[q||p_0] + C - \frac{1}{\alpha}\sum_n LogSumExp[-\alpha l(y_n, f^w(x_n))] \tag{5.27}$$

with Log-Sum-Exp being over K stochastic samples from $\hat{w}_k \sim q(w)$. Comparing with the original BB-α energy function formulation equation 5.21, the approximated form in equation 5.27 has less computational burden due to the negligence of the posterior evaluation. In addition, the overall form is similar to standard objective functions in deep learning, which means practically easier to implement.

Specifically, in regression problems, the loss function is defined as $\mathcal{L}(y, f_w(x)) = \frac{1}{\tau}||y - f^w(x_n)||_2^2$ and the likelihood term can be interpreted as $y \sim N(y; f^w(x_n), \tau^{-1}I)$[159]. Re-formulating the energy function in equation 5.27 returns the final objective[159]:

$$\tilde{\mathcal{L}}_\alpha^{MC}(q) = -\frac{1}{\alpha}\sum_n logSumExp[-\frac{\alpha\tau}{2}||y_n - f^{\hat{w}_k}(x_n)||_2^2] + \frac{ND}{2}\log\tau + \sum_i p_i||M||_2^2 \tag{5.28}$$

where τ is the precision of the model, $\hat{w}_k$ are the stochastic samples from dropout masked weights, $f^{\hat{w}_k}(x_n)_{k=1}^K$ are a set of K stochastic forward passes through the neural network, D and p_i represents the drop probability and keep probability of dropout rate of the i_{th} layer respectively, N represents the size of the mini-batch and M are the neural network weights

without dropout[159][168].

In terms of the practical implementation with DDPG, due to the fundamental theory that one state has one deterministic action, only the critic network can be transformed into a Bayesian network. Hence, it is implemented by replacing the critic network loss function with equation 5.28, doing the stochastic feed-forward process through the dropout mask multiple times, generating and collecting the corresponding outputs, calculating the relevant gradients through new loss function and updating the target network parameters following the rules introduced in previous sections.

5.3 Case Study

In this section, three case studies have been designed aiming at investigating the superior performance of the proposed BDDPG approach in multi-energy micro-grid network. We start by modelling the normal, steady state network operation, which is a classic energy management problem. Resilient operation is studied next, where the micro-grid system is connected to the main grid during normal condition in case study II while in case study III, micro-grid operates in island mode.

5.3.1 Data Description

The topology of examined multi-energy micro-grid is shown in Fig. 5.3, which is modified from[169]. It is notable that here, the system is simplified properly compared to the original structure since we only focus on the electrical sector. For instance, the network has 6 electrical nodes (buses), 6 gas nodes, 1 gas-supplied generator (G_3), 1 traditional generator unit (G_1), 2 gas wells (GW_1, GW_2), 1 gas storage unit (GS), 1 PV unit (G_2), 3 power loads (PL_1, PL_2, PL_3) and 3 gas loads (GL_1, GL_2, GL_3). The parameters of the network units including generator capacities, line capacities, line impedance, gas/power conversion ratio etc. are from the supplementary materials of [169]. The average daily power and gas loads profiles are shown in the right top

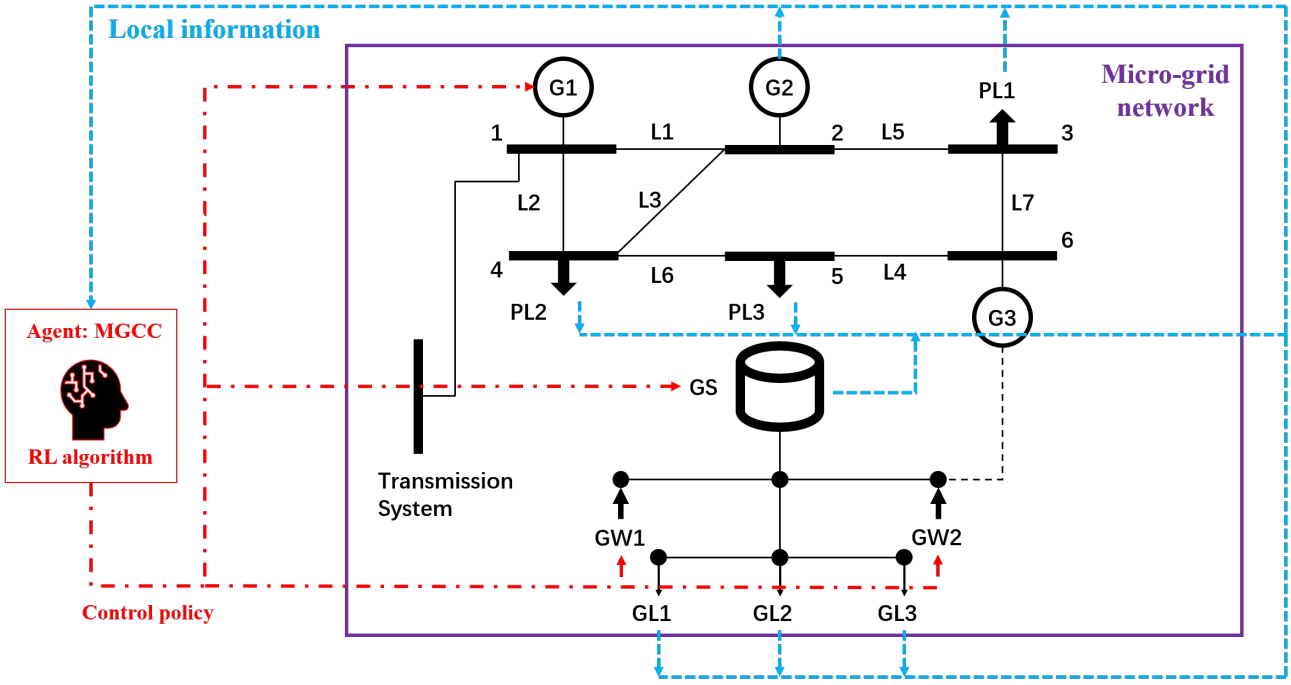


Figure 5.2: The studied power-gas network with corresponding load profiles

figure in Fig. 5.3. In order to explore the system operation under emergency situation, PV generation profile and critical demand profile are included and demonstrated in the right bottom figure in Fig. 5.3, which is from [139]. Table 5.1 has provided the summary of the detailed parameter setting.

5.3.2 Experiment Setup

To demonstrate the superior performance of the proposed approach, we compare our Bayesian DDPG with the current state-of-the-art DDPG method. It is notable here that since our task has multi-dimensional continuous action space, it is meaningless to consider the traditional DQN, DPG methodologies due to their limitations as illustrated in previous chapters. We would like to refer the readers to other high level works such as [24][149][150], where DDPG has been proved to have better performance than other approaches.

We set up four DNNs for the online actor, target actor, online critic, target critic respectively. All DNNs have three hidden layers with 64-128-64 neurons and Relu activation function at each

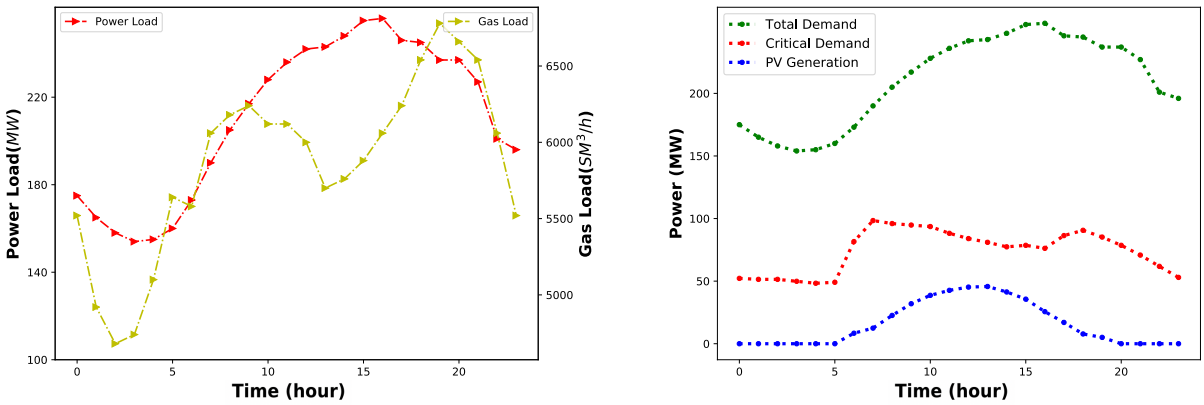


Figure 5.3: Power and Gas Demand

Table 5.1: Hyper-parameters of the studied multi-energy micro-grid network

Parameter	Value
$\overline{G}^{GW}$	10000 Sm^3/h, 12000 Sm^3/h
$\overline{P}^{DG}$	220 MW
$\overline{GS}$	15000 Sm^3
η^{GP}	500 Sm^3/MW
$\overline{P}_t^{GG}$	50 MW
c^{PL}	1000 \$/MW, 900 \$/MW, 800 \$/MW
c^{GL}	3.6 \$/Sm^3, 3.8 \$/Sm^3, 4 \$/Sm^3
ξ_1	100
δ	5 MW
b^{GS}	0.0342 \$/Sm^3
b^{GW}	0.04 \$/Sm^3, 0.036 \$/Sm^3
a^{DG}	176 \$/h, 137 \$/h
b^{DG}	13.5 \$/MWh, 17.7 \$/MWh
c^{DG}	0.0004 \$/MWh^2, 0.005 \$/MWh^2

layer respectively. The Adam optimizer is used for training with a learning rate of $\alpha_\phi = 10^{-4}$ and $\alpha_\theta = 10^{-4}$ for the critic and actor network respectively. We use $\tau = 0.001$ as the target network's soft updating rate. For the critic, we use a discount factor of $\gamma = 0.75$. For the Bayesian model, the dropout keep probability is set to be 0.95. α-divergence and model precision τ_v are set to be 0.98 and 12 respectively. We train the networks with a mini-batch size $N = 128$ and for $M_{train} = 2000$ episodes with 72 time steps in each episode. The proposed RL approaches are implemented based on Tensorflow 2.2.0 in Python[130]. The simulations are carried out on a Macbook Pro laptop with 2.7 GHz Intel Core i7 processor and 16 GB of RAM. Table 5.2 has provided the summary of the detailed parameter setting.

Table 5.2: Hyperparameters of the Proposed BDDPG

Parameter	Value
Layer type	dense
Number of state space	13
Number of action space	6
Hidden layer structure	64-128-64
Batch size	128
Number of episode	2000
Number of samples (T_{sample})	50
Dropout rate	0.95
Optimizer	Adam
Normalisation	[0,1]normalisation
Actor learning rate	0.0001
Critic learning rate	0.0001
Discount factor	0.75
α**-divergence**	0.98
model precision	12

5.3.3 Case Study 1: Multi-Energy Micro-grid System Energy Management

In this test, we illustrate the advantages of using Bayesian RL agent as a real-time MGCC comparing with naive DDPG method. In order to prove that RL agent has the flexibility of modelling multi-dimensional continuous action space, the test is carried out under the assumption that the micro-grid is operated in island mode. The simulation in this case study is performed with hourly resolution, i.e. $T = 1, 2, ..., 24$ and the test period length is one day.

Performance of DDPG and BDDPG approach

Fig. 5.4 to 5.5 have shown the power demand balance and the individual generator contribution of each method respectively. As can be observed, both approaches are able to exploit the flexibility from multiple supply sources and maintain an accepted level of demand-supply balance. Furthermore, DDPG has a total MSE of 88.62 and it is interesting to point out that the majority of the error occurs when there is abundant PV generation. In terms of BDDPG method, significant improvement of controlling accuracy could be observed with a total MSE of

52.57, which is a 40.68% improvement.

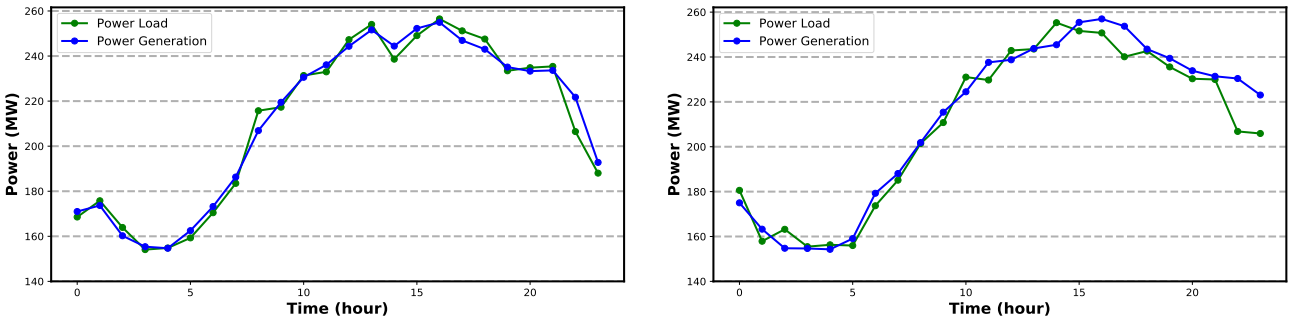


Figure 5.4: Balancing of load and generation: BDDPG (left) DDPG (right)

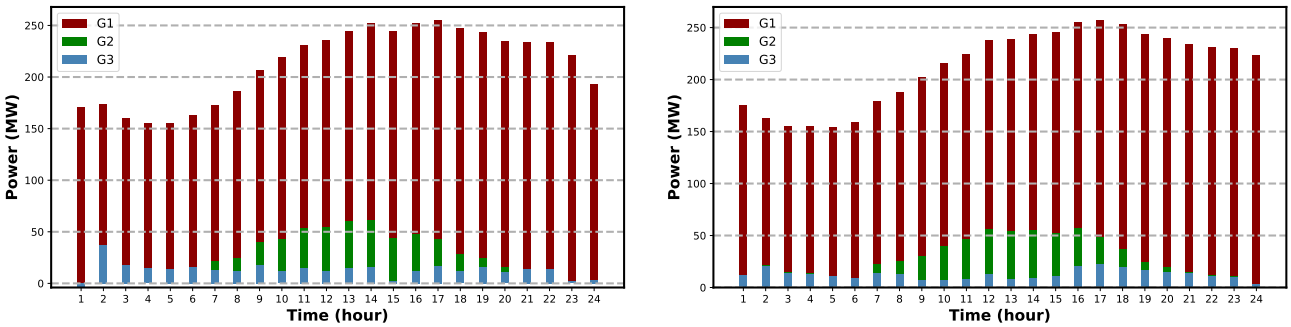


Figure 5.5: Individual schedule of generators: BDDPG (left) DDPG (right)

Comparison of two approaches

As a standard procedure in RL, the training quality of DDPG and BDDPG methods are assessed over 2000 episodes. Fig. 5.6 illustrates the average reward for the examined methods of 10 random seeds.

As illustrated in Fig. 5.6, BDDPG significantly outperforms the naive DDPG method, with the cumulative reward converged to around 200 while DDPG has a stabilised cumulative reward of around 100. The superior performance of BDDPG could be explained in the following reasons: (1) Bayesian model has the ability of capturing the massive uncertainties in PV generation, which can be supported by the fact that large amount of error occurs when PV generation is involved (as shown in Fig. 5.4); (2) DDPG optimises the parameters along the direction of maximising the Q value from critic network, thus, as previously explained, there is potential possibility that DDPG selects the greedy, short sighted, high-risky or unreasonable actions

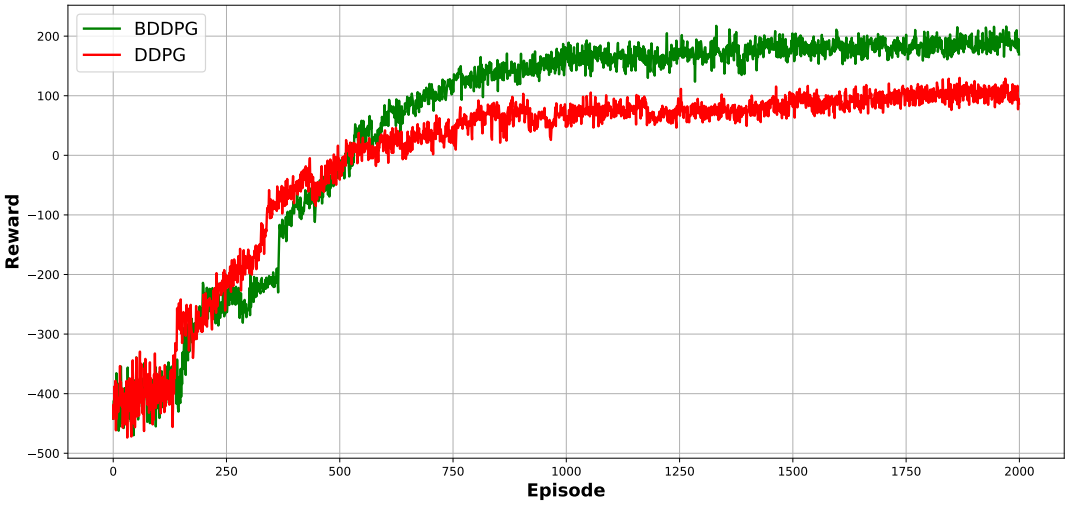


Figure 5.6: Average reward of DDPG and BDDPG over 10 random seeds.

with high Q value for the current state. In other words, due to the Q value maximising operator $Q^*(s_t, a_t) = \max Q_\pi(s_t, a_t)$, Q-learning based methods are essentially more likely to select overestimated values, resulting in an overoptimistic value estimation, which can be found a confirmed conclusion in [168] and [17]. On the other hand, by using Monte Carlo dropout sampling, the uncertainties over value functions in continuous control domains are modelled and explored to full advantage. BDDPG thus is able to offset the effect from uncertainties and tackle the overestimation issue.

As illustrated in Fig. 5.7, the area shaded with light and dark blue indicates the explored domain from Monte Carlo dropout sampling. After sampling, the red line that represents the mean of the posterior will participate in the training process. In contrast, the green line from DDPG is slower to converge and has more variance, which leads to a less stable training process. It is observed that though the uncertainty distribution of BDDPG is large during early stages of training, eventually it will converge to a reasonable and more stable level.

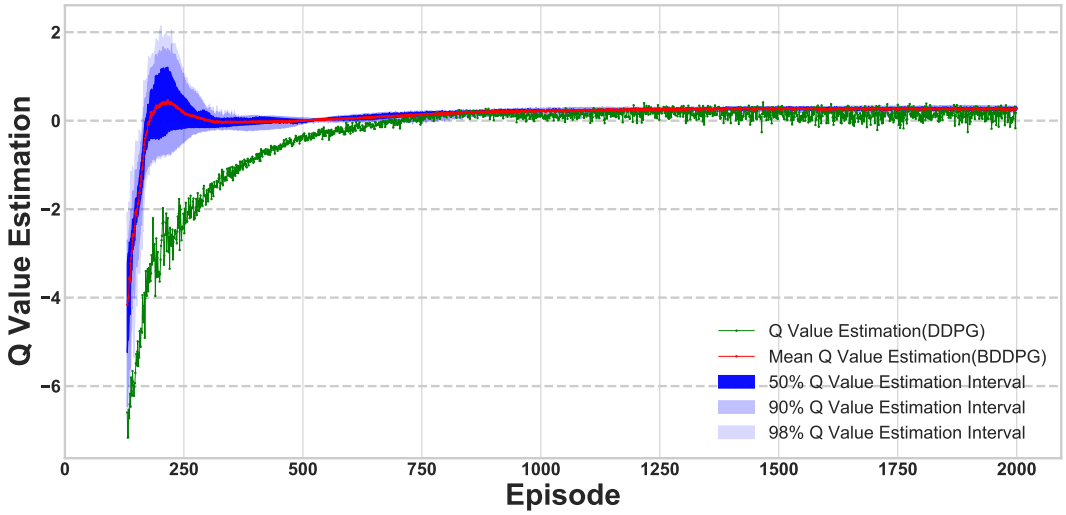


Figure 5.7: The illustrative Q-value estimation process of DDPG and BDDPG over 10 random seeds.

5.3.4 Case Study 2: Multi-Energy Micro-grid Resilient Control

In this test, we aim to evaluate the performance of the proposed BDDPG method and naive DDPG method when facing with extreme events. This experiment is carried out based on the assumption that the micro-grid is supplied from the transmission system during normal condition, then when an extreme event happens, the micro-grid is dis-connected from the transmission system thus operating in island mode. Due to the prohibitive cost, diesel generator G_1 and gas unit G_3 can only be regarded as back up generators during resilience period. On the other hand, PV unit G_2 is assumed to be available all the time, which is an un-controllable variable. The critical demand is assumed to be a proportion of the total demand with some uncertainties. Hence the objective of this test is to provide energy balance at minimum cost during normal mode and support only critical demand during resilience mode. In order to show a completed normal-resilience-normal cycle, the test period is set to be 3 days, i.e. $T = 1, 2, ..., 72$, thus the extreme events are assumed to last for 24 hours on day 2.

Performance of DDPG and BDDPG approach

Fig. 5.8 and 5.9 have shown the energy balance and the behaviour of each generator of each method respectively. For instance, during the periods hour 1-24 and 48-72, when the whole network is in normal condition, micro-grid electricity demand is supplied by the transmission system and PV generation. Gas demand is supplied only by the gas well. The gas storage device tends to obtain more energy content during the normal period, which will eventually contribute to the back-up capacity when an extreme event happens. During the event period hour 24-48, the micro-grid switches to island mode, thus generators G_1 and G_3 start operating. The diesel generator G_1 uses its own back-up fuel supply, thus can be treated as an independent unit. On the other hand, gas generator G_3 uses gas supply from gas storage device GS and gas wells GW_1 and GW_2, which means the gas demand has the priority to be met over the gas generator since any failure of gas service would cause a large amount of penalty. In addition, it is easily observed that large errors can always be found during the switching phase, e.g. hour 24, 48. This is because of the uncertainties of the extreme events such as the length of the duration, the exact happening and ending hour etc. It is important to point out that, in this test, the events period setting is realised by adding an extra dimension of binary variables into the training feature, which means that in the practical real-time implementation, the agent can give policy decisions according to different event scenarios. In summary, the proposed BDDPG can provide cost-saving operation strategies during normal condition and is also able to adaptively adjust to the resilience condition through capturing the uncertainties during the training stage. In addition, the proposed method also learns i) the priority of each sector in the network in different operating conditions and ii) the flexibility of gas storage device in order to provide emergency supply.

Comparison of two approaches and with optimisation approach

To demonstrate the superior performance of the proposed method, once again MSE is calculated as evaluation. From the calculation, BDDPG has a total MSE of 91.07, which is 34.87% lower

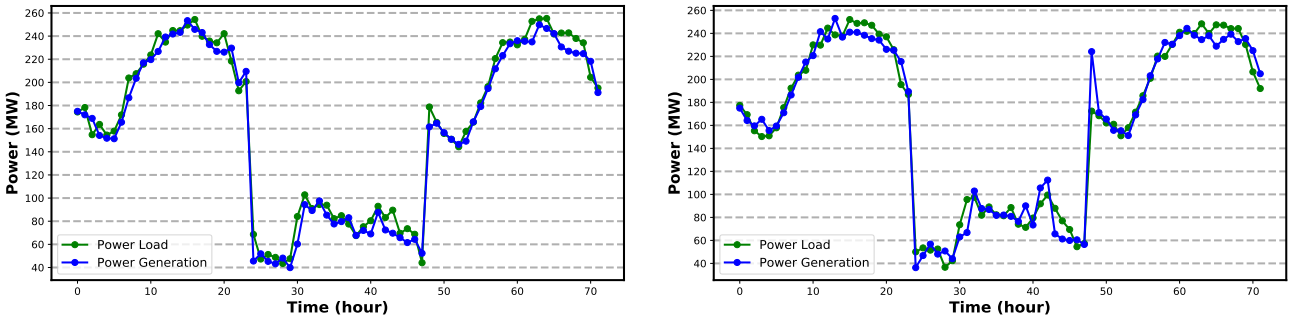


Figure 5.8: Balancing of load and generation: BDDPG(left) DDPG(right)

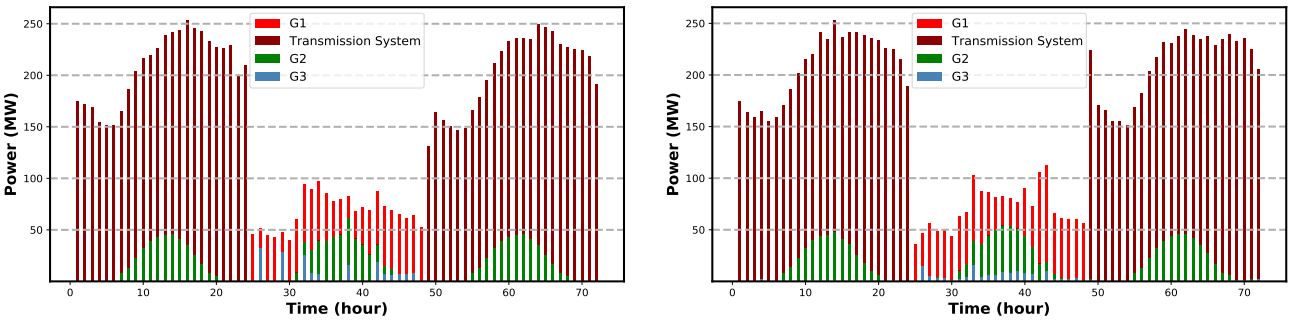


Figure 5.9: Individual schedule of generators: BDDPG(left) DDPG(right)

than 139.83 of naive DDPG. However, it is notable that during resilience mode, the cost function is asymmetric since any losses of critical load will cause a significant penalty, but there is only extra fuel cost for over-supply instead of penalty. Hence the cumulative reward is employed to provide a more comprehensive evaluation, which considers the asymmetric nature of the loss function. Similar as in previous case study, 10 different random seeds are generated and for each seed the RL agent is trained for 2000 episodes, where each episode consists of 72 time steps (i.e. 3 days). Fig. 5.10 has illustrated average cumulative reward of DDPG and BDDPG over 10 random seeds. For instance, in the first hundreds of episodes large penalties can be observed. This is the stage when tuples of actions and states are filling into the memory buffer and the model starts training only after the memory buffer is full. When the agent starts learning, BDDPG learns its control policy in a faster and more stable fashion and eventually BDDPG manages to converge to a higher optimal value than naive DDPG. To prove this, the last 400 episodes' average cumulative reward (i.e. zoomed in area in Fig. 5.10) is calculated, which could be considered as stabilised results. BDDPG has an average of 1235.25, which is 9.08% higher than DDPG (1132.39).

As a recent concept of off-line training approach, people always ask that why using such a method instead of traditional optimisation methods. As discussed in previous sections, one of the most important advantages of RL is that it is able to provide real-time decision-making in an uncertain environment, which is the drawback of most of the optimisation approaches. To prove this, stochastic programming (SP), which is one of the widely employed optimisation methods, is used as comparison[170]. In [170], SP is implemented based on a rolling scheduling procedure, which is solved on an hourly basis. For instance, at the first hour t_0, deterministic operational decisions are directly calculated from the available real-time data indexed at t_0 Then, from $t_0 + 1$ to the end of the scheduling horizon, various scenarios are generated at each time step and the scenario-dependent operational decisions are optimized in order to represent the uncertainties in each scenario[170]. Hence the objective of each stochastic optimization is to minimize both the cost at t_0 and the expected cost of all scenarios from $t_0 + 1$ to the end of the scheduling horizon, which includes operational cost at normal operating condition plus the potential penalties during the resilience mode.

Table 5.3 has shown the comparison among the discussed methods in terms of total cost and computational time. It is notable that different from optimisation methods, RL methods train the model following the trial-and-error rule, thus the cumulative reward shown in Fig. 5.10 includes the part of corresponding penalties and needs to be removed. For instance, BDDPG has slightly higher computation time (7s VS. 3s) due to the Monte Carlo sampling process but achieves 5.31% lower total cost than the naive DDPG method. On the other hand, despite the highest accuracy, the computational time for SP is around 4292 seconds since it considers 100 stochastic scenarios for a rolling horizon of 72 time steps. It can be found that both RL methods can provide real-time decision making while SP optimisation is suffering from high computational burden thus cannot guarantee hourly (or higher) resolution real-time decision making.

Table 5.3: Case Study 2: Comparison of RL methods with SP

Method	**DDPG**	**BDDPG**	**SP**
Total Cost	$138,387.89$\$	$131,405.93$\$	$129,500.43$\$
Computation Time	$< 3s$	$< 7s$	$4292s$

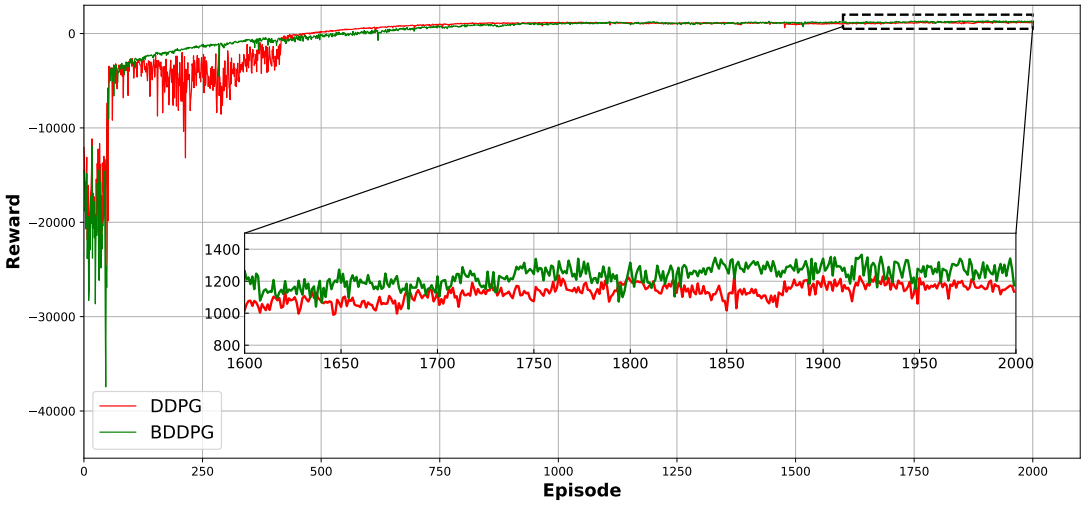


Figure 5.10: Average reward of DDPG and BDDPG over 10 random seeds.

5.3.5 Case Study 3: Island Multi-Energy Micro-grid System Resilient Control

In this test, we further investigate the extreme situation when the micro-grid is operated in island mode during normal days, then when an extreme event happens, the micro-grid losses its PV generation. In other words, case study 2 considers the situation when the extreme events happen outside the micro-grid thus has no influence on the micro-grid itself, while the effect of extreme events inside micro-grid is considered in this case study. In addition, due to the fact that the maximum capacity of PV unit is less than the electricity demand, we assume that the micro-grid is also supplied by G_1 during normal condition. Similar as before, due to the prohibitive cost, gas unit G_3 can only be regarded as back up generators during extreme events period. Furthermore, G_1 is assumed to have only partial generation capacity during the resilience mode (i.e. 30%) as a reasonable response to the extreme events. Hence in this study, the difficulty lies in making efficient use of the limited generation resources so that both cost-saving normal operation and safety-oriented resilient operation can be satisfied. In order to show a completed normal-resilience-normal operation cycle, the test period is set to be 3 days, i.e. $T = 1, 2, ..., 72$, thus the extreme events are assumed to last for 24 hours on day 2.

Performance of DDPG and BDDPG approach

Fig. 5.11 and 5.12 have shown the energy balance and the behaviour of each generator for each method respectively. Similar as in case study 2, during hour 1-24 and 48-72, micro-grid is in normal operation. Micro-grid electricity demand is supplied by the distributed generator G_1 and PV unit, which plays as an important role of reducing the total electricity bills. When entering the extreme events hour 24-48, the micro-grid is still in island mode, but PV unit is out of service due to the extreme events. It is notable here that we assume the effect of the extreme events is on the local bus-bar area, thus node 2 in Fig. 5.2 together with line $L1, L3, L5$ are all erased from the network, thus PV generation cannot be transmitted to the micro-grid customers. G_3 starts operating when enters the resilience mode. Similar as before, the gas demand has the priority to be met over the gas generator. Hence, in this test, due to the limited generation capacity and the potential uncertainties in demand (both electricity and gas demand), it is likely that the theoretical generation capacity is lower than the total demand, which means under this circumstance there has to be partial critical loads out of service. In terms of a brief comparison of control accuracy, the two methods have total MSE of 77.47 and 122.78 respectively, where BDDPG is 36.91% lower than naive DDPG.

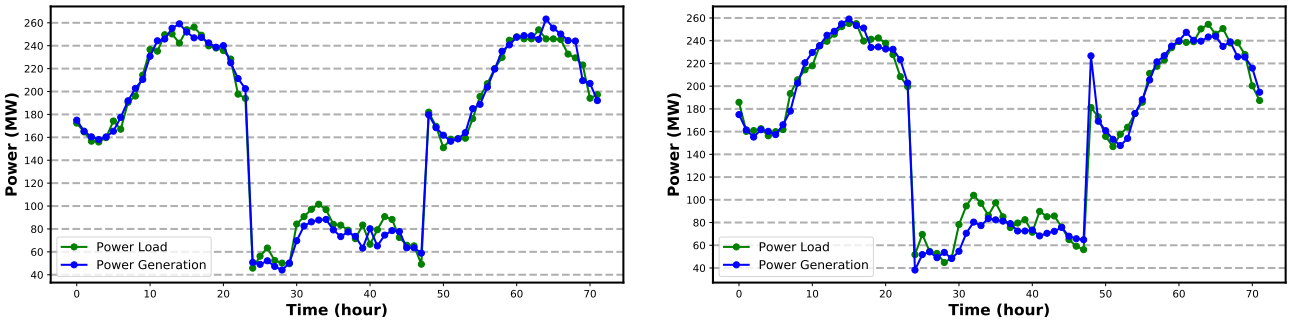


Figure 5.11: Balancing of load and generation: BDDPG (left) DDPG (right)

Comparison of two approaches and with optimisation approach

Fig. 5.13 has illustrated average cumulative reward of DDPG and BDDPG over 10 random seeds for 2000 episodes. For instance, when the agent starts learning, BDDPG is able to learn from

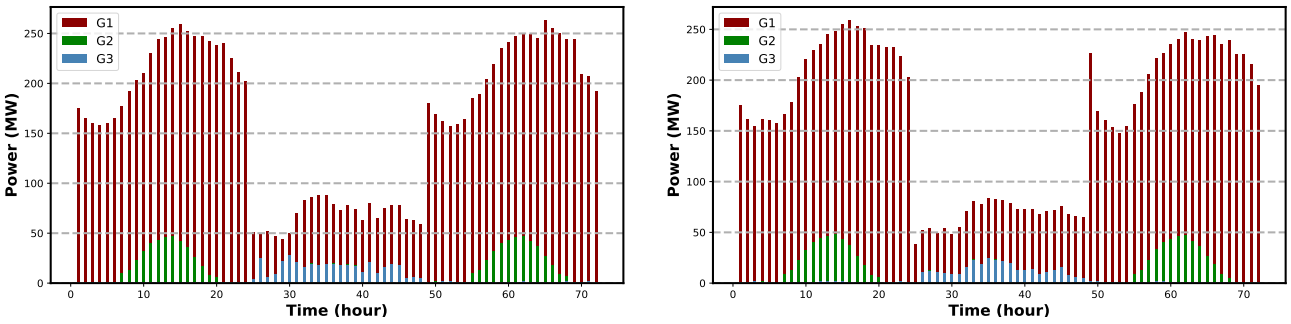


Figure 5.12: Individual schedule of generators: BDDPG (left) DDPG (right)

the generated experience with less disturbance, which implies that BDDPG tends to interact with the environment in a more reasonable fashion and eventually BDDPG could manage to converge to a higher optimal value than naive DDPG. The last 400 episodes' average cumulative reward (i.e. zoomed in area in Fig. 5.13) is calculated, which could be regarded as stabilised results. BDDPG (green line in Fig. 5.13) has an average of 1245.48, which is 27.04% higher than 980.39 of DDPG (red line in Fig. 5.13). Furthermore, compared with SP method in terms of total cost and computational time, BDDPG has slightly higher computation time (7s VS. 3s) due to the Monte Carlo sampling process but achieves total cost of 185,655.90$, which is 5.08% lower than 195,586.81$ from naive DDPG method. Despite the lowest total cost, SP is restricted by its high computational burden, which is around 4022 seconds. In addition, SP obtains its lower cost mainly because of the fact that it takes more greedy policy. For instance, when there is energy stored in the gas storage unit, SP tends to sell all of them due to the storage costs and starts re-charging it right before the event happens. However, as explained earlier, there are uncertainties about the exact start and duration hour of an event. Hence, an extreme event happens earlier than expected could easily cause zero back-up capacity thus compromises the whole network. More importantly, as discussed before, due to the limited generation capacity, occasionally there will be particular period when the theoretical generation capacity is less than the total demand, which could lead to different results for different methods. For instance, optimisation-based methods employ various constrains conditions such as energy balance (i.e. power generation = power consumption), operational parameters within the bounded range (i.e. transmission line capacity or bus-bar voltage less than the maximum value), which could lead to the situation that no optimal solution exists. Under this circumstance, optimisation-based

methods would fail to give any control decision, which is fatal during the extreme events period. On the other hand, since RL agents are trained by minimising objective function (maximising cumulative reward in this test), they can always provide near optimal results rather than NaN. In addition, some of the hard constrains in the system could be slightly violated during special situation for a temporary period of time. For example, there could be a minor gap between the generation and the demand and also, for a short period, power flow through a transmission line can exceed the upper limit. In other words, these kinds of operation flexibility is highly situation-dependent, thus it needs not only fast decision making response but also the controlling model to deal with various scenario uncertainties.

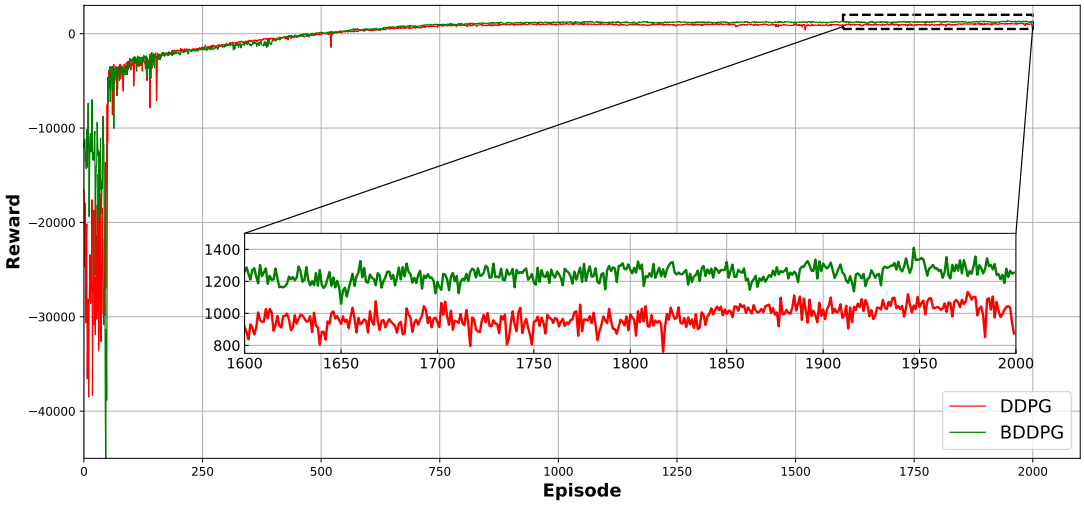


Figure 5.13: Average reward of DDPG and BDDPG over 10 random seeds.

Table 5.4: Case Study 3: Comparison of RL methods with SP

Method	DDPG	BDDPG	SP
Total Cost	$195,586.81$	$185,655.90$	$180,284.14$
Computation Time	$< 3s$	$< 7s$	$4022s$

5.4 Conclusion

This part of work proposes a novel Bayesian Deep Reinforcement Learning based real-time control scheme in order to provide energy management and resilient control in a multi-energy

micro-grid system. The proposed approach is able to capture the uncertainties in various energy sectors such as PV unit, energy storage devices etc. By using the Monte Carlo dropout sampling mean of the Bayesian value function distribution, the proposed BDDPG method could effectively avoid value function overestimation issue. The overall performance of the proposed method is analyzed and compared with naive DDPG method and another widely used optimisation method i.e. stochastic programming. Case studies have demonstrated that the proposed BDDPG method manages to learn a near-optimum policy, which has a significantly better performance of over 30% than naive DDPG method. In addition, the proposed approach can achieve approximately 5% lower total cost for various operating conditions (i.e. extreme events that occur either outside or inside the micro-grid) and can achieve high time resolution decision making. Comparing to traditional approaches, the proposed method is more stable and has shown great practical value.

Chapter 6

Thesis Summary and Future Work

6.1 Thesis Summary

This thesis focuses on a novel probabilistic uncertainty modelling technique, i.e. Bayesian Deep Learning, in order to deal with the challenges arising from the influx of uncertainties due to the large-scale penetration of RES. This chapter summarises the key findings and conclusions of this thesis and outlines the potential direction for further research.

Using Bayesian Deep Learning to Capture Uncertainty for Residential Net Load Forecasting:

This part of work proposes a novel probabilistic net load forecasting framework with the ability of capturing epistemic uncertainty and aleatoric uncertainty simultaneously. The designed framework consists of clustering-forecasting-aggregating stages, where we build deep learning models for each individual cluster and aggregate the probabilistic forecasts of each cluster at the end to obtain the final predicted net load at the aggregated level. The effectiveness and importance of different PV visibility levels are also studied considering the practical challenge of partially installed smart meters. The overall performance of the proposed method is analyzed and

compared with a series of state-of-the-art probabilistic forecasting models, which demonstrates the superior performance of the proposed BDLSTM method.

A Confidence-Aware Machine Learning Framework for Dynamic Security Assessment:

This part of work designs a confidence-aware machine learning framework for DSA based on the Conditional Bayesian Deep Auto-Encoder network. The proposed CBDAC model uses dropout to achieve Bayesian approximation with further improvement by using conditional masks. It is firstly demonstrated that the proposed Bayesian model has better performance than other state-of-the-art models. We also consider the practical situation when the system is facing with topology changes. In addition, by using the CBDAC model, epistemic uncertainty can be regarded as an indicator of the model confidence. Furthermore, we verify the feasibility of using limited data in order to update the model and thus propose a selective model updating strategy. Indicated by the model confidence, the proposed strategy significantly alleviates the time consumption under the situation of frequent system topology changes, which is of great practical value.

A Bayesian Deep Reinforcement Learning-based Resilient Control for Multi-Energy Micro-gird:

This part of work proposes a novel Bayesian Deep Reinforcement Learning-based resilient control approach for multi-energy micro-grid. In particular, the proposed approach replaces deterministic network in traditional Reinforcement Learning with Bayesian probabilistic network in order to obtain an approximation of the value function distribution, which effectively solves Q-value overestimation issue. The proposed model is able to provide both energy management during normal operating conditions and resilient control during extreme events in a multi-energy micro-grid system. Comparing with naive DDPG method and optimisation method, the effectiveness and importance of employing Bayesian Reinforcement Learning approach is

investigated and illustrated across different operating scenarios. Case studies have shown that by using the Monte Carlo posterior mean of the Bayesian value function distribution instead of a deterministic estimation, the proposed BDDPG method achieves a near-optimum policy in a more stable process, which verifies the robustness and the practicability of the proposed approach.

6.2 Future work

Future work will further exploit and develop this powerful technique, Bayesian deep learning, for more challenging tasks. For example, in terms of forecasting topics, wind power forecasting is always a challenging area due to its high variability and uncertainty. With higher level RES penetration in the future, an accurate and believable wind power forecasting model would make significant contribution to the system power flow management by providing efficient use of RES generation. In addition, in recent years, large scale energy storage schemes have attracted lots of attention and there are massive investments ongoing as a response to the zero carbon emission target by many governments in the world. Hence, in terms of the system security aspect, the cost-effective behaviour and the integrated operation of these infrastructures would fundamentally change the power system operation philosophy thus urges more advanced techniques. In terms of the technical details, with the ability to quantify both epistemic and aleatoric uncertainties, it might be helpful to further investigate using the Bayesian model as an uncertainty indicator, which could have great potential in the AI safety aspect. Furthermore, selecting an appropriate prior is still an open question for Bayesian deep learning, which will also be worth investigating in the future. Another potential direction could be how to optimize the updating strategy, according to the research findings in the second part of this thesis. Algorithms such as active sampling or incremental learning could be employed and improved. It is also believed that it is of high interest to consider the full workflow from measurements, over data processing and state estimation together.

On the other hand, RL has the following future work directions. (1): It is valuable to investigate

more complicated, real-world example networks with richer energy sources (e.g. electricity, gas, H_2, heat, etc.), various generation sectors representing the supply of electricity (e.g. wind, PV, CHP, gas storage, battery system, pumped-storage hydroelectricity etc.), and critical demand with more details (e.g. a rolling schedule of priority level). (2): Another direction could be applying multi-agent BDDPG to inter-connected micro-grids instead of a single network. In this context, coordinated control strategies for a group of micro-grids in the distribution network will take into account the aforementioned energy elements with the aim of maintaining the critical service during extreme events period and economical normal state operation of the system.

Bibliography

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[2] Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012, 2014.

[3] Elizabeth L. Ratnam, Steven R. Weller, Christopher M. Kellett, and Alan T. Murray. Residential load and rooftop pv generation: an australian distribution network dataset. *International Journal of Sustainable Energy*, 36(8):787–806, Oct. 2017.

[4] Bikash Pal and Balarko Chaudhuri. *Robust control in power systems*. New York, NY, USA: Springer-Verlag, 2005.

[5] Amanpreet Kaur, Lukas Nonnenmacher, and Carlos F.M. Coimbra. Net load forecasting for high renewable energy penetration grids. *Energy*, 114:1073 – 1084, Nov. 2016.

[6] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

[7] Louis A. Wehenkel. *Automatic Learning Techniques in Power Systems*. Kluwer Academic Publishers, USA, 1998.

[8] M. Sun, I. Konstantelos, and G. Strbac. A deep learning-based feature extraction framework for system security assessment. *IEEE Transactions on Smart Grid*, 10(5):5007–5020, Sep. 2019.

[9] Goran Strbac, Nikos Hatziargyriou, Joao Pecas Lopes, Carlos Moreira, Aris Dimeas, and Dimitrios Papadaskalopoulos. Microgrids: Enhancing the resilience of the european megagrid. *IEEE Power and Energy Magazine*, 13(3):35–43, 2015.

[10] David Newbery, Goran Strbac, and Ivan Viehoff. The benefits of integrating european electricity markets. *Energy Policy*, 94:253–263, 2016.

[11] Zhaohong Bie, Yanling Lin, Gengfeng Li, and Furong Li. Battling the extreme: A study on the power system resilience. *Proceedings of the IEEE*, 105(7):1253–1266, 2017.

[12] National Research Council et al. *The resilience of the electric power delivery system in response to terrorism and natural disasters: summary of a workshop*. National Academies Press, 2013.

[13] Kathryn Gordon and Maeve Dion. Protection of'critical infrastructure'and the role of investment policies relating to national security. *Investment Division, Directorate for Financial and Enterprise Affairs, Organisation for Economic Cooperation and Development, Paris*, 75116, 2008.

[14] Haixiang Gao, Ying Chen, Yin Xu, and Chen-Ching Liu. Resilience-oriented critical load restoration using microgrids in distribution systems. *IEEE Transactions on Smart Grid*, 7(6):2837–2848, 2016.

[15] Natalia-Maria Zografou-Barredo, Charalampos Patsios, Ilias Sarantakos, Peter Davison, Sara Louise Walker, and Philip C. Taylor. Microgrid resilience-oriented scheduling: A robust misocp model. *IEEE Transactions on Smart Grid*, 12(3):1867–1879, 2021.

[16] Mohammad Hassan Amirioun, Farrokh Aminifar, and Mohammad Shahidehpour. Resilience-promoting proactive scheduling against hurricanes in multiple energy carrier microgrids. *IEEE Transactions on Power Systems*, 34(3):2160–2168, 2019.

[17] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[18] Karl Mason and Santiago Grijalva. A review of reinforcement learning for autonomous building energy management. *Computers and Electrical Engineering*, 78:300–312, 2019.

[19] Richard Bellman. The theory of dynamic programming. Technical report, Rand corp santa monica ca, 1954.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[21] Elena Mocanu, Decebal Constantin Mocanu, Phuong H. Nguyen, Antonio Liotta, Michael E. Webber, Madeleine Gibescu, and J. G. Slootweg. On-line building energy optimization using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 10(4):3698–3708, 2019.

[22] Nathan Tsang, Collin Cao, Serena Wu, Zilin Yan, Ashkan Yousefi, Alexander Fred-Ojala, and Ikhlaq Sidhu. Autonomous household energy management using deep reinforcement learning. In *2019 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–7, 2019.

[23] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page I–387–I–395. JMLR.org, 2014.

[24] Yujian Ye, Dawei Qiu, Xiaodong Wu, Goran Strbac, and Jonathan Ward. Model-Free Real-Time Autonomous Control for A Residential Multi-Energy System Using Deep Reinforcement Learning. *IEEE Transactions on Smart Grid*, 3053(c):1–1, 2020.

[25] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[26] Zoubin Ghahramani. A history of bayesian neural networks. NIPS Workshop on Bayesian Deep Learning, 2016.

[27] Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

[28] Burr Settles. Active learning literature survey. 2009.

[29] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *ArXiv*, abs/1112.5745, 2011.

[30] Hao Wang and Dit-Yan Yeung. Towards bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3395–3408, Dec. 2016.

[31] Tao Hong and Shu Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, Jul.–Sept. 2016.

[32] T. Hong, P. Wang, and H. L. Willis. A naive multiple linear regression benchmark for short term load forecasting. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–6, Jul. 2011.

[33] Pu Wang, Bidong Liu, and Tao Hong. Electric load forecasting with recency effect: A big data approach. *International Journal of Forecasting*, 32(3):585 – 597, Jul.–Sept. 2016.

[34] N. Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *IEEE Transactions on Power Systems*, 16(3):498–505, Aug. 2001.

[35] N.Amjady. Short-term bus load forecasting of power systems by a new hybrid method. *IEEE Transactions on Power Systems*, 22(1):333–341, Feb 2007.

[36] R. J. Hyndman and S. Fan. Density forecasting for long-term peak electricity demand. *IEEE Transactions on Power Systems*, 25(2):1142–1153, May 2010.

[37] V. Dordonnat, A. Pichavant, and A. Pierrot. Gefcom2014 probabilistic electric load forecasting using time series and semi-parametric regression models. *International Journal of Forecasting*, 32(3):1005 – 1011, Jul.-Sept. 2016.

[38] J. Xie, T. Hong, T. Laing, and C. Kang. On normality assumption in residual simulation for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, 8(3):1046–1053, May 2017.

[39] J. Xie and T. Hong. Temperature scenario generation for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, 9(3):1680–1687, May 2018.

[40] S. Ben Taieb, R. Huser, R. J. Hyndman, and M. G. Genton. Forecasting uncertainty in electricity smart meter data by boosting additive quantile regression. *IEEE Transactions on Smart Grid*, 7(5):2448–2455, Sep. 2016.

[41] B. Liu, J. Nowotarski, T. Hong, and R. Weron. Probabilistic load forecasting via quantile regression averaging on sister forecasts. *IEEE Transactions on Smart Grid*, 8(2):730–737, Mar. 2017.

[42] Zhifeng Guo, Kaile Zhou, Xiaoling Zhang, and Shanlin Yang. A deep learning model for short-term power load and probability density forecasting. *Energy*, 160:1186 – 1200, Oct. 2018.

[43] Yi Wang, Ning Zhang, Yushi Tan, Tao Hong, Daniel S. Kirschen, and Chongqing Kang. Combining Probabilistic Load Forecasts. *IEEE Transactions on Smart Grid*, 3053(c):1–11, 2018.

[44] Yi Wang, Dahua Gan, Mingyang Sun, Ning Zhang, Zongxiang Lu, and Chongqing Kang. Probabilistic individual load forecasting using pinball loss guided lstm. *Applied Energy*, 235:10–20, Feb. 2019.

[45] P. E. McSharry, S. Bouwman, and G. Bloemhof. Probabilistic forecasts of the magnitude and timing of peak electricity demand. *IEEE Transactions on Power Systems*, 20(2):1166–1172, May 2005.

[46] M. Rafiei, T. Niknam, J. Aghaei, M. Shafie-Khah, and J. P. S. Catalao. Probabilistic load forecasting using an improved wavelet neural network trained by generalized extreme learning machine. *IEEE Transactions on Smart Grid*, 9(6):6961–6971, Nov. 2018.

[47] Y. Wang, N. Zhang, Q. Chen, D. S. Kirschen, P. Li, and Q. Xia. Data-driven probabilistic net load forecasting with high penetration of behind-the-meter pv. *IEEE Transactions on Power Systems*, 33(3):3255–3264, May 2018.

[48] Saeed Sepasi, Ehsan Reihani, Abdul M. Howlader, Leon R. Roose, and Marc M. Matsuura. Very short term load forecasting of a distribution system with high pv penetration. *Renewable Energy*, 106:142 – 148, Jun. 2017.

[49] Liu Jin, Dong Cong, Liu Guangyi, and Yu Jilai. Short-term net feeder load forecasting of microgrid considering weather conditions. In *2014 IEEE International Energy Conference (ENERGYCON)*, pages 1205–1209, May 2014.

[50] M. Sun, I. Konstantelos, and G. Strbac. A deep learning-based feature extraction framework for system security assessment. *IEEE Transactions on Smart Grid*, in press.

[51] M. Sun, Y. Wang, F. Teng, Y. Ye, G. Strbac, and C. Kang. Clustering-based residential baseline estimation: A probabilistic perspective. *IEEE Transactions on Smart Grid*, in press.

[52] Elena Mocanu, Phuong H. Nguyen, Madeleine Gibescu, and Wil L. Kling. Deep learning for estimating building energy consumption. *Sustainable Energy, Grids and Networks*, 6:91 – 99, Jun. 2016.

[53] Zhifeng Guo, Kaile Zhou, Xiaoling Zhang, and Shanlin Yang. A deep learning model for short-term power load and probability density forecasting. *Energy*, 160:1186 – 1200, Oct. 2018.

[54] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, Dec. 1997.

[55] H. Shi, M. Xu, and R. Li. Deep learning for household load forecasting-a novel pooling deep rnn. *IEEE Transactions on Smart Grid*, 9(5):5271–5280, Sep. 2018.

[56] W. Kong, Z. Y. Dong, D. J. Hill, F. Luo, and Y. Xu. Short-term residential load forecasting based on resident behaviour learning. *IEEE Transactions on Power Systems*, 33(1):1087–1088, Jan. 2018.

[57] W. Zhang, H. Quan, and D. Srinivasan. An improved quantile regression neural network for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, pages 1–1, in press.

[58] K. Chen, K. Chen, Q. Wang, Z. He, J. Hu, and J. He. Short-term load forecasting with deep residual networks. *IEEE Transactions on Smart Grid*, pages 1–1, in press.

[59] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark A Hasegawa-Johnson, and Thomas S Huang. Dilated recurrent neural networks. In *Advances in Neural Information Processing Systems 30*, pages 77–87. Curran Associates, Inc., 2017.

[60] A. Graves, N. Jaitly, and A. Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278, Dec. 2013.

[61] Qinghua Hu, Rujia Zhang, and Yucan Zhou. Transfer learning for short-term wind speed prediction with deep neural networks. *Renewable Energy*, 85:83 – 95, Jan. 2016.

[62] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[63] A.Gelman. *Prior Distribution*. John Wiley & Sons, 2002.

[64] Andrew Gelman. Prior distributions for variance parameters in hierarchical models (Comment on Article by Browne and Draper). *Bayesian Analysis*, 1(3):515–534, 2006.

[65] Paul H Garthwaite, Joseph B Kadane, and Anthony O'Hagan. Statistical Methods for Eliciting Probability Distributions. *Journal of the American Statistical Association*, (470):680–701, 2005.

[66] David J C Mackay. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1992.

[67] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, Mar. 1951.

[68] Y. Wang, Q. Chen, M. Sun, C. Kang, and Q. Xia. An ensemble forecasting method for the aggregated load with subprofiles. *IEEE Transactions on Smart Grid*, 9(4):3906–3908, Jul. 2018.

[69] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, Sept. 1967.

[70] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, Apr. 1963.

[71] M. Sun, F. Teng, X. Zhang, G. Strbac, and D. Pudjianto. Data-driven representative day selection for investment decisions: A cost-oriented approach. *IEEE Transactions on Power Systems*, pages 1–1, 2019.

[72] O. Abedinia, N. Amjady, and H. Zareipour. A new feature selection technique for load and price forecast of electrical power systems. *IEEE Transactions on Power Systems*, 32(1):62–74, Jan. 2017.

[73] Nima Amjady and Ali Daraeepour. Design of input vector for day-ahead price forecasting of electricity markets. *Expert Systems with Applications*, 36(10):12281 – 12294, Dec. 2009.

[74] P. A. Estevez, M. Tesmer, C. A. Perez, and J. M. Zurada. Normalized mutual information feature selection. *IEEE Transactions on Neural Networks*, 20(2):189–201, Feb. 2009.

[75] N. Kwak and Chong-Ho Choi. Input feature selection for classification problems. *IEEE Transactions on Neural Networks*, 13(1):143–159, Jan. 2002.

[76] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, May 2015.

[77] Shu Fan, Kittipong Methaprayoon, and Wei-Jen Lee. Multiregion load forecasting for system with large geographical area. *IEEE Transactions on Industry Applications*, 45(4):1452–1459, 2009.

[78] Tao Hong, Min Gui, Mesut E. Baran, and H. Lee Willis. Modeling and forecasting hourly electric load by multiple linear regression with interactions. In *IEEE PES General Meeting*, pages 1–8, 2010.

[79] Martin T. Hagan and Suzanne M. Behr. The time series approach to short term load forecasting. *IEEE Transactions on Power Systems*, 2(3):785–791, 1987.

[80] Paul Bromiley. Products and convolutions of gaussian probability density functions. *Tina-Vision Memo*, 3(4):1, 2003.

[81] Bo-Juen Chen, Ming-Wei Chang, and Chih-Jen lin. Load forecasting using support vector machines: a study on eunite competition 2001. *IEEE Transactions on Power Systems*, 19(4):1821–1830, Nov. 2004.

[82] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[83] François Chollet et al. Keras. `https://keras.io`, 2015.

[84] Dustin Tran, Alp Kucukelbir, Adji B. Dieng, Maja Rudolph, Dawen Liang, and David M. Blei. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*, 2016.

[85] Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679 – 688, Oct.-Dec. 2006.

[86] M. Rafiei, T. Niknam, J. Aghaei, M. Shafie-Khah, and J. P. S. Catalao. Probabilistic load forecasting using an improved wavelet neural network trained by generalized extreme learning machine. *IEEE Transactions on Smart Grid*, 9(6):6961–6971, Nov. 2018.

[87] H. Shaker, H. Zareipour, and D. Wood. A data-driven approach for estimating the power generation of invisible solar sites. *IEEE Transactions on Smart Grid*, 7(5):2466–2476, Sep. 2016.

[88] M. Sun, F. Teng, X. Zhang, G. Strbac, and D. Pudjianto. Data-driven representative day selection for investment decisions: A cost-oriented approach. *IEEE Transactions on Power Systems*, 34(4):2925–2936, 2019.

[89] P. Panciatici, G. Bareux, and L. Wehenkel. Operating in the fog: Security management under uncertainty. *IEEE Power and Energy Magazine*, 10(5):40–49, Sep. 2012.

[90] M. Kezunovic. Monitoring of power system topology in real-time. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, volume 10, pages 244b–244b, Jan 2006.

[91] D. R. Gurusinghe and A. D. Rajapakse. Post-disturbance transient stability status prediction using synchrophasor measurements. *IEEE Trans on Power Systems*, 31(5):3656–3664, Sep. 2016.

[92] I. Konstantelos, M. Sun, S. H. Tindemans, S. Issad, P. Panciatici, and G. Strbac. Using vine copulas to generate representative system states for machine learning. *IEEE Transactions on Power Systems*, 34(1):225–235, 2019.

[93] Prabha Kundur, Neal J Balu, and Mark G Lauby. *Power system stability and control*, volume 7. McGraw-hill New York, 1994.

[94] M.A. Pai. *Energy Function Analysis for Power System Stability*. Power Electronics and Power Systems. Springer US, 2012.

[95] H. Deng, J. Zhao, Y. Liu, and X. Wu. A real-time generator-angle prediction method based on the modified grey verhulst model. In *2011 4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, pages 179–184, July 2011.

[96] M. H. Haque and A. H. M. A. Rahim. Determination of first swing stability limit of multimachine power systems through taylor series expansions. *IEE Proceedings C - Generation, Transmission and Distribution*, 136(6):373–380, Nov 1989.

[97] J. H. Sun and K. L. Lo. Transient stability real-time prediction for multi-machine power systems by using observation. In *Proceedings of TENCON '93. IEEE Region 10 International Conference on Computers, Communications and Automation*, volume 5, pages 217–221 vol.5, Oct 1993.

[98] K. Sun, S. Likhate, V. Vittal, V. S. Kolluri, and S. Mandal. An online dynamic security assessment scheme using phasor measurements and decision trees. *IEEE Trans on Power Systems*, 22(4):1935–1943, Nov 2007.

[99] V. Krishnan, J. D. McCalley, S. Henry, and S. Issad. Efficient database generation for decision tree based power system security assessment. *IEEE Trans on Power Systems*, 26(4):2319–2327, Nov 2011.

[100] C. Liu, K. Sun, Z. H. Rather, Z. Chen, C. L. Bak, P. Thøgersen, and P. Lund. A systematic approach for dynamic security assessment and the corresponding preventive control scheme based on decision trees. *IEEE Trans on Power Systems*, 29(2):717–730, Mar 2014.

[101] I. Konstantelos, G. Jamgotchian, S. H. Tindemans, P. Duchesne, S. Cole, C. Merckx, G. Strbac, and P. Panciatici. Implementation of a massively parallel dynamic security assessment platform for large-scale grids. *IEEE Trans on Smart Grid*, 8(3):1417–1426, May 2017.

[102] M. H. Vasconcelos, L. M. Carvalho, J. Meirinhos, N. Omont, P. Gambier-Morel, G. Jamgotchian, D. Cirio, E. Ciapessoni, A. Pitto, I. Konstantelos, G. Strbac, M. Ferraro, and C. Biasuzzi. Online security assessment with load and renewable generation uncertainty: The itesla project approach. In *2016 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–8, Oct 2016.

[103] J. L. Cremer, I. Konstantelos, and G. Strbac. From optimization-based machine learning to interpretable security rules for operation. *IEEE Transactions on Power Systems*, 34(5):3826–3836, Sep. 2019.

[104] Dahai You, Ke Wang, Lei Ye, Junchun Wu, and Ruoyin Huang. Transient stability assessment of power system using support vector machine with generator combinato-

rial trajectories inputs. *International Journal of Electrical Power and Energy Systems*, 44(1):318–325, 2013.

[105] J. J. Q. Yu, D. J. Hill, A. Y. S. Lam, J. Gu, and V. O. K. Li. Intelligent time-adaptive transient stability assessment system. *IEEE Trans on Power Systems*, 33(1):1049–1058, Jan 2018.

[106] Chengxi Liu, Fei Tang, and Claus Leth Bak. An accurate online dynamic security assessment scheme based on random forest. *Energies*, 11(7):1914, Jul 2018.

[107] S. R. Samantara, I. Kamwa, and G. Joos. Ensemble decision trees for phasor measurement unit-based wide-area security assessment in the operations time frame. *IET Generation, Transmission Distribution*, 4(12):1334–1348, Dec 2010.

[108] C. Ren and Y. Xu. Transfer learning-based power system online dynamic security assessment: Using one model to assess many unlearned faults. *IEEE Transactions on Power Systems*, 35(1):821–824, Jan 2020.

[109] Chao Ren and Yan Xu. A Fully Data-Driven Method Based on Generative Adversarial Networks for Power System Dynamic Security Assessment with Missing Data. *IEEE Transactions on Power Systems*, 34(6):5044–5052, Nov 2019.

[110] Noor Izzri Abdul Wahab, Azah Mohamed, and Aini Hussain. Fast transient stability assessment of large power system using probabilistic neural network with feature reduction techniques. *Expert Systems with Applications*, 38(9):11112–11119, 2011.

[111] P. N. Papadopoulos and J. V. Milanović. Probabilistic framework for transient stability assessment of power systems with high penetration of renewable generation. *IEEE Trans on Power Systems*, 32(4):3078–3088, Jul 2017.

[112] H. Lee and B. Lee. Confidence-aware deep learning forecasting system for daily solar irradiance. *IET Renewable Power Generation*, 13(10):1681–1689, 2019.

[113] Alessandro Brusaferri, Matteo Matteucci, Pietro Portolani, and Andrea Vitali. Bayesian deep learning based method for probabilistic forecast of day-ahead electricity prices. *Applied Energy*, 250(December 2018):1158–1175, 2019.

[114] Jishnu Mukhoti and Yarin Gal. Evaluating bayesian deep learning methods for semantic segmentation. *CoRR*, abs/1811.12709, 2018.

[115] Ross Harper and Joshua Southern. A bayesian deep learning framework for end-to-end prediction of emotion from heartbeat. *CoRR*, abs/1902.03043, 2019.

[116] M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang. Using bayesian deep learning to capture uncertainty for residential net load forecasting. *IEEE Transactions on Power Systems*, 35(1):188–201, Jan 2020.

[117] A.A. Fouad, F. Aboytes, V.F. Carvalho, S.L. Corey, K.J. Dhir, and R. Vierra. Dynamic security assessment practices in north america. *IEEE Transactions on Power Systems*, 3(3):1310–1321, 1988.

[118] Benjamin Donnot, Isabelle Guyon, Antoine Marot, Marc Schoenauer, and Patrick Panciatici. Optimization of computational budget for power system risk assessment. In *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6. IEEE, 2018.

[119] Marco Martinelli, Enrico Tronci, Giovanni Dipoppa, and Claudio Balducelli. Electric power system anomaly detection using neural networks. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 1242–1248. Springer, 2004.

[120] Y Lin, J Wang, and M Cui. Reconstruction of Power System Measurements Based on Enhanced Denoising Autoencoder. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, pages 1–5, aug 2019.

[121] Yixing Wang, Meiqin Liu, Zhejing Bao, and Senlin Zhang. Stacked sparse autoencoder with PCA and SVM for data-based line trip fault diagnosis in power systems. *Neural Computing and Applications*, 31(10):6719–6731, 2019.

[122] Naftali Tishby, Esther Levin, and Sara A Solla. Consistent inference of probabilities in layered networks: Predictions and generalization. In *International Joint Conference on Neural Networks*, volume 2, pages 403–409, 1989.

[123] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2):105–112, 2009.

[124] Y. Zhang, Y. Xu, and Z. Y. Dong. Robust classification model for pmu-based on-line power system dsa with missing data. *IET Generation, Transmission Distribution*, 11(18):4484–4491, 2017.

[125] M. He, V. Vittal, and J. Zhang. Online dynamic security assessment with missing pmu measurements: A data mining approach. *IEEE Transactions on Power Systems*, 28(2):1969–1977, 2013.

[126] Benjamin Donnot, Isabelle Guyon, Antoine Marot, Marc Schoenauer, and Patrick Panciatici. Fast power system security analysis with guided dropout. In *26th European Symposium on Artificial Neural Networks, ESANN, Bruges, Belgium, April*, 2018.

[127] G. Anagnostou and B. C. Pal. Impact of overexcitation limiters on the power system stability margin under stressed conditions. *IEEE Transactions on Power Systems*, 31(3):2327–2337, May 2016.

[128] CWTNJ Balu and Dominic Maratukulam. *Power system voltage stability*. McGraw Hill, 1994.

[129] Francois Chollet et al. *Deep learning with Python*, volume 361. Manning New York, 2018.

[130] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A

system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[131] M McGranaghan, M Olearczyk, and C Gellings. Enhancing distribution resiliency: Opportunities for applying innovative technologies. *Electr. Today*, 28(1):46–48, 2013.

[132] Gang Huang, Jianhui Wang, Chen Chen, Junjian Qi, and Chuangxin Guo. Integration of preventive and emergency responses for power grid resilience enhancement. *IEEE Transactions on Power Systems*, 32(6):4451–4463, 2017.

[133] Mingyu Yan, Xiaomeng Ai, Mohammad Shahidehpour, Zhiyi Li, Jinyu Wen, Shay Bahramira, and Aleksi Paaso. Enhancing the transmission grid resilience in ice storms by optimal coordination of power system schedule with pre-positioning and routing of mobile dc de-icing devices. *IEEE Transactions on Power Systems*, 34(4):2663–2674, 2019.

[134] IE Commission et al. Microgrids for disaster preparedness and recovery-with electricity continuity plans and systems. *White Paper, Geneva, Switzerland*, 96:21–28, 2014.

[135] M. H. Amirioun, F. Aminifar, and H. Lesani. Resilience-oriented proactive management of microgrids against windstorms. *IEEE Transactions on Power Systems*, 33(4):4275–4284, 2018.

[136] Xindong Liu, Mohammad Shahidehpour, Zuyi Li, Xuan Liu, Yijia Cao, and Zhaohong Bie. Microgrids for enhancing the power grid resilience in extreme conditions. *IEEE Transactions on Smart Grid*, 8(2):589–597, 2017.

[137] Zhiyi Li, Mohammad Shahidehpour, Farrokh Aminifar, Ahmed Alabdulwahab, and Yusuf Al-Turki. Networked microgrids for enhancing the power system resilience. *Proceedings of the IEEE*, 105(7):1289–1310, 2017.

[138] Yin Xu, Chen-Ching Liu, Kevin P. Schneider, Francis K. Tuffner, and Dan T. Ton. Microgrids for service restoration to critical load in a resilient distribution system. *IEEE Transactions on Smart Grid*, 9(1):426–437, 2018.

[139] Halim Lee, Gil-Seong Byeon, Jin-Hong Jeon, Akhtar Hussain, Hak-Man Kim, Anastasios Oulis Rousis, and Goran Strbac. An energy management system with optimum reserve power procurement function for microgrid resilience improvement. *IEEE Access*, 7:42577–42585, 2019.

[140] Lei Sun, Weijia Liu, C. Y. Chung, Ming Ding, Rui Bi, and Linjie Wang. Improving the restorability of bulk power systems with the implementation of a wf-bess system. *IEEE Transactions on Power Systems*, 34(3):2366–2377, 2019.

[141] Mostafa Nazemi, Moein Moeini-Aghtaie, Mahmud Fotuhi-Firuzabad, and Payman Dehghanian. Energy storage planning for enhanced resilience of power distribution networks against earthquakes. *IEEE Transactions on Sustainable Energy*, 11(2):795–806, 2020.

[142] Ahmed S. Musleh, Haris M. Khalid, S. M. Muyeen, and Ahmed Al-Durra. A prediction algorithm to enhance grid resilience toward cyber attacks in wamcs applications. *IEEE Systems Journal*, 13(1):710–719, 2019.

[143] Tao Ding, Ming Qu, Zekai Wang, Bo Chen, Chen Chen, and Mohammad Shahidehpour. Power system resilience enhancement in typhoons using a three-stage day-ahead unit commitment. *IEEE Transactions on Smart Grid*, 12(3):2153–2164, 2021.

[144] Rong-Peng Liu, Shunbo Lei, Chaoyi Peng, Wei Sun, and Yunhe Hou. Data-based resilience enhancement strategies for electric-gas systems against sequential extreme weather events. *IEEE Transactions on Smart Grid*, 11(6):5383–5395, 2020.

[145] Hanchen Xu, Alejandro D. Domínguez-García, and Peter W. Sauer. Optimal tap setting of voltage regulation transformers using batch reinforcement learning. *IEEE Transactions on Power Systems*, 35(3):1990–2001, 2020.

[146] Qiuling Yang, Gang Wang, Alireza Sadeghi, Georgios B. Giannakis, and Jian Sun. Two-timescale voltage control in distribution grids using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 11(3):2313–2323, 2020.

[147] Sunyong Kim and Hyuk Lim. Reinforcement learning based energy management algorithm for smart energy buildings. *Energies*, 11(8):2010, 2018.

[148] Chenxiao Guan, Yanzhi Wang, Xue Lin, Shahin Nazarian, and Massoud Pedram. Reinforcement learning-based control of residential energy storage systems for electric bill minimization. In *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, pages 637–642. IEEE, 2015.

[149] Yujian Ye, Dawei Qiu, Mingyang Sun, Dimitrios Papadaskalopoulos, and Goran Strbac. Deep reinforcement learning for strategic bidding in electricity markets. *IEEE Transactions on Smart Grid*, 11(2):1343–1355, 2020.

[150] Dawei Qiu, Yujian Ye, Dimitrios Papadaskalopoulos, and Goran Strbac. A deep reinforcement learning method for pricing electric vehicles with discrete charging levels. *IEEE Transactions on Industry Applications*, 56(5):5901–5912, 2020.

[151] Md Kamruzzaman, Jiajun Duan, Di Shi, and Mohammed Benidris. A deep reinforcement learning-based multi-agent framework to enhance power system resilience using shunt resources. *IEEE Transactions on Power Systems*, pages 1–1, 2021.

[152] Mohammad Mehdi Hosseini and Masood Parvania. Resilient operation of distribution grids using deep reinforcement learning. *IEEE Transactions on Industrial Informatics*, pages 1–1, 2021.

[153] Huanhuan Nie, Ying Chen, Yue Xia, Shaowei Huang, and Bingqian Liu. Optimizing the post-disaster control of islanded microgrid: A multi-agent deep reinforcement learning approach. *IEEE Access*, 8:153455–153469, 2020.

[154] Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. Efficient exploration through bayesian deep q-networks. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9, 2018.

[155] Junyu Xuan, Jie Lu, Zheng Yan, and Guangquan Zhang. Bayesian deep reinforcement learning via deep kernel learning. *International Journal of Computational Intelligence Systems*, 2018.

[156] Rowan McAllister, Yarin Gal, Alex Kendall, Mark Van Der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc., 2017.

[157] Tingqi Zhang, Mingyang Sun, Jochen Lorenz Cremer, Ning Zhang, Goran Strbac, and Chongqing Kang. A confidence-aware machine learning framework for dynamic security assessment. *IEEE Transactions on Power Systems*, pages 1–1, 2021.

[158] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[159] Yingzhen Li and Yarin Gal. Dropout inference in bayesian neural networks with alpha-divergences. In *International conference on machine learning*, pages 2052–2061. PMLR, 2017.

[160] Tom Minka. Power EP. Technical Report MSR-TR-2004-149, 2004.

[161] Shun-ichi Amari. *Differential-geometrical methods in statistics*, volume 28. Springer Science & Business Media, 2012.

[162] Thang Bui, Daniel Hernández-Lobato, Jose Hernandez-Lobato, Yingzhen Li, and Richard Turner. Deep gaussian processes for regression using approximate expectation propagation. In *International conference on machine learning*, pages 1472–1481. PMLR, 2016.

[163] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[164] José Miguel Hernández-Lobato, Yingzhen Li, Mark Rowland, Daniel Hernández-Lobato, Thang D. Bui, and Richard E. Turner. Black-box α-divergence minimization. In *Proceedings*

of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, page 1511–1520. JMLR.org, 2016.

[165] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[166] Alfréd Rényi. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, pages 547–561. University of California Press, 1961.

[167] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[168] Peter Henderson, Thang Doan, Riashat Islam, and David Meger. Bayesian policy gradients via alpha divergence dropout inference. *NIPS Bayesian Deep Learning Workshop*, 2017.

[169] Cheng Wang, Wei Wei, Jianhui Wang, Feng Liu, Feng Qiu, Carlos M. Correa-Posada, and Shengwei Mei. Robust defense strategy for gas–electric systems against malicious attacks. *IEEE Transactions on Power Systems*, 32(4):2953–2965, 2017.

[170] Zhi Chen, Lei Wu, and Yong Fu. Real-time price-based demand response management for residential appliances via stochastic optimization and robust optimization. *IEEE Transactions on Smart Grid*, 3(4):1822–1831, 2012.