Energy Management & Economic Evaluation of Grid-Connected Microgrid Operation

National University of Singapore

Department of Electrical & Computer Engineering

EE5003 - Electrical Engineering Project (Year: 2020)

Name: Hau Chong Aih
Matric Number: A0111953L
Supervisor: Prof. Panda Sanjib Kumar
Examiner: Prof. Dipti Srinivasan

Contents

List of Figures

List of Tables

Abstract

In the paper, we present our strategy and algorithms in managing the energy in the microgrid, more specifically the energy transaction with the main grid of each hour in consideration of the consumer load, solar power and dynamic electricity pricing. The main objective of the algorithm is to optimize the operation of the energy storage system (ESS) and maximize the monetary benefit of the microgrid given the fluctuating consumer load, solar power and electricity price. Beside the monetary benefit, the algorithm also factors in the minimal energy to be reserved since it is of utmost importance to ensure the continuity of mission-critical operations in the microgrid. We study and analyze the performance of two energy management algorithms - 1) model predictive control linear programming with forecast future knowledge (MPCLPF) and 2) reinforcement learning without future knowledge. In MPCLPF, different forecasting algorithms are analyzed and the optimal forecasting algorithms are integrated into the model predictive control. With the predicted information, the ESS operation is optimized with linear programming taking into account future knowledge. In reinforcement learning, the algorithm makes use of only the current information to optimize the energy transaction decision. Two behaviors of microgrid operators, namely risk-seeking (RSRL) and risk-averse (RARL) are studied to analyze its impact on the transaction decision. The performance of the algorithms (MPCLPF, RSRL and RARL) are evaluated in terms of gaining monetary benefit and maintaining system reliability. Besides, a graphical user interface is designed to better visualize the functionality of the algorithms.

1 Introduction

Due to the exponential increase in energy demand, it results in rapid depletion of fossil fuels and increased greenhouse gas emissions. Renewable energy resources are being deployed on a large scale to create a sustainable energy system with minimal environmental pollution [6]. The integration of distributed energy into the power system creates the concept of microgrids. A microgrid is defined as a distribution network of distributed energy resources, energy storage system (ESS), consumer load and critical loads as depicted in Figure 1. It can operate in either islanded or grid-connected mode subject to the characteristic of the main grid [7]. Microgrids offer several advantages such as ancillary services, demand response, decentralization of energy supply and reduction in greenhouse gas emission.

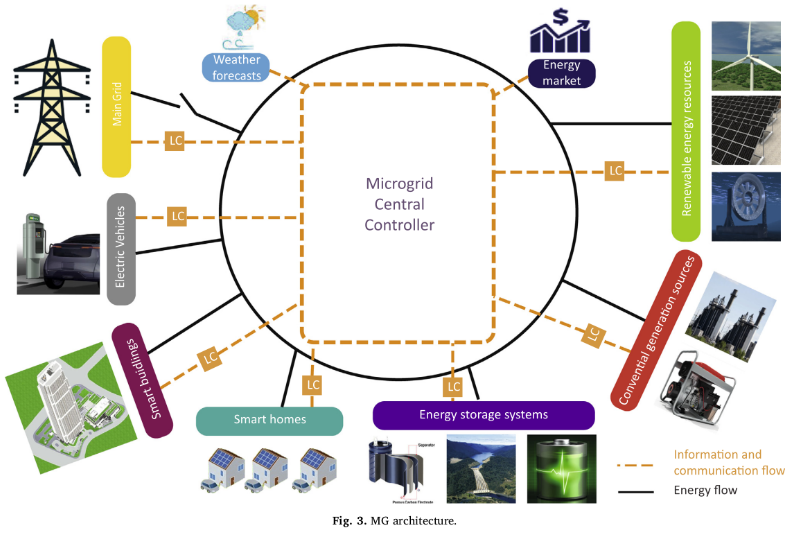


Figure 1: Microgrid Architecture [1]

An energy management system (EMS), which encompasses both supply and demand-side management, is essential for microgrids to optimally use the distributed energy in a coordinated and reliable way. The EMS performs functions such as monitoring, analyzing and forecasting load consumption, power generation and electricity prices. With these functionalities, it allows microgrids to gain benefit from optimal generation dispatch, energy saving, enhanced reliability and loss cost reduction. Thus, it is important to have a suitable EMS in microgrids. In the past years, many researchers have proposed different energy management strategies using various approaches to achieve the efficient and optimal operation of microgrids. An review of these solution approaches and strategies are described in the subsection 1.1.

1.1 Literature Review

A mixed integer linear programming (MILP) model EMS was proposed in [8] to achieve energy trading profit using the integration of thermal storage and solar panel. The authors deployed a radial basis neural network (NN) to forecast the consumer load and solar power to manage the uncertainty. The author of [9] proposed an optimal revenue maximization model by considering utility peak shaving incentives and providing demand response services. The forecast information is used in solving the MILP optimization problem. In [10], the authors performed a comparative analysis among the EMS models with perfect forecast and imperfect forecast and accurate information in respect of saving and computational time. On top of MILP model EMS, a rule based approach was proposed in [11] to manage a microgrid consisting of prosumers and a micro-turbine. The central EMS uses rule-based optimization method to control energy operation of whole microgrid while the power balancing and frequency regulation is managed by a prosumer EMS using the forecast information.

Due to the rising popularity of artificial intelligence and availability of computation power, there are increasing number of artificial intelligence based EMS proposed. The authors of [12] proposed a multi-objective EMS which minimizes the emission and operation costs of microgrid by making use of fuzzy logic to determine the charging and discharging rate of battery. A ensembled NN was used to predict the consumer load. A Lagrange programming NN approach was used in [13] to minimize the overall cost of microgrid with a radial basis neural NN used for prediction. The approach achieved a better result than that of the particle swarm optimization. In [14], a recurrent NN was used for prediction and a EMS was designed to maximize the usage of renewable energy within a microgrid. A reinforcement learning based EMS was proposed in [15] to achieve the maximum utilization of battery and renewable energy resources and the minimum dependence on main grid. To account renewable energy uncertainty, Markov Chain model is used to generate scenarios for forecast renewable energy.

The decentralization of power system allows different members in the power system to behave as an individual agent. The agents can communicate and interact with each other to exchange information and energy. It gives rises to the multi-agent system EMS aiming to optimize the operation of multiple agents in the system. In [16], a multi-agent system based approached was presented for load generation balance through load shedding and battery scheduling. Through effective coordination of renewable energy, battery and consumer load, the system maintains the energy balance. The auto regressive moving average method is used in prediction of loads and renewable energy. The authors of [17] proposed a multi-objective multi-agent EMS for a grid-connected microgrid system to minimize the emission and operation cost and line losses. The EMS performs in a hierarchical order - the energy optimization is solved in the upper level, the coordination among different is performed in the middle level and the voltage and frequency regulation is achieved in the low level.

As contrast to the papers reviewed above managing the uncertainty using the forecast method, there are authors adopted stochastic optimization approach which includes scenario generation. In [18], a stochastic coordination framework was introduced to minimize the operational cost of storage system and energy trading cost. A two point estimated method is used to handle the uncertainty due to wind power and electricity price. The author of [19] proposed a stochatic EMS for an isolated microgrid to minimize the operation cost of ESS and penalty cost on load shedding and dumped power. A scenario tree based approach is adopted to generate scenarios and the number of scenarios are reduced through scenario reduction methods.

To consistently optimize the decision making considering future information, model predictive control (MPC) was proposed in several papers. A multi-objective EMS was introduced in [20] to minimize the operation and emission cost of a microgrid. An artificial NN was used in forecasting the future information required in solving the optimization. In [21], the authors proposed a MPC EMS to minimize the operation cost of conventional generator and penalty cost on load shedding with the use of an artificial NN forecasting algorithm to predict the consumer loads. The effects of demand response on microgrid's performance was studied in the paper also.

1.2 Motivation

From the literature review, we break down the EMS into three parts: 1) objective functions, 2) uncertainty management, 3) optimization method. Firstly, the objective function of the EMS determines how the microgrid behaves in response to different scenarios. In these papers, the factors considered in the objective functions include greenhouse gas emission cost, energy transaction cost, operational cost, network loss and load shedding penalty. Despite the variety of the factors considered, none of these papers consider the reliability of microgrid which determines how long the critical mission operation in microgrid can operate in the event of the main grid power outage. Secondly, the uncertainty management handles the stochasticity arises from the renewable energy and consumer loads. The papers either assume the availability of forecast data or adopt neural network forecasting model and scenario generation model. Since the forecast information is of significance in EMS, a comparative analysis should be performed among different forecasting models to optimize the selection. Thirdly, the optimization method decides how the problem shall be solved to optimize the objective functions. The papers have proposed excellent methods in solving the problem identified by the respective authors. However, there is no paper study the performance of different optimization methods.

To prevent an overwhelming scope, in the project, we focus on a single agent EMS which interacts only with the main grid and we analyze the performance of two optimization approaches, namely model predictive control (MPC) and reinforcement learning (RL). The MPC is selected as it considers the future information in optimization and consistently optimizes the decision once the information is updated. By having such characteristics, the MPC is more far sighted. However the MPC works only with an accurate prediction models which makes the EMS complex. On the other hands, RL makes decision based on the current information once it is trained which makes the EMS light weight and simple to implement. With the optimization approaches and in view of the limitations identified, the objective of the project is as follows:

- Study role of ESS being a source of contingency reserve for the critical mission operation on top of energy trading medium.
- Identify the optimal forecasting models for consumer load, renewable energy and electricity price used in model predictive control.
- Study and identify the optimal energy management algorithms to maintain microgrid's reliability and maximize monetary benefits. It includes formulation of a suitable objective function and constraint for the MPC based EMS and design a suitable reward function for the RL based EMS.
- Develop a graphical user interface for good visualization of the algorithms' decision.

In the report, we present our strategy and algorithms in managing the energy in the microgrid, more specifically the energy transaction with the main grid of each hour in consideration of the consumer load, solar power and dynamic electricity pricing. The main objective of the algorithm is to optimize the operation of ESS and maximize the monetary benefit of the microgrid given the fluctuating consumer load, solar power and electricity price. Beside the monetary benefit, the algorithm also factors in the minimal energy to be reserved since it is of utmost importance to ensure the continuity of mission-critical operations in the microgrid. We first define the models used in the algorithms in Section 1.3, followed by the detailed explanation of two algorithms deployed (model predictive control linear programming with forecast knowledge and reinforcement learning with no future knowledge) in Section 2 and 3. We then compare and analyze the results of the two algorithms and also the impact of different MGO's behavior to the decision making in Section 4. Lastly we present a graphical user interface to visualize the algorithms' decision in Section 5 and draw our conclusion in Section 6.

1.3 Model

1.3.1 Prosumer Model

In the project, we consider a standalone microgrid, composed of an aggregated consumer load, a PV farm and an ESS, which is connected only with the main grid through a power distribution line. The microgrid is managed by a microgrid operator (MGO) which has full control of all the microgrid operations including the energy transaction operation with the main grid. To increase the microgrid's stability or ensure the continuity of mission-critical operation, the amount of target contingency reserve (hereafter referred to as ESS target state-of-charge (SOC)) during a power outage of the main grid is computed and should be maintained by the microgrid operator.

Without loss of generality, we also make the following assumption to the microgrid system: 1) we consider an N time-slot system, where the total number of slots denoted as N, with the duration of each slot is normalized to a unit time. Each time slot, denoted as $\triangle t$, has a duration of an hour which indicates there are 24 time slots for energy scheduling within a day; 2) a quasi-static time-varying energy model is considered for convenience of analysis, in which the energy rate is constant within each slot but varies from one to another; 3) the microgrid is assumed to be power-balanced such that any excess (insufficient) power is exported (imported) to (from) the grid [22].

1.3.2 Energy Storage System (ESS) Model

As an important component of microgrid, a comprehensive model of ESS is required since the ESS affects the overall performance of the algorithm significantly. In this report, the ESS can do both charging and discharging action and the magnitude of action ranges from -1 to 1 where the negative value represents the ESS discharge and vice versa. Two different ESS action sets are defined for the two algorithms: 1) a continuous action set $A_t \in [-1, 1]$ for linear programming, 2) a discrete action set $A_t \in \{-1, -0.8, ..., 0.8, 1\}$ for reinforcement learning. The action hereafter is always referred as the ESS charging or discharging action. Within each time slot t, the ESS is limited to perform only any one action, and the SOC is updated as:

$$SOC_{t+1} = \begin{cases} SOC_t + \frac{A_t \times P_{rated} \times \eta_c \times \triangle t}{E_{rated}} & \text{if } A_t \geq 0 \\ \\ SOC_t + \frac{A_t \times P_{rated} \times \triangle t}{E_{rated} \times \eta_d} & \text{otherwise} \end{cases} \tag{1}$$

where η_c, η_d, P_{rated} and E_{rated} denote the charging, discharging efficiencies, power rating and energy capacity of the ESS. Besides, the wear and tear cost is taken into consideration in the energy transaction to discourage any unnecessary arbitrage and prevent an increase in long term cost due to the degradation [23]. The ESS wear cost coefficient, denoted as k, is defined where C_i, δ and N_c denote the ESS initial investment cost, depth-of-discharge (DOD) and life cycles at a rated DOD:

$$k = \frac{C_i}{\eta_d \times E_{rated} \times \delta \times N_c} \tag{2}$$

2 Model Predictive Control Linear Programming with Forecast Future Knowledge

Model predictive control (MPC) is an advanced process control method that allows the process at the current time step to be optimized while keeping the future time steps into account. MPC works in the following sequence: 1) obtains the state or the condition of the current and future time steps and note that only the current state is guaranteed to be available while the future states can only be forecast or only available in the ideal scenario; 2) solves the optimization control problem for a finite time horizon with the state information and only implement the action for the current time step; 3) advance to the next time step and repeat step 1 [24]. The main advantage of MPC is the fact that it considers future states while trying to optimize the decision for the current time step and also it always further optimize or adjust its decision every new time step. The optimizer of MPC is chosen to be linear programming in this report due to its simplicity and suitability to our work. Linear programming, also known as linear optimization, is a method to solve an optimization problem where the objective function and as well as all the constraints are linear equalities or inequalities [25].

In this report, the MPC linear programming is used to optimize the energy transaction decisions every hour. The optimization is constraint by the operation limit of ESS and the reservation of energy that should be maintained for critical-mission operation. Both the objective function and constraints are explained in detail in 2.1. To obtain the information of the future states, the forecasting models are studied and analyzed in 2.2. The best forecasting models are deployed to predict the consumer load, solar power and dynamic electricity price in the future states. With future knowledge, the MPC linear programming generates the optimal actions that can optimize the objective function. The MPC linear programming architecture is depicted in Figure 2.

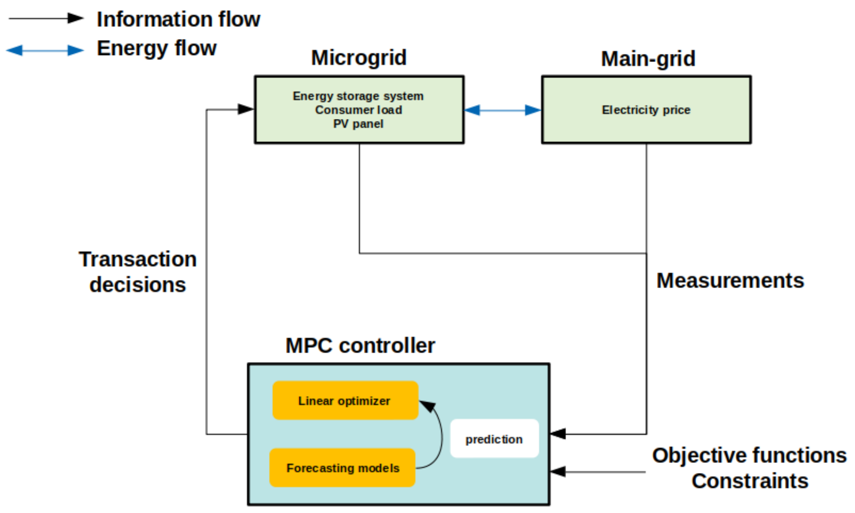


Figure 2: Model Predictive Control

2.1 Objective Function & Constraint

As discussed in the first section, the objective of the MPC linear programming is to optimize the energy transaction decisions every hour while the constraint is to operate within the ESS limit and reserve a sufficient amount of energy for critical-mission operation, also referred as ESS target SOC. The objective function Obj_t of linear programming is designed to minimize the cost of purchasing the electricity from the main grid and is formulated as follows:

$$Obj_t = min \sum_{t_k=t}^{t+23} [(Pr_{t_k} + k) \times A_{t_k} \times P_{rated} \times \frac{L_{t_k}}{L_{t_k} + PV_{t_k}}] \tag{3}$$

where Pr_{t_k} denotes the electricity price at time t_k, A_{t_k} denote the ESS operating action,

P_{rated} denotes the rated power of ESS, L_{t_k} and PV_{t_k} denote the consumer load and solar power at time t_k. Intuitively, the charging or discharging action of ESS is much determined by the electricity price when there is no solar power; ESS tends to charge (discharge) when the solar power is much higher (lower) than the consumer load. Beside the objective function, the constraint is also formulated in equation 4 and 5 such that the contingency reserved for the finite planning horizon is always higher than the target SOC and also the operating limit defined for the ESS operating action. The equation 6 provides a bound for the selection of operating action.

$$\begin{bmatrix} A_t & 0 & 0 & ... & 0 \\ A_t & A_{t+1} & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ A_t & ... & ... & A_{t+22} & 0 \\ A_t & ... & ... & A_{t+22} & A_{t+23} \end{bmatrix} \times \frac{Prated}{E_{rated}} <= \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \\ 1 \end{bmatrix} \times (1 - SOC_t) \tag{4}$$

$$\begin{bmatrix} A_t & 0 & 0 & ... & 0 \\ A_t & A_{t+1} & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ A_t & ... & ... & A_{t+22} & 0 \\ A_t & ... & ... & A_{t+22} & A_{t+23} \end{bmatrix} \times \frac{-Prated}{E_{rated}} <= \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \\ 1 \end{bmatrix} \times (SOC_t - SOC_{target}) \tag{5}$$

$$A_{t_k} \in [-1, 1] \; \forall t_k \in \{t, t+1, ...t+23\} \tag{6}$$

2.2 Forecasting Models

One important component of MPC linear programming is the forecasting model deployed in predicting the consumer load, solar power and electricity price. With the forecasting models, it makes the energy management algorithm more far-sighted and takes into consideration of future information in optimizing the energy transaction decision. In this subsection, we introduce and evaluate the performance of 4 deep learning forecasting models of time series data, namely long-short-term-memory (LSTM), sequence-to-sequence (Seq2Seq), attentive sequence-to-sequence (AttSeq2Seq) and transformer.

2.2.1 Long-short-term-memory (LSTM)

LSTM is a special kind of recurrent neural network (RNN) that is capable of learning long-term dependencies. RNN is a chain-like neural network that has been widely applied in speech recognition, translation and image captioning. The chain-like structure of RNN relays the information of past time steps to the present time step which makes them useful in capturing the relationship between sequences and lists. However, the underlying problem of RNN is its reliance on a single gate or neuron in each sequence as depicted in Figure 3 to capture the information of all sequences before it. When the sequence grows, the performance of RNN might deteriorate due to the vast information presented to the gate [26].

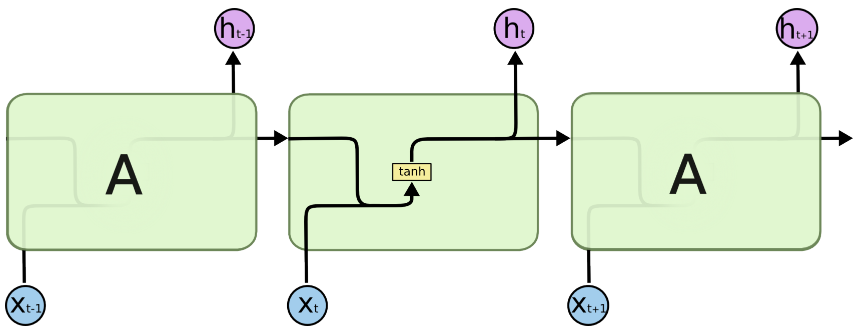


Figure 3: Repeating Module in a RNN Containing a Single Gate

In view of the problems, LSTM is introduced and designed to avoid the long term dependency problem [?]. The same chain-like structure is used by LSTM but there are now 4 gates, namely forget gate, input gate, content gate and output gate as depicted in

Figure 4. The forget gate is used to control the amount of information inflow of the past time steps to the current time step; the input and content gate is used to control the new information to be updated in the current time step; the output gate is used to decide the information to be fed to the next time step. Having four gates to control and modify the information flow, the two states, hidden states (also called output states) and cell states, are produced at each time step which helps LSTM better addressing the issue of RNN faced in a long sequence.

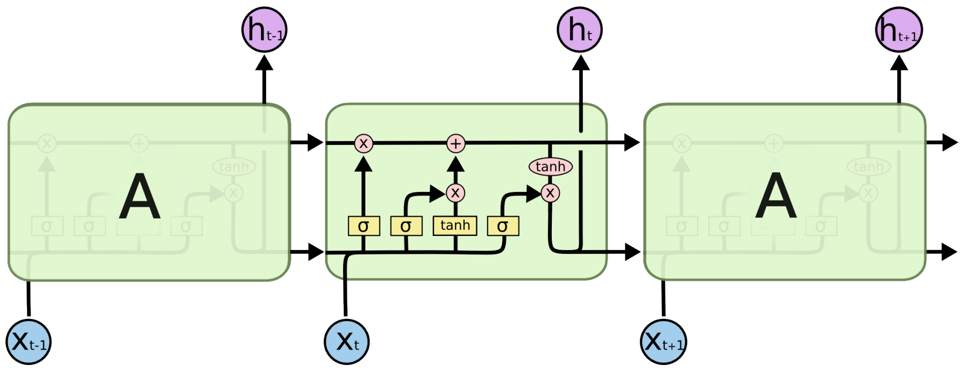


Figure 4: Repeating Module in a LSTM Containing 4 Gates

In this report, a LSTM neural network architecture is designed to predict the future 24-hours data based on the hourly data of the prior two weeks as depicted in Figure 5. The architecture contains 1 input layer with 168 neurons taking in the hourly data, 2 LSTM layers with 32 neurons each and 1 output layer with 24 neurons predicting the future values. It should be noted that only the final hidden state in the second layer of LSTM is used for the prediction of the outputs.

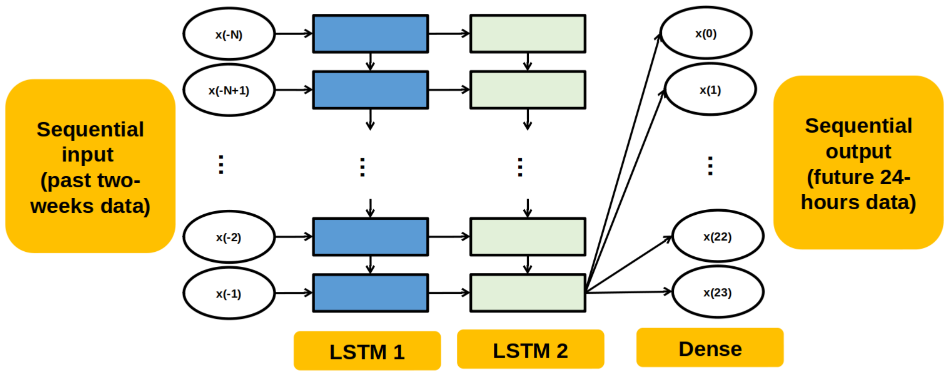

Sequential input (past two-weeks data)
x(-N)
x(-N+1)
x(-2)
x(-1)
x(0)
x(1)
x(22)
x(23)
Sequential output (future 24-hours data)
LSTM 1
LSTM 2
Dense

Figure 5: LSTM Neural Network Architecture

2.2.2 Sequence-to-sequence (Seq2Seq)

From Section 2.2.1, a LSTM neural network architecture is designed where the future 24-hours data are predicted independently based on the last sequence output of the LSTM layer. In the section, we modify the LSTM architecture to a Seq2Seq model which is proposed to encode the input sequence to a vector of a fixed dimensionality and subsequently decode the target sequence from the vector [27]. The underlying rationale of the Seq2Seq model is to capture the sequential dependency of the targets when making the prediction.

The Seq2Seq model comprises 2 deep LSTM layers acting as encoder and decoder respectively as depicted in Figure 6. The encoder aims to encapsulate the information from the input sequence in order to help the decoder making accurate predictions while the decoder receives the final hidden state of the encoder and deciphers the encoded information to make a prediction. The output of the decoder is predicted sequentially in which the information of past time steps stored in hidden and cell states is used in prediction of the current time step.

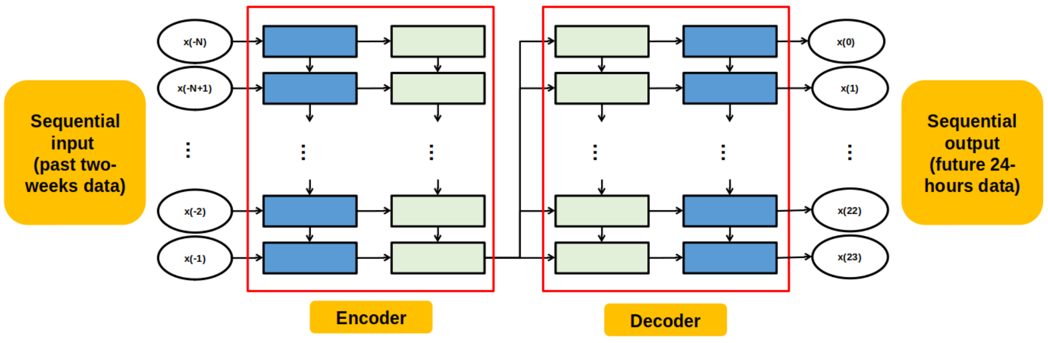

Sequential input (past two-weeks data)
x(-N)
x(-N+1)
x(-2)
x(-1)
Encoder
Decoder
x(0)
x(1)
x(22)
x(23)
Sequential output (future 24-hours data)

Figure 6: Seq2Seq Neural Network Architecture

2.2.3 Attentive Sequence-to-sequence (AttSeq2Seq)

In this section, we modify the Seq2Seq model to an AttSeq2Seq model which contains an additional attention layer between encoder and decoder as depicted in Figure 7 [28]. In the Seq2Seq model, the encoder processes the input sequence and compresses the information into a context vector out of the encoder's final hidden state. The context vector is then passed as an input to every sequence of the decoder. On the other hand in the AttSeq2Seq model, the context vector has access to all the hidden states of the encoder and it is produced individually for each sequence of the decoder.

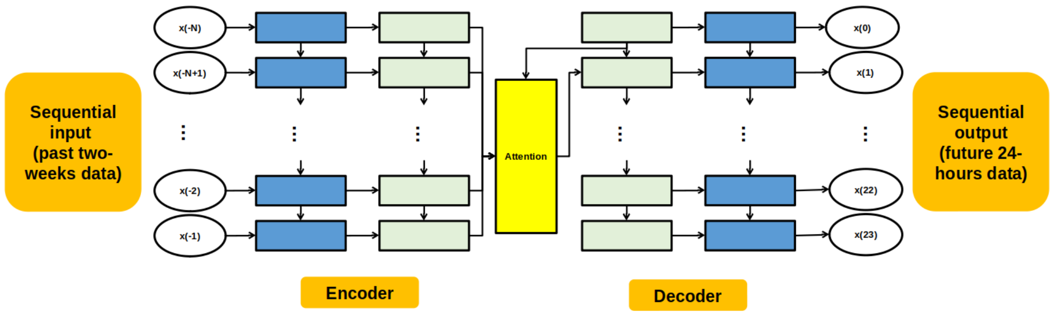

Sequential input (past two-weeks data)
x(-N)
x(-N+1)
x(-2)
x(-1)
Attention
Encoder
Decoder
x(0)
x(1)
x(22)
x(23)
Sequential output (future 24-hours data)

Figure 7: AttSeq2Seq Neural Network Architecture

Beside the context vector, the attention layer is the key that differentiates Seq2Seq and AttSeq2Seq model. As depicted in Figure 7, the attention layer works by first concatenating the hidden states of the encoder and the hidden state of the previous time step of the decoder; second, followed by processing it with a fully connected layer with softmax activation function; lastly, the attention is multiplied with the hidden states of the encoder which serves as a weighted input to the current sequence of the decoder. By having the attention layer in place, the decoder knows which hidden states of the encoder it should pay more attention in order to achieve a better result.

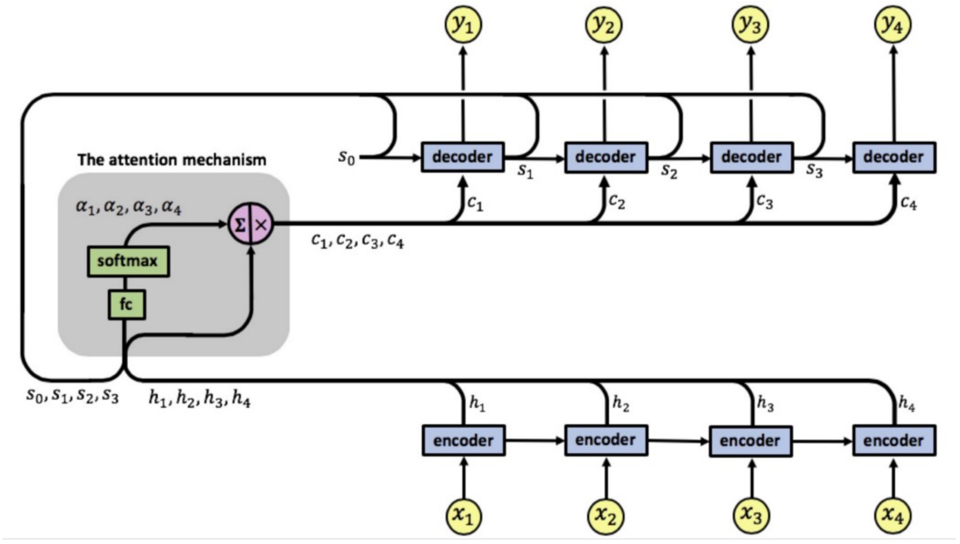


Figure 8: An Example of Attention Layer of AttSeq2Seq Model [2]

2.2.4 Transformer (TX)

In Section 2.2.1 to 2.2.3, we introduce 3 time series forecasting models based on LSTM. In recent year, there is more and more attention diverted to the convolution neural network (CNN) due to its capability in parallel computation which results in faster model training than that of using LSTM. A novel natural language processing (NLP) TX model architecture as depicted in Figure 9 is introduced to achieve the functionality as per LSTM in capturing the dependencies and also parallelization [3].

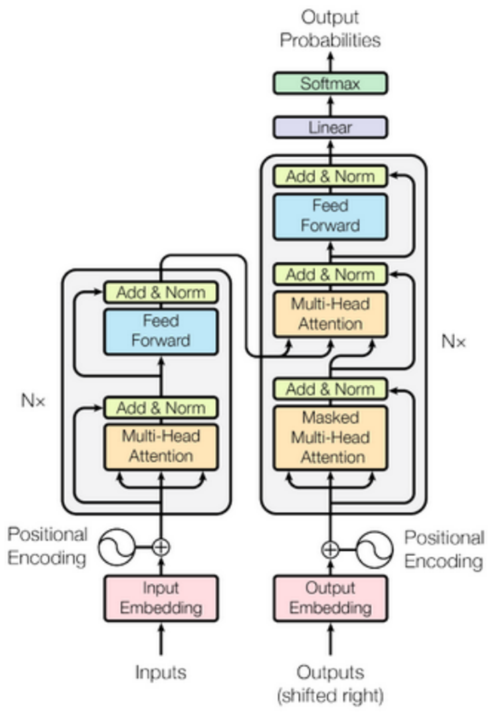


Figure 9: Transformer Architecture [3]

The TX model works by having an identical number of identical encoders and decoders stacked on top of each other as depicted in Figure 10. In each encoder layer, there is a self-attention layer which first creates 3 matrices (query, key and query) from the encoder input sequence and finds the attention matrix afterward. The attention matrix is then processed with a feed-forward layer before passed to the next layer of the encoder. The output of the last encoder layer is passed to the decoder. In each decoder, the attention matrix of the decoder input sequence is first found which is later used to find the encoder-decoder attention matrix. The encoder-decoder attention matrix helps the decoder to focus on the appropriate parts of the input sequence. At the end of the decoder layers, the output is passed through a feed-forward layer to obtain the target or prediction. Interested reader can refer to [3] for more detail information.

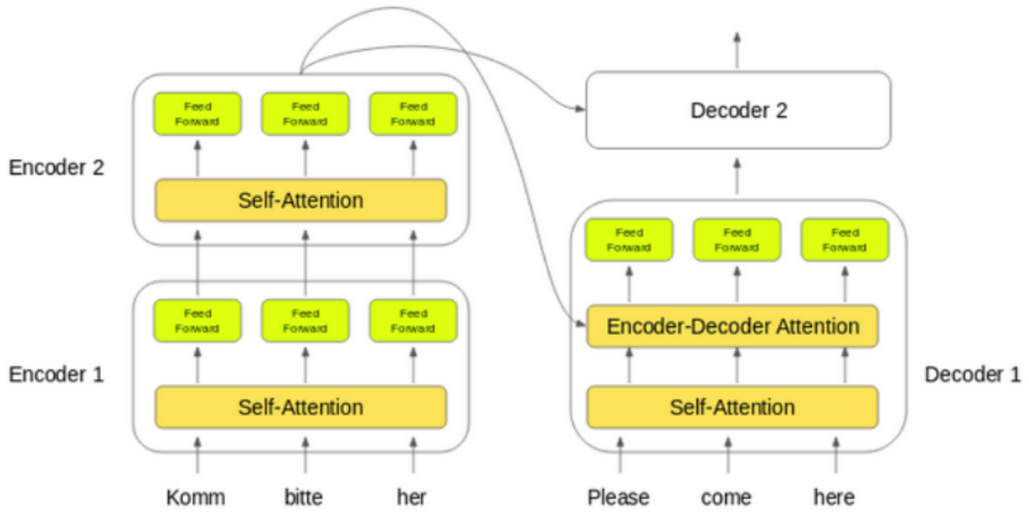


Figure 10: Transformer Working Principle Diagram [4]

In this report, the transformer model is modified to predict the future 24-hours data based on the hourly data of the prior two weeks as depicted as Figure 13. The first modification is the replacement of the embedding layer with a convolutional layer. Since the model is not used for NLP purposes, the embedding layer which is normally used to transform semantic information to geometric information is not needed. The second modification is the replacement of dense layers with convolutional layers when generating query, key and value in the self-attention layer. The rationale is that if the patterns in time series may change with time significantly due to various events and so whether an observed point is part of the patterns, is highly dependent on its surrounding context. Thus, the dense layers which generate query, key and value based on their point-wise values fails to fully leverage local context-like shape as shown in Figure 11. With convolutional layers, the local context can be better utilized which enhances the locality as shown in Figure 12. Interested reader can refer to [5] for more detail information.

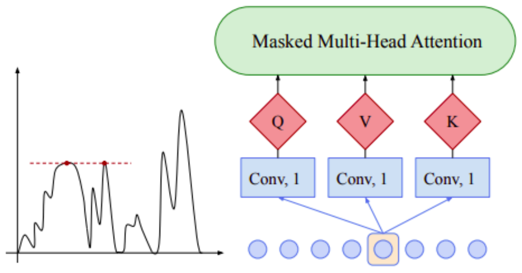


Figure 11: Failure of Dense Layer in Capturing Local Context [5]

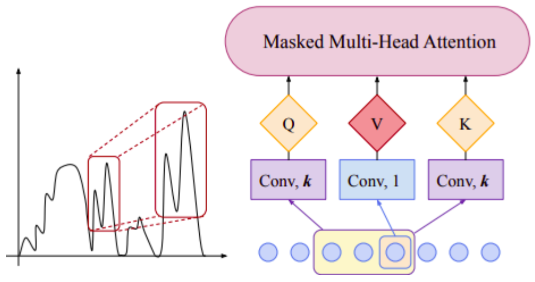


Figure 12: Enhancement of Locality with Convolutional Layer [5]

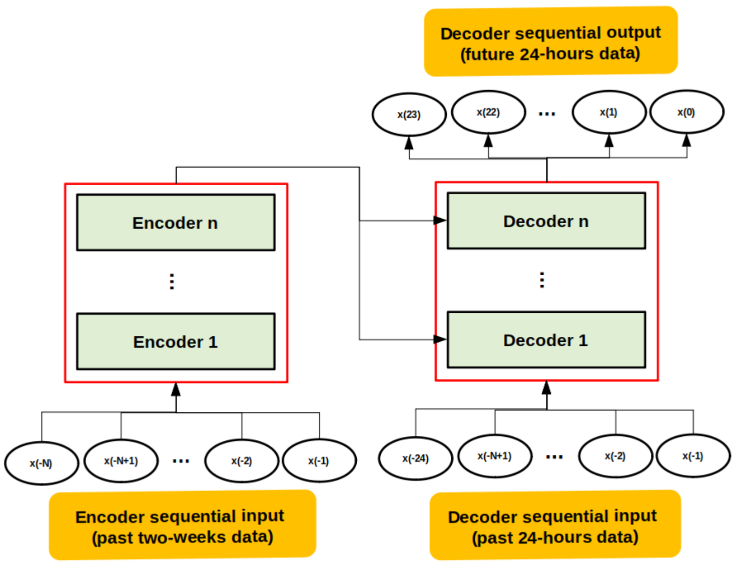

Decoder sequential output (future 24-hours data)
x(23)
x(22)
...
x(1)
x(0)
Encoder n
Encoder 1
Decoder n
Decoder 1
x(-N)
x(-N+1)
...
x(-2)
x(-1)
x(-24)
x(-N+1)
...
x(-2)
x(-1)
Encoder sequential input (past two-weeks data)
Decoder sequential input (past 24-hours data)

Figure 13: Transformer Architecture

2.2.5 Simulation Setting & Forecasting Result Comparison

In this section, the performance of 4 forecasting models is compared and analyzed. The consumer load, solar power and electricity price are obtained from [29]. Prior to the training of each model, a pre-processing pipeline is executed including the normalization of data, one-hot encoding of months and hours features and also transformation of point-wise data to sequential-wise data. The pre-processed data set is then split into training and testing sets. The inputs (features) and the outputs (targets) of respective models are shown in Figures 5, 6, 7, 13. The details of each model can be found in the Appendix attached. Each model is trained with the training set for 20 epochs and the predictions are made and the mean absolute error (MAE) is computed for both training and testing set. The process is repeated for consumer load, solar power and electricity price.

The MAE of different models for consumer load, solar power and electricity price predictions are depicted in Figures 14, 15, 16. The following observations are made with respect to the MAE corresponding to each forecasting models:

- The Seq2Seq model has the least MAE for the consumer load and solar power prediction. while the TX model has the best performance for the electricity price prediction.
- The LSTM model achieves an average performance for all 3 predictions.
- The AttSeq2Seq model has the worst performance among all the models for all 3 predictions.

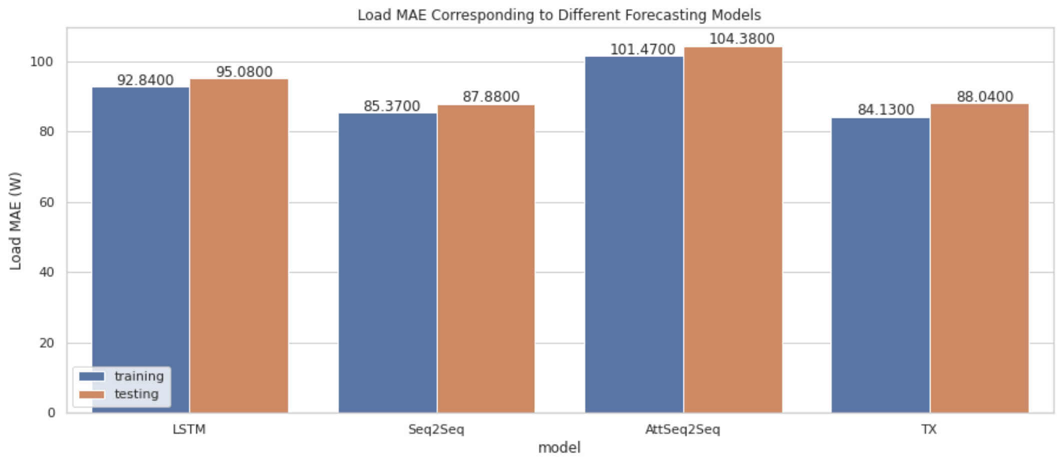


Figure 14: Consumer Load MAE Corresponding to Different Forecasting Models

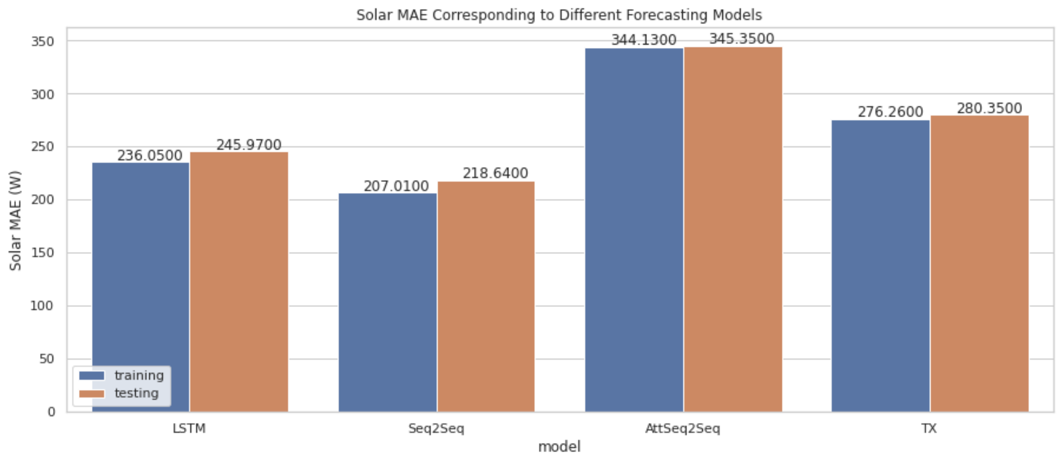


Figure 15: Solar Power MAE Corresponding to Different Forecasting Models

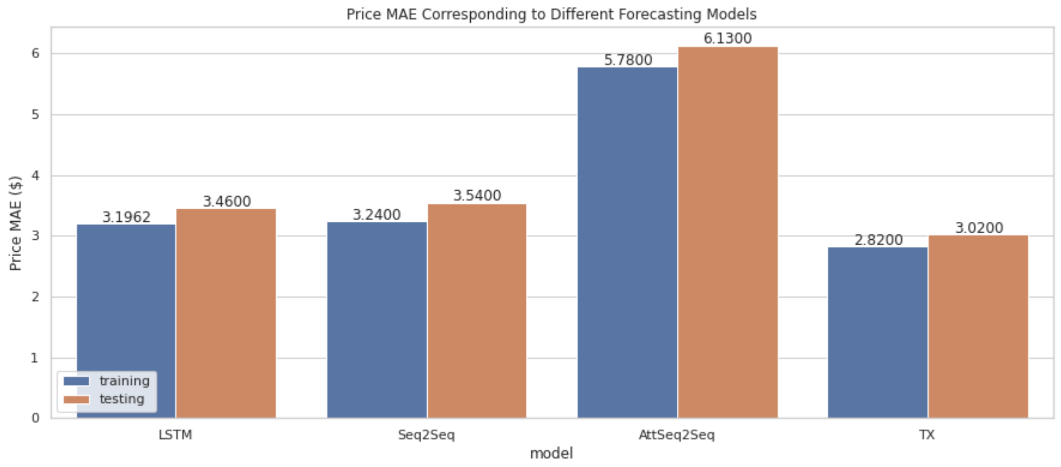


Figure 16: Electricity Price MAE Corresponding to Different Forecasting Models

The Seq2Seq model performs better than the LSTM model in all 3 predictions due to its ability to capture the dependency between time steps which allows a more accurate prediction to be made. The AttSeq2Seq model has the worst performance and it is found from the investigation that the attention obtained in the AttSeq2Seq model is not accurate as depicted in Figure 17. It is due to the poor locality of the AttSeq2Seq model since the attention is computed using point-wise data. On the other hand, the TX model can produce the attention map more accurately due to the enhancement of locality through the deployment of the convolutional layer as depicted in Figure 18. It makes the TX model the second-best prediction model for consumer load and the best model for electricity price. It is worth noting that the TX model works better in electricity price prediction due to the vastly different patterns present in electricity price while the patterns found in consumer load as observed in 19 displays a great similarity from days to days. Besides, the TX model does not work well with solar power prediction since solar power is stochastic and more uncertain and it results in difficulty of the TX model creating an accurate attention map.

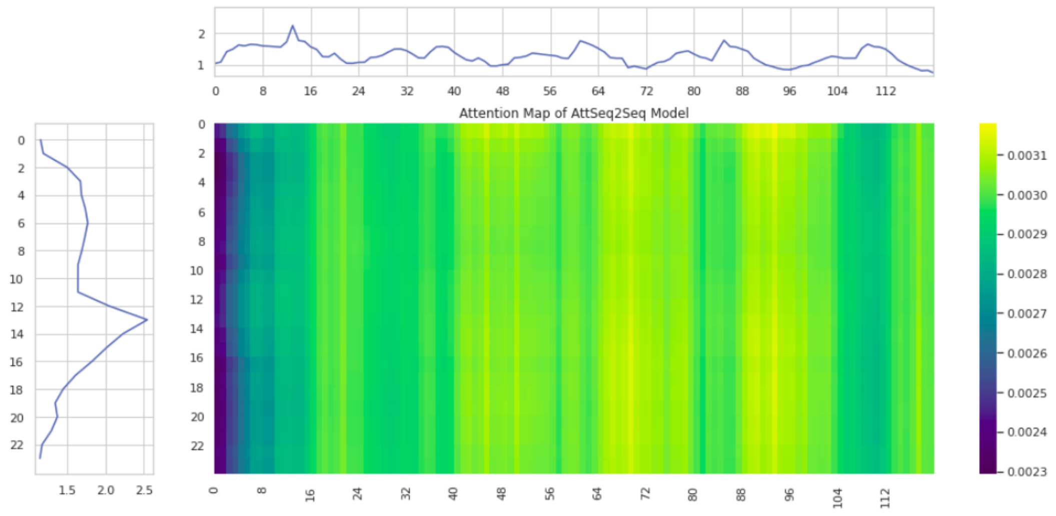


Figure 17: Price Attention Map of AttSeq2Seq Model (Decoder Input to Encoder Output)

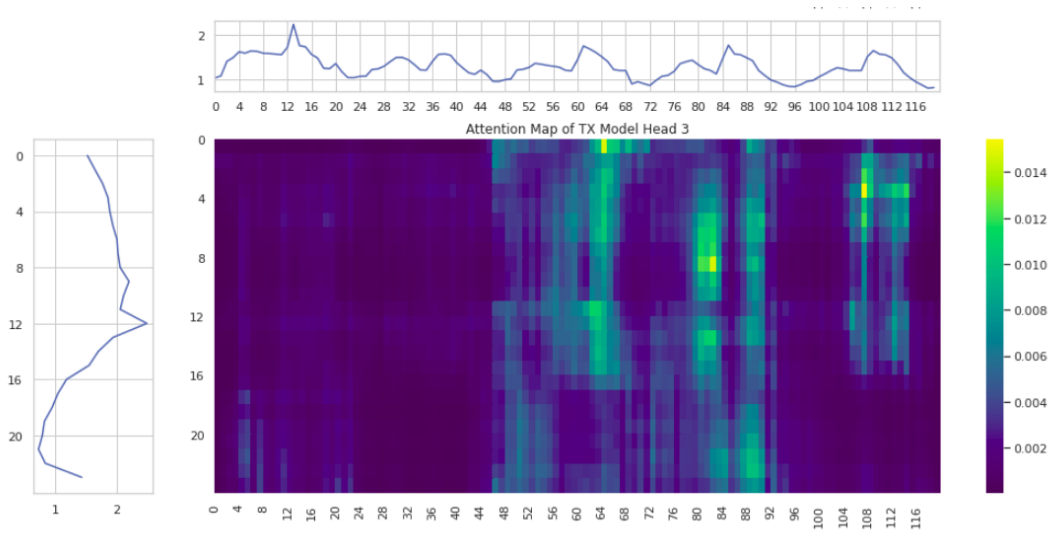


Figure 18: Price Attention Map of TX Model (Decoder Input to Encoder Input)

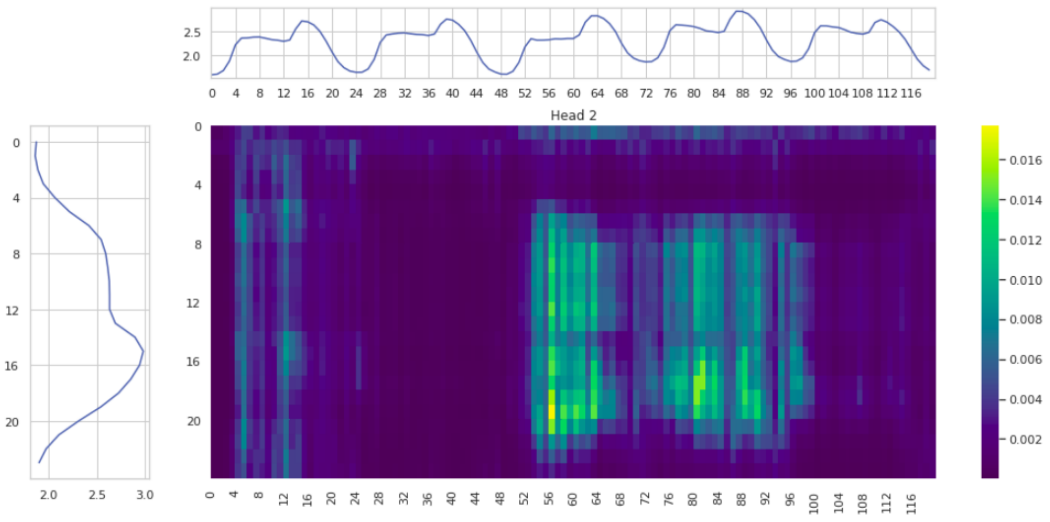


Figure 19: Consumer Load Attention Map of TX Model (Decoder Input to Encoder Input)

In conclusion, we found that the Seq2Seq model works the best for consumer load and solar power prediction while the TX model predicts the electricity price more accurately. The predictions made by the respective models are depicted in Figures 20, 21, 22. The optimal models are used in the MPC linear programming to gain knowledge of the future state which is used in optimization afterward.

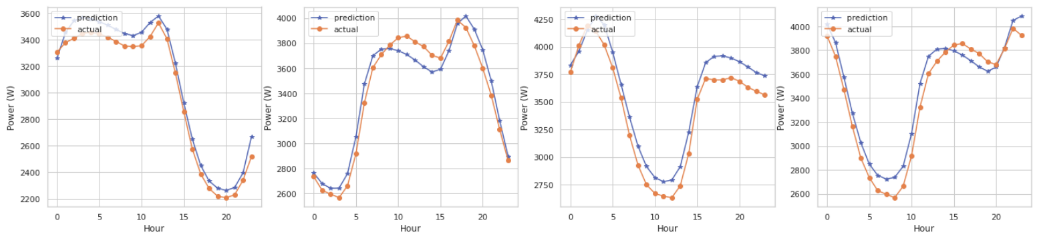


Figure 20: Consumer Load Prediction of Seq2Seq Models

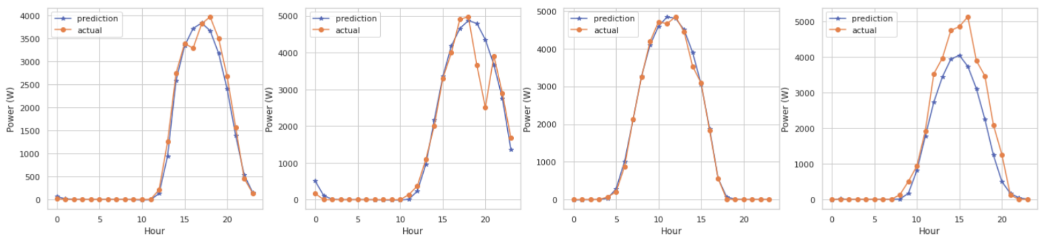


Figure 21: Solar Power Prediction of Seq2Seq Models

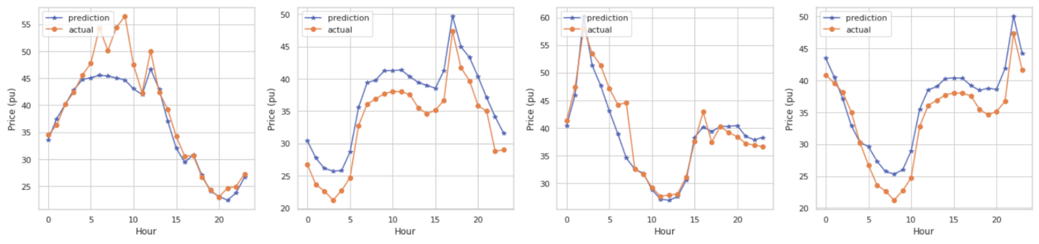


Figure 22: Electricity Price Prediction of TX Models

2.3 Procedure

With the objective function and constraints defined and the optimal forecasting models found, MPC linear programming can be performed following the procedures: 1) predict the consumer load, solar power and electricity price of the next 24 hours using the respective optimal predictive model as presented in Section 2.2; 2) with the forecast future and current known knowledge or information, run the linear programming to get the optimal sequence of actions; 3) execute the action only for the current hour and update the ESS SOC; 4) advance to the next hour and repeat the process from step 1.

3 Risk Averse & Risk Seeking Reinforcement Learning with No Future Knowledge

Reinforcement learning is a cutting edge machine learning area that is popular in the recent research community of the intelligence system. Reinforcement learning has been experimented to achieve a human level or even better performance in various decision-making process [23], for instance, AlphaGo a reinforcement learning based computer program defeated a professional human Go player in 2016. The reinforcement learning differs from the supervised learning in a way that in the supervised learning the training label or target is available and the model is trained with the correct label; whereas in reinforcement learning the target is not available and the model is trained via trial and error method in which the model learns from the mistake it makes in the past to further improve its decision in the future.

Reinforcement learning can be described as a Markov Decision Process (MDP) which is defined by the state space S, action space A, utility or reward function r (the utility and reward function are used interchangeably in the report), state transition probability matrix P and a discount factor $\gamma \in [0, 1]$. In each state S_t, the agent takes an action A_t and it results in receiving a reward r_t and movement to a new state s_{t+1} as depicted in Figure 23. The objective of the reinforcement learning is to learn a policy such that the present value of the sum of discounted future reward - the Q-value or return R_t (the Q-value and return are used interchangeably in the report), is maximized. The discount factor γ adjusts the perspective of the agent in considering the impact of its decision on the future state: 1) a small γ forces the agent to focus more on immediate rewards from next few steps and heavily discount rewards from future steps; 2) a large γ forces the agent to take into account future rewards more strongly and becomes more farsighted.

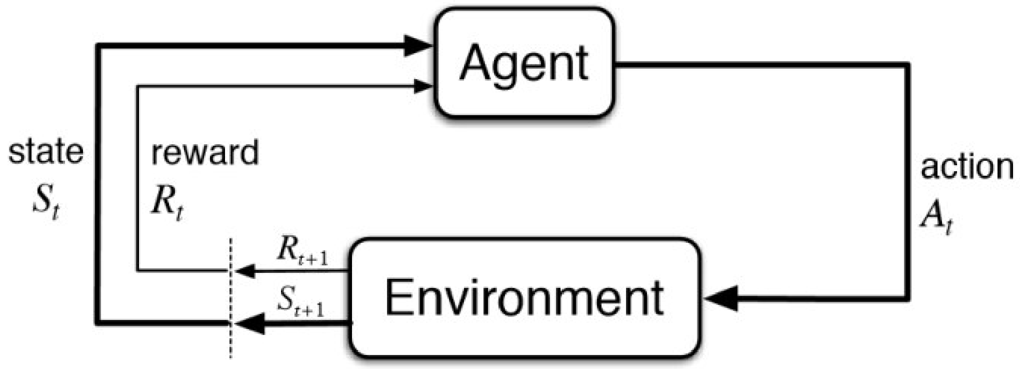


Figure 23: Reinforcement learning model

In reinforcement learning, the optimal policy is found by maximizing the Q-value and the Bellman equation comes in handy in solving the reinforcement learning problem. The Bellman equation describes the recursive relationship of Q-value of the current step and future one step as follows:

$$Q(S_t, A_t) = r_t + \gamma \times max_{A_{t+1}}(Q(S_{t+1}, A_{t+1})) \tag{7}$$

where $Q(S_t, A_t)$ is the Q-value at time step t and $max_{A_{t+1}}(Q(S_{t+1}, A_{t+1}))$ denotes the maximum Q-value taking optimal action at the subsequent time step. Under a policy, the value of taking action A_t at S_t must equal to the expected reward of transitioning into the next state S_{t+1} plus the discounted expected q-value of taking the best decision A_{t+1} at S_{t+1}. The recursive relationship between the Q-values of the consecutive steps ensures that the optimal policy can be found through iterative update rule and Q-value can converge to an optimal value. With the best policy found, the agent can act optimally according to the current state without having to know any future knowledge.

In this report, reinforcement learning is deployed to guide the MGO who acts as an agent in making make energy transaction decisions each hour using ESS in response to the consumer load, solar power, dynamic electricity pricing and reserved energy at that hour. The MGO needs to trade the energy with the main grid according to its operation limit to maximize the monetary benefit of the microgrid and also to maintain a sufficient amount of energy reserved for critical-mission operation.

3.1 Utility Function

An important aspect of reinforcement learning is to design a suitable utility or reward function such that the agent behaves or acts correctly according to the environment. Given all the requirements, a piecewise utility function is designed to guide the trading strategies. As discussed in 1.3.1, there is a minimum amount of target contingency reserve, also referred as ESS target SOC, to be maintained for sustaining the operation of critical-mission activities. The utility function is designed to be piecewise in which the utility depends on a sequence of intervals, below and beyond the target SOC. Below the target SOC, the ESS must charge as soon as possible regardless of the price. Similarly, the price must be high enough for the ESS to discharge below the target SOC. Beyond the target SOC, the criterion to charge and discharge the ESS differs. The closer the SOC is to the full SOC, lower the price should be to charge the ESS to make better use of remaining storing capacity. In contrast to the former case, the price should be higher to discharge the ESS to fully utilize the remaining available energy in the case where the SOC is closer to the target SOC. A negative exponential piecewise multiplier M is introduced, as depicted in Figs. 25 and 26, to incorporate all the criteria described. In Figure 25, the parameters m_{bef}^{upp} and m_{bef}^{low} can be adjusted to control the MGO's behavior whereas in Figure 26 the parameters m_{aft}^{ch} and m_{aft}^{dis} are tuned to control the stringency of operation criterion when the SOC is beyond the target SOC. In addition, two penalties are introduced to further control the operating action of ESS. The first penalty Pnt_t^{ESS} is assigned when the agent selects the action that will result in violation of ESS operating limit. The second penalty Pnt_t^{PV} is assigned when the ESS with spare energy capacity does not store the surplus solar power as depicted in Figure 24. The rationale of the first penalty is to take care of the ESS condition to prolong the machine life while the rationale of the second penalty is to preserve the free energy as much as possible so long as it is within the ESS operating limit. The utility function u is as follows:

$$\begin{aligned} u(Pr_t, Pr_t^{avg}, SOC_t|A_t) = {} & (M \times Pr_t^{avg} - (Pr_t + k)) \\ & \times (SOC_{t+1} - SOC_t) \times E_{rated} - Pnt_t^{ESS} - Pnt_t^{PV} \end{aligned} \tag{8}$$

$$Pr_t^{avg} = \frac{\sum\limits_{t_k=t-1}^{t-24} Pr_{t_k}}{24} \tag{9}$$

$$Pnt_t^{ESS} = \begin{cases} 0 \;\; \text{if } SOC_t + A_t \leq 1 \; or \; SOC_t - A_t \geq 0 \\ 10 \;\; \text{otherwise} \end{cases} \tag{10}$$

$$Pnt_t^{PV} = \begin{cases} 0 \text{ if } PV_t \leq (L_t + A_t \times P_{rated}) \\ exp\Big(2.5 \times (1 - SOC_{t+1})\Big) - 1 \text{ if } PV_t > (L_t + A_t \times P_{rated}) \end{cases} \tag{11}$$

$$M = \begin{cases} 1 + m_{bef}^{upp} \times exp\left(\ln\left(\dfrac{m_{bef}^{low} - 1}{m_{bef}^{upp}} \right) \times \dfrac{SOC_t}{SOC_{target}} \right) \\ \hfill \text{if } SOC_t \leq SOC_{target} \\ exp\left(\ln\left(m_{aft}^{ch} \right) \times \dfrac{SOC_t - SOC_{target}}{1 - SOC_{target}} \right) \\ \hfill \text{if } SOC_t > SOC_{target} \; \& \; A_t \geq 0 \\ m_{aft}^{dis} \times exp\left(\ln\left(\dfrac{1}{m_{aft}^{dis}} \right) \times \dfrac{SOC_t - SOC_{target}}{1 - SOC_{target}} \right) \\ \hfill \text{if } SOC_t > SOC_{target} \; \& \; A_t < 0 \end{cases} \tag{12}$$

where Pr_t, Pr_t^{avg} and M denote the dynamic price at time t, the average price across the 24 time slot before time t and the multiplier while Pnt_t^{ESS} and Pnt_t^{PV} denote the penalty cost implemented for violation of ESS operating limit and

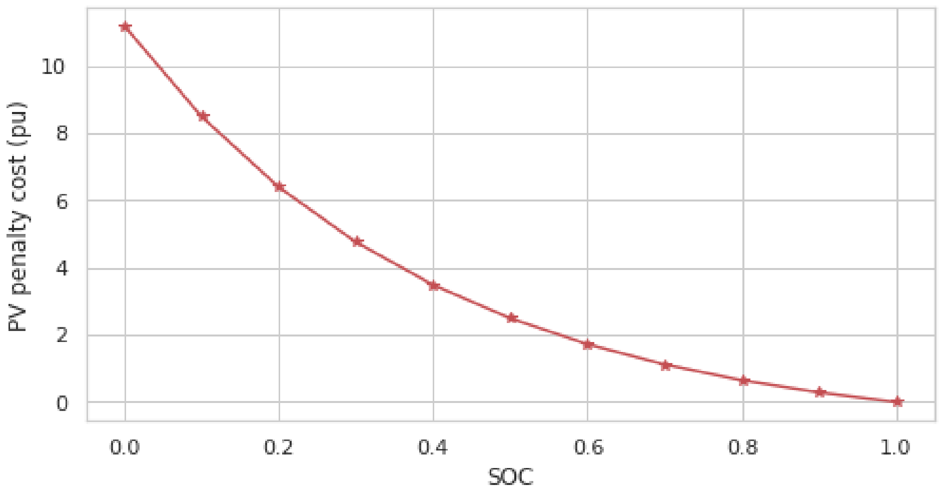


Figure 24: Penalty Assigned in the Event Surplus Solar Power is not Stored

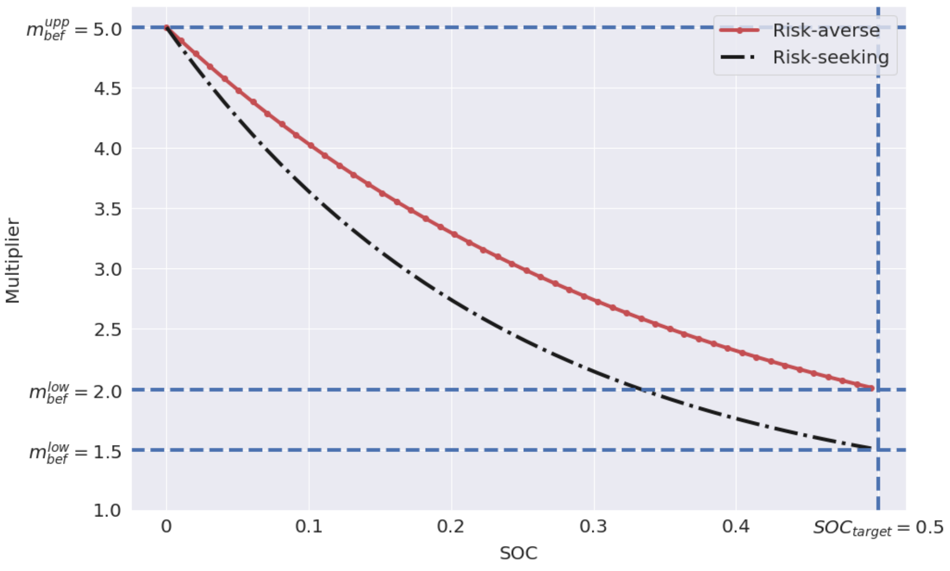


Figure 25: Multiplier below the target SOC corresponding to risk-seeking and risk-averse behaviour

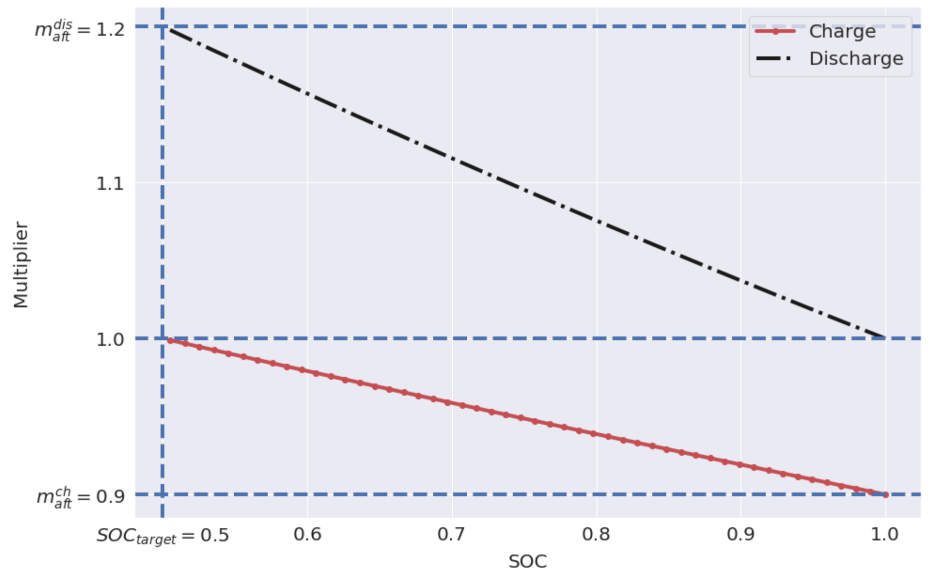


Figure 26: Multiplier beyond the target SOC corresponding to charge and discharge scenario

3.2 Deep-Q-Network (DQN)

The energy transaction decision problem is modeled as a MDP problem described by a continuous state space S, a discrete action space A, a reward function r, a state transition probability matrix P and a discount factor $\gamma \in [0, 1]$. For each time slot t, the state, $S_t = [Pr_t, Pr_t^{avg}, SOC_t]$, contains the information for decision making while the action, A_t is the charging or discharging operation made by the agent given the state observed. The reward, $r_t = u(S_t|A_t)$, which is explained in Section 3.1, is defined as the immediate reward resulting from taking a certain action by observing the state space. The discount factor γ is introduced to factor in the impact of current action to the performance of the subsequent time step.

In a conventional way of solving the reinforcement learning problem, a state-action table, known as Q-table, is created to store the Q-value of taking a certain decision in a certain state. The Q-table is updated using the iterative updating rule considering the recursive relationship between the Q-values. However, the state in the report is continuous and it is impossible to create a state-action table to record all the combinations of states and action. Given the limitation of Q-table, a neural network, known as Deep-Q-Network (DQN), is created which takes the states as the inputs and produces the Q-value of every action as outputs. DQN is trained via the trial and error method [30] in which a random decision is made at the initial stage of training and the Q-value gets updated subsequently using the Bellman equation as follows:

$$Q(S_t, A_t) = r_t + \gamma \times max_{A_{t+1}}(Q(S_{t+1}, A_{t+1})) \tag{13}$$

where $Q(S_t, A_t)$ is the Q-value at time step t, r_t is the reward from taking action A_t at the current state S_t and $max_{A_{t+1}}(Q(S_{t+1}, A_{t+1}))$ denotes the maximum Q-value taking optimal action at the subsequent time step which is predicted by DQN. DQN might not predict correct Q-values of every action but it is able to give better and better approximation as the training goes.

Since there are multiple available actions for decision making, an epsilon-greedy policy is deployed to achieve a good trade-off between exploration and exploitation and help the training of DQN. In exploration, DQN probes all the possible search space for finding other promising solutions that are yet to be refined; in exploitation, DQN probes the limited but promising search space for improving the existing solutions. In the report, an exploring probability is first generated at each instance of decision making for ESS operation. Should the probability is smaller than the epsilon, DQN randomly selects an action from the pools of actions; on the other hand, DQN selects the action which DQN predicts to achieve the highest utility function if the probability is larger than the epsilon. The epsilon should be adjusted after each decision making to allow DQN move from strong explorative to strong exploitative and converge eventually. However, a small epsilon should be maintained to allow minimal stochasticity and exploration which allows possible improvement of results. The epsilon is computed as follows:

$$\epsilon = \epsilon_{stp} + (\epsilon_{str} - \epsilon_{stp}) \times \exp^{-d \times s} \tag{14}$$

where ϵ, ϵ_{str} and ϵ_{stp} denote the epsilon of current instance, initial epsilon and final epsilon (minimal) while d and s denote the decay rate and total number of decision made till the current instance. The approximately correct Q-value of each action can be predicted when the DQN is trained and the agent can acts optimally by selecting the action with the highest Q-value. It should be noted the transition probability P is not required in our problem since the charging or discharging action made by the agent does not affect how the consumer load, solar power and dynamic electricity pricing changes.

3.3 Prioritised Experience Replay

DQN training is performed using the difference between prediction and the target value computed using equation (13) after each decision making. With such a training method, DQN tends to forget its previous experience due to overwriting of the new experience. To resolve the challenge, prioritized experience replay (PER) [31] is deployed in this report. In experience replay, the state, action and rewards are saved in the memory after each decision making and a batch is sampled uniformly from the memory during the training. In contrast to experience replay, PER not only stores the above-mentioned elements but also the error of each experience, which is the difference between the prediction and target, in the memory. PER places more priority on those experiences with high error which are important to the training. The priority of each experience p_{ex} is proportional to the magnitude of error err_{ex}, and a minimal constant error err, is added to the priority to ensure that no experience has zero probability to be selected:

$$p_{ex} = |err_{ex}| + err \tag{15}$$

To prevent overfitting with the same experience as per greedy prioritization, stochastic prioritization is used in which the probability of each experience $Prob_{ex}$ being selected for training, is not only affected by the priority but also the randomness hyperparameter α:

$$Prob_{ex} = \frac{p_{ex}^{\alpha}}{\sum p_{ex}^{\alpha}} \tag{16}$$

With zero value of α, the experience is selected uniformly whereas with unity value of α, the experience is selected purely based on the experience priority.

Due to the priority sampling, PER introduces bias towards high-priority samples which risks DQN being over-fitted. In order to correct the bias, the importance sampling weights are introduced to adjust the weights of the experiences w_{ex}, that are selected frequently:

$$w_{ex} = (\frac{1}{R \times Prob_{ex}})^{\beta} \tag{17}$$

where R and β denote the batch sample size and importance sampling control rate of each training respectively. The control rate affects how the batch samples affect the learning and it should be close to 0 at the beginning since the prediction of DQN might not be accurate and annealed up to 1 over the training duration when DQN starts to converge.

3.4 Procedure

The training of the reinforcement learning model can be performed as follows: 1) create an empty memory list to store the experiences (state, action, reward, priority); 2) select a pre-training period and feed the state information into DQN and select the optimal action with highest Q-value. Compute the priority and store the experience in the memory; 3) select values for ϵ, α and β; 4) for each hour, uniformly generate a random number and select the best action given the Q-value predicted by DQN if the number is smaller than ϵ else randomly select an action; 5) compute the priority and store the experience; 6) train DQN with a batch size of experience selected from memory; 7) decrease ϵ and increase β; 8) repeat step 4 to 7 until DQN converges.

4 Analysis & Comparison of Results of Algorithms

In this section, we provide simulation results of the proposed energy management algorithms: 1) model predictive control linear programming with forecast knowledge (MPCLPF), 2) risk averse reinforcement learning with no future knowledge (RARL), 3) risk seeking reinforcement learning with no future knowledge (RSRL). Besides, the simulation result of model predictive control linear program with perfect knowledge (MPCLPP) is presented to provide a guideline of the effectiveness of the proposed algorithm. The DQN used in risk averse and seeking reinforcement learning contains 1 input layer with 3 neurons taking in the state space values, 2 fully connected hidden layers with 40 and 80 neurons respectively and 1 output layer with 11 neurons predicting the Q-value of each action. ReLU and linear activation function are used for the first three layers and output layers respectively. The hyperparameters for epsilon-greedy policy and PER are tabulated in Table 1. The consumer load, solar power and dynamic electricity price are obtained from [29]. A lithium-ion battery ESS, with P_{rated} and E_{rated} at $1000kW$ and $5000kWh$ respectively, life cycle of 4996, δ of 1, and η_c and η_d of 90%, is used in the experiment. C_i is set to be $171\$/kWh$. The target SOC required for microgrid's critical operation is assumed to be 0.5 which is equivalent to 2500kWh.

Table 1: Hyperarameters of Risk Averse & Risk Seeking Reinforcement Learning Model

ϵ_{str}	ϵ_{stp}	d	err	$alpha$	β
1.0	0.01	0.0001	0.01	0.6	0.4

The simulations are performed to analyze the results of different algorithms in response to different condition and also to study the impact of MGO's behavior, namely risk-seeking and risk-adverse, on the ESS operation decisions. The performance of the proposed algorithms is also bench-marked against the model prediction control with perfect knowledge. The simulations are carried out for the entire horizon available in the data set. An overall result is presented in Section 4.1 and followed by the analysis of the result in selected unique events 4.2, 4.3 and 4.4.

4.1 Overall Result

The numerical simulation results of different algorithm, 1) the probability of SOC being lower than the target SOC (the probability of the Microgrid not having enough reserved energy for critical-mission operations) denoted as $P(SOC < SOC_{target})$ and 2) the daily average monetary benefit denoted as B_{daily}^{avg}, are compiled and tabulated in Table 2. The observations and analysis are as follows:

- $P(SOC < SOC_{target})$ are 0 for both MPCLPP and MPCLPF. It implies that the reserved energy is always kept sufficient for the critical-mission operation. It is due to the nature of the constrained linear programming which enforces the reserved energy constraint to be fulfilled for each decision making.
- $P(SOC < SOC_{target}))$ are 0.55% and 20.15% respectively for RARL and RSRL. It implies that 0.55% and 20.15% of the time, the Microgrid is having reserved energy lower than the requirement. In comparison, the Microgrid operating with RARL is more conservative than that with RSRL.
- B_{daily}^{avg} are \$70.54 and \$65.93 respectively for MPCLPP and MPCLPF. With the perfect knowledge, MPCLP can make a better decision as compared to the forecast knowledge. However, the performance is comparable in both cases.
- B_{daily}^{avg} are \$37.01 and \$77.14 respectively for RARL and RSRL. With the risk averse perspective, the Microgrid does not trade energy aggressively as compared to that with the risk seeking perspective
- In overall, MPCLPF achieves a comparable performance to MPCLPP and outperforms RARL and RSRL in term of maintaining the reliability and upholding the profitability of the Microgrid. In return, a more complex system is needed for MPCLPF since a

robust forecasting algorithm is of utmost important to ensure the performance of the energy management algorithm.

Table 2: Simulation Results Corresponding to Different Algorithms

	MPCLPP	MPCLPF	RARL	RSRL
$P(SOC < SOC_{target})$ (%)	0	0	0.55	20.15
B_{daily}^{avg} ($)	70.54	65.93	37.01	77.14

4.2 Case 1: Day with Surplus Solar Power & Spared ESS Capacity

In this case study, we present the action decided by MPCLPF, RARL and RSRL in a day with surplus solar power and spared ESS capacity. The actions taken by the respective algorithms are depicted in Figure 27, 28 and 29. The observation and analysis are as follows:

- MPCLPF is more far-sighted in making decision by observing the number of charging and discharging cycles. MPCLPF is able to identify the trough and peak with relative to the other hour and makes the best decision throughout a day. Besides, it never goes below the target SOC level.
- Both RARL and RSRL can charge and discharge at the appropriate timings. However, there is a ramp in charging and discharging power as compared to MPCLPF which is not a healthy operation habit for the machine.
- All three algorithms make the correct decision in storing the surplus solar power with the spared ESS capacity to maximize the monetary benefits.

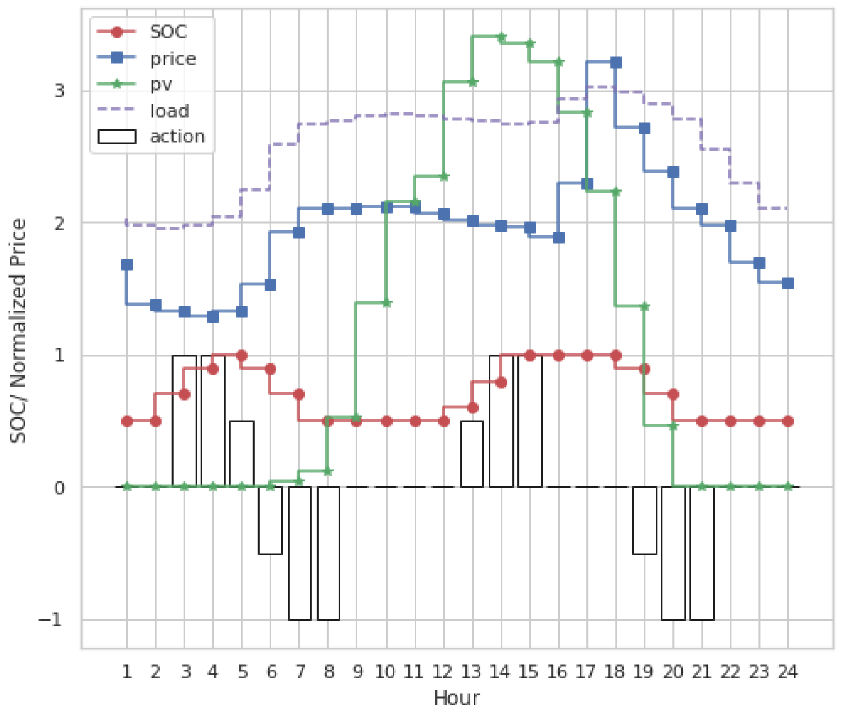


Figure 27: MPCLPF ESS actions in case 1

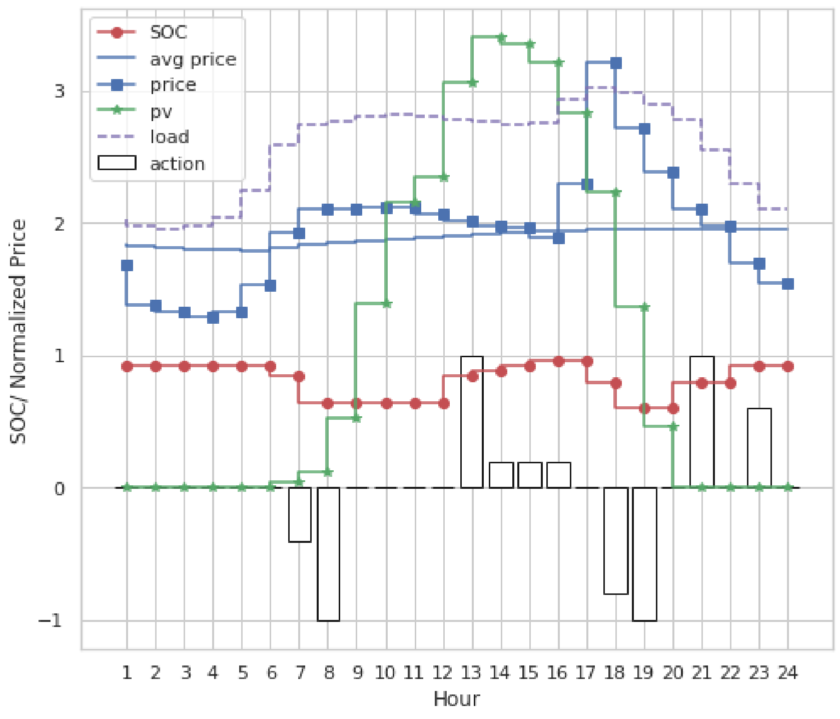


Figure 28: RARL ESS actions in case 1

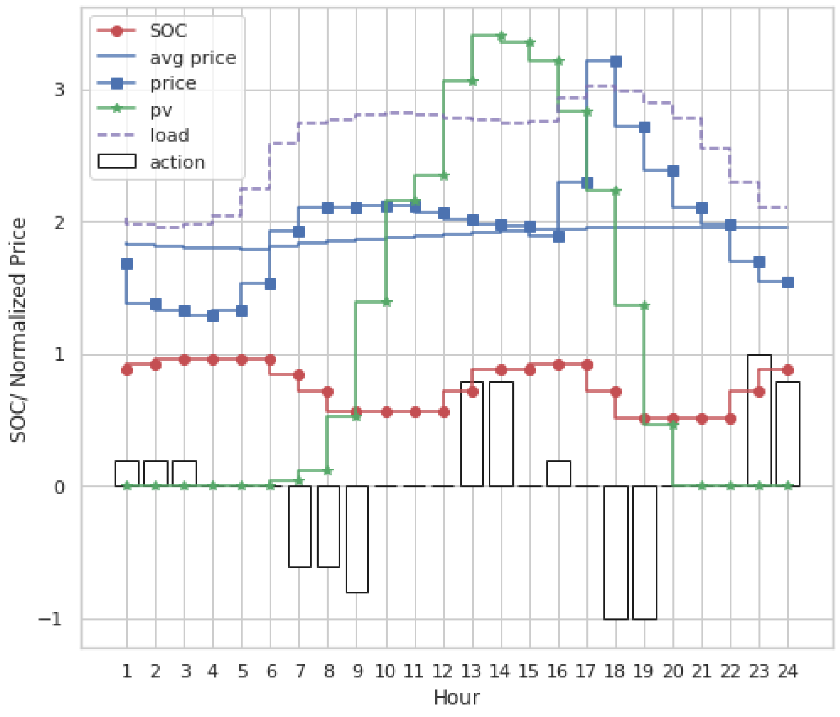


Figure 29: RSRL ESS actions in case 1

4.3 Case 2: Day with Surplus Solar Power & Full ESS Capacity

In this case study, we present the action decided by RARL and RSRL in a day with surplus solar power and with full ESS capacity. The actions taken by the respective algorithms are depicted in Figure 30 and 31. The observation and analysis are as follows:

- Both RARL and RSRL do not charge when the ESS capacity is full during the hour with surplus solar power. Since the utiltiy function of reinforcement learning is designed such that the penalty for violating ESS operation limit is larger than the penalty of not storing surplus solar power, RARL and RSRL do the correct decision and allows the surplus solar power to be trade with the main grid to maintain the power balance in the Microgrid.

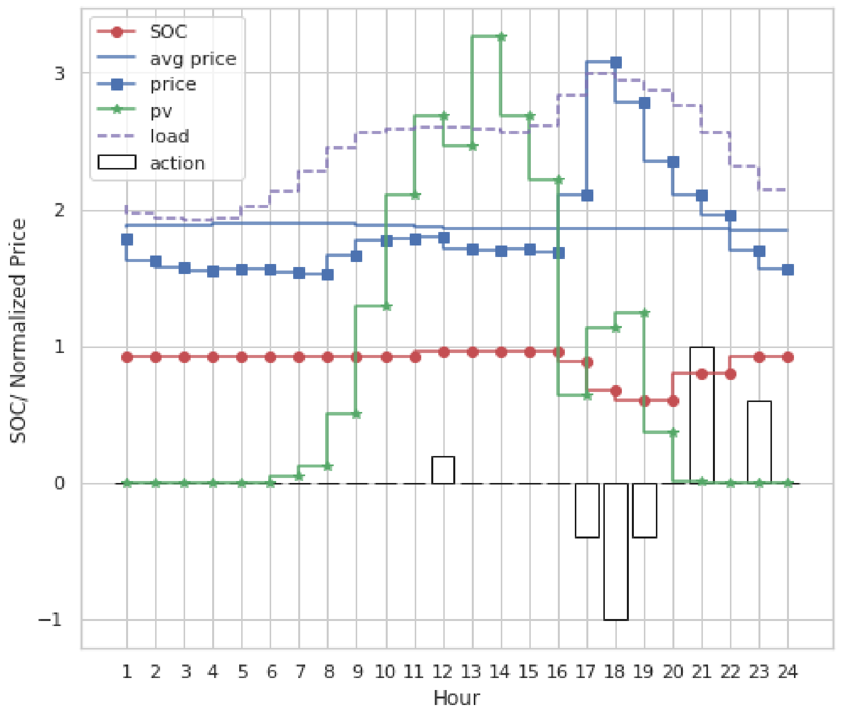


Figure 30: RARL ESS actions in case 2

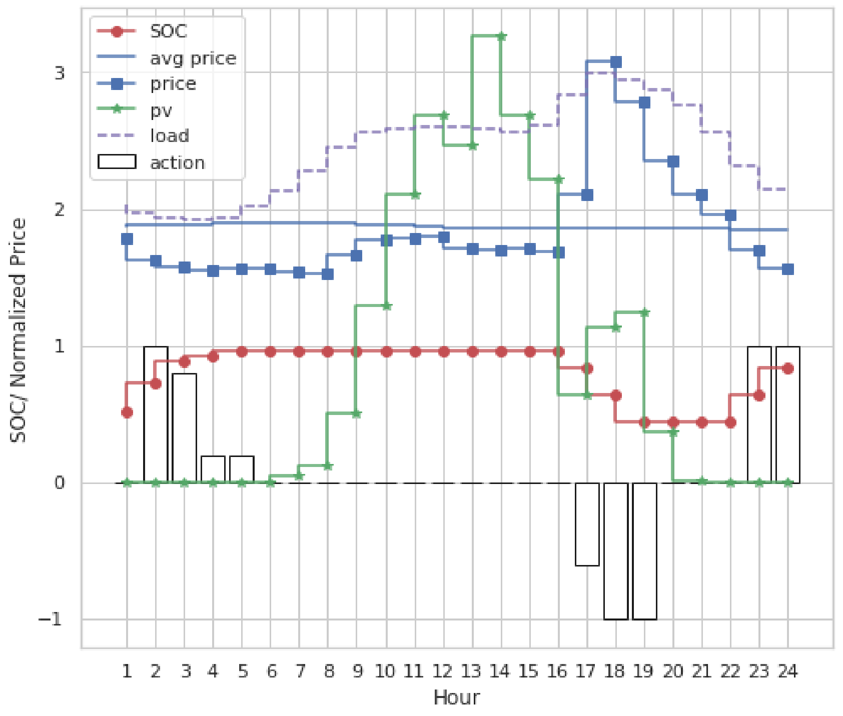


Figure 31: RSRL ESS actions in case 2

4.4 Case 3: Day with Extremely High Electricity Price

In this case study, we present the action decided by MPCLPF, RARL and RSRL in a day with extremely high electricity price. The actions taken by the respective algorithms are depicted in Figure 32, 34 and 36. The zoom in SOC for the respective algorithms are also depicted in Figure 33, 35 and 37. The The observation and analysis are as follows:

- MPCLPF can identify the hours with the highest electricity price and only discharge during those hours to maximize the possible monetary return. MPCLPF operates strictly above the target SOC even when the electricity price is extremely high and presents a good oppurnunity to take advantage of the rare event.
- RARL and RSRL are more flexible in handling the extreme scenario with the high dynamic electricity price. However, it should be noted that the higher monetary benefit is achieved with the trade off of more risk borne by the MGO and more loss in the event of power failure of main-grid. Both ESS goes below the target SOC during the hours with extremely high electricity price to maximize the monetary benefit.
- The case study proves the effectiveness of the utility function of reinforcement learning. Recalls that the piecewise utility function is designed to better utilize the remaining energy during discharging and the remaining spare capacity during charging, ESS discharges (charges) in a more meticulous way when the ESS capacity is lower during the hour $11th$ to $17th$ (the ESS capacity is higher during the hour $18th$ to $24th$). Specifically, the lower (higher) the SOC is, the higher (lower) the dynamic pricing is needed to discharge (charge). However, it should be noted that the dynamic pricing when ESS is discharged (charged) might not be the highest (lowest) throughout the day but it is high (low) enough for the algorithms to realize that such action taken is a near-optimal solution.

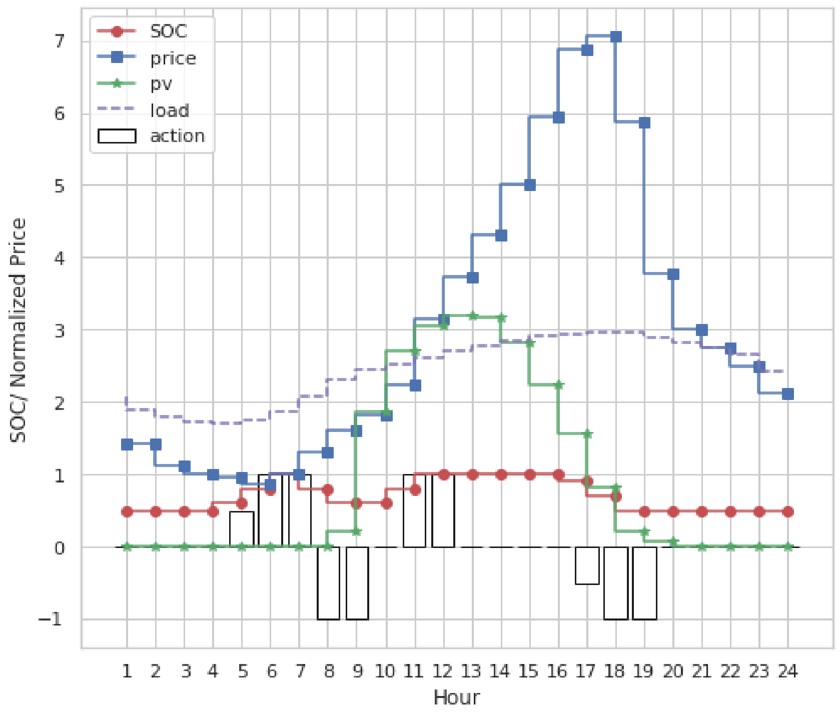


Figure 32: MPCLPF ESS actions in case 3

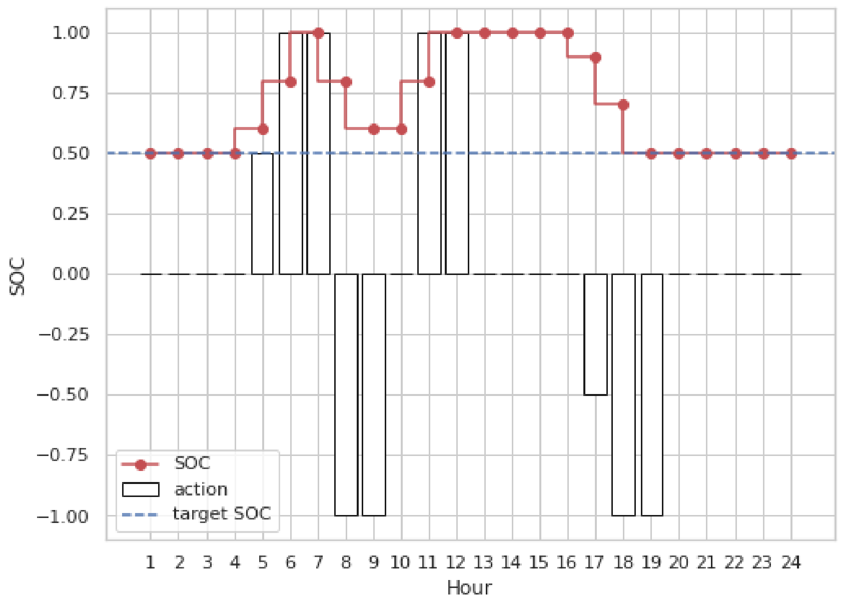


Figure 33: MPCLPF zoom in SOC in case 3

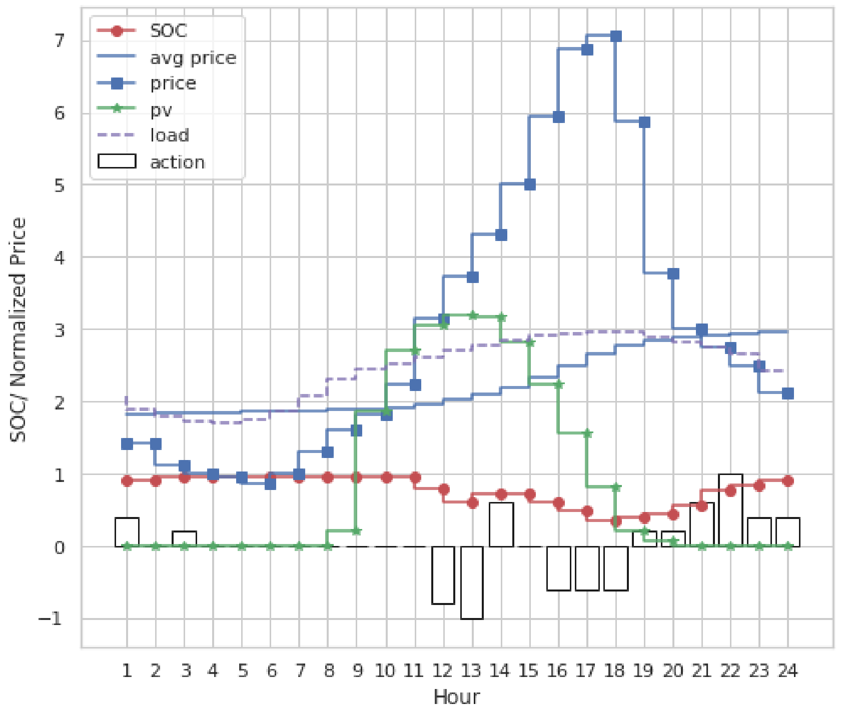


Figure 34: RARL ESS actions in case 3

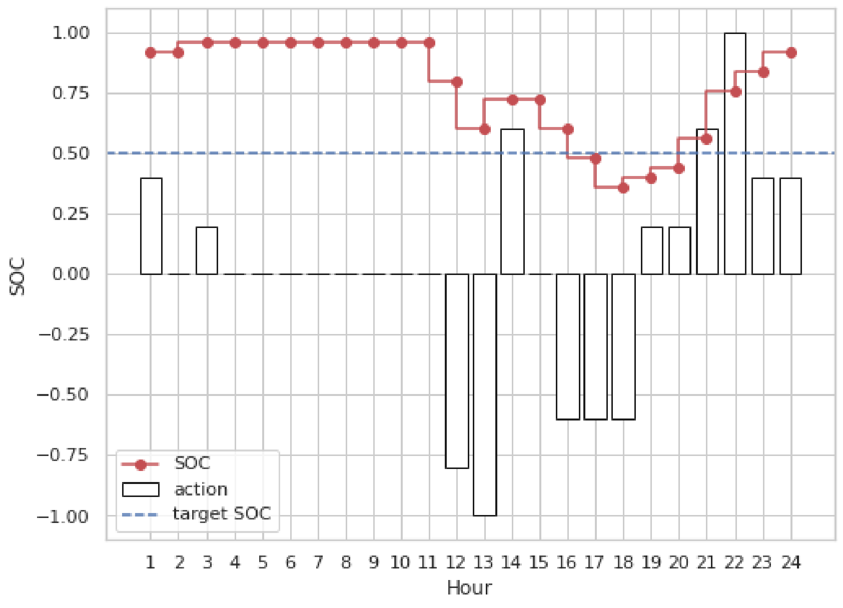


Figure 35: RARL zoom in SOC in case 3

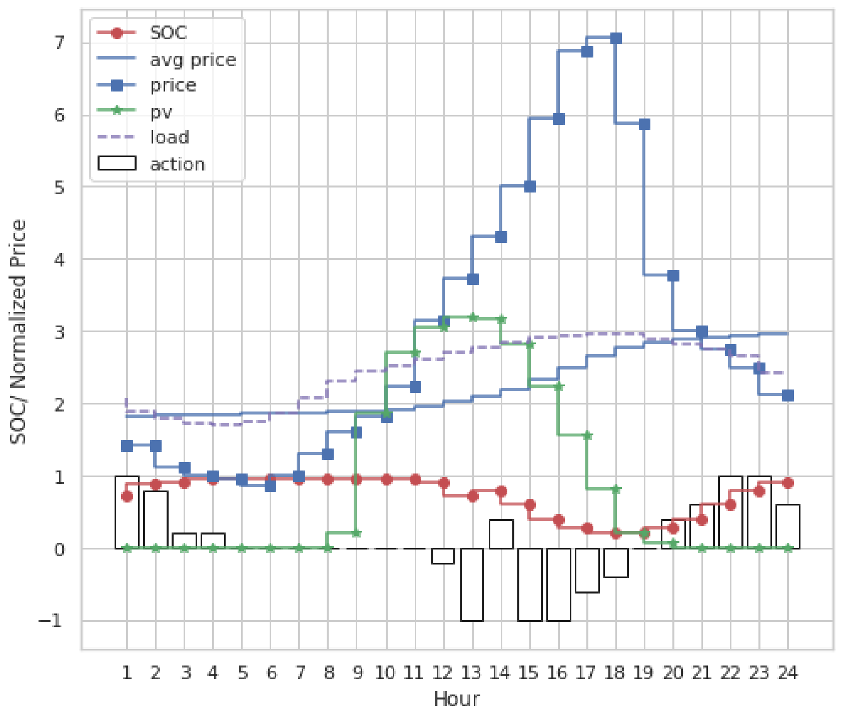


Figure 36: RSRL ESS actions in case 3

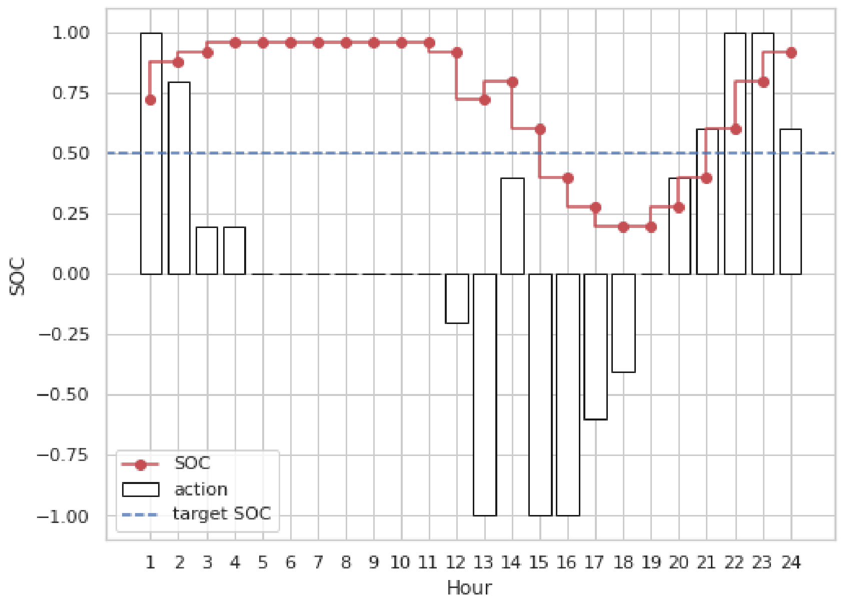
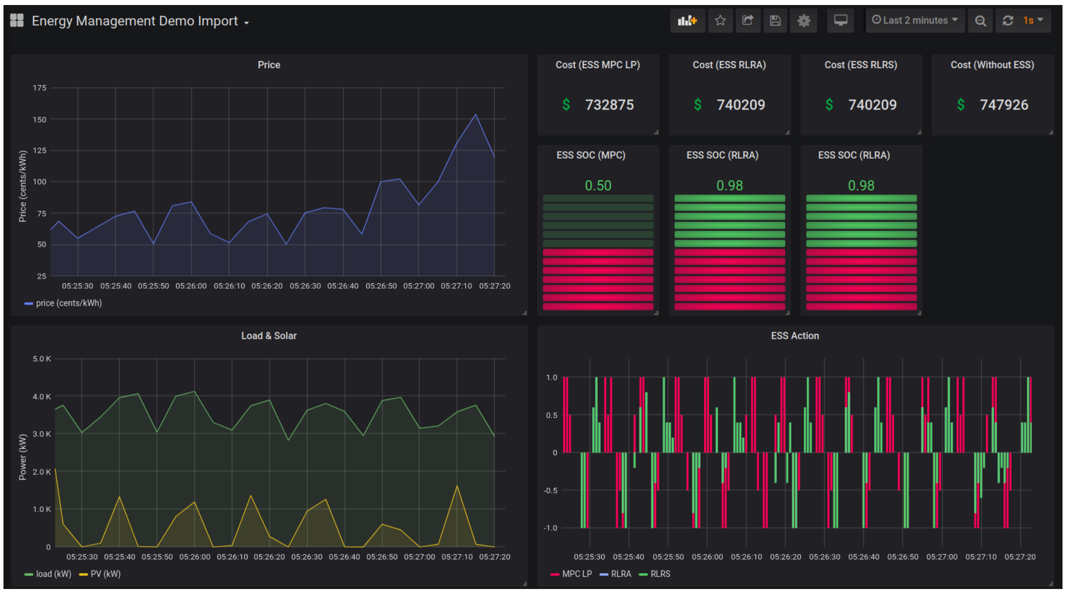


Figure 37: RSRL zoom in SOC in case 3

5 Graphical User Interface

In this section, we demonstrate the deployment of the algorithms and create a graphical user interface to provide any user with good visualization of the algorithm's decision and its resultant effect. The graphical user interface is an integration of software: 1) InfluxDB is an open-source time-series database developed by InfluxData which provides high-availability storage and retrieval of time series data in fields such as operation monitoring, real-time analytics, etc. [32]; 2) Grafana is a multi-platform open-source interactive visualization web application which provides charts, graph, etc. when connected to data sources [33]. The graphical user interface works in the following sequential manners:

- **Step 1:** the timestamped real-time data from the sensors is pushed into the InfluxDB;
- **Step 2:** the energy management algorithms retrieve the necessary data from InfluxDB and make decisions based on the data observed;
- **Step 3:** the decisions and its effects on the ESS SOC and the accumulated monetary benefit are computed and stored in InfluxDB;
- **Step 4:** the Grafana queries the data from the InfluxDB and displays in graph representation as depicted in Figure 38.



Cost (ESS MPC LP) $ 732875
Cost (ESS RLRA) $ 740209
Cost (ESS RLRS) $ 740209
Cost (Without ESS) $ 747926
ESS SOC (MPC) 0.50
ESS SOC (RLRA) 0.98
ESS SOC (RLRA) 0.98

Figure 38: Graphical user interface for energy management algorithm

6 Conclusion

In this report, we present two energy management algorithms suitable for microgrid to automate energy transaction with main grid and develop a graphical user interface for visualization. The algorithms not only maximize the monetary benefit but also maintains microgrid reliability by maintaining sufficient amount of energy reserve for critical-mission operations. The first energy management algorithm introduced is MPC linear programming (MPCLC) which integrates MPC with linear programming and different optimal forecasting algorithms for prediction of future dynamic electricity price, consumer load and solar power. The MPCLC achieves the best performances with daily average monetary benefit of \$65.74 and always maintains the reserved energy higher than the required contingency reserve. The second energy management algorithm introduced is reinforcement learning (RL) which only requires the current information to optimize the energy transaction decision. The reward function of reinforcement learning is designed such that the MGO can be risk-averse (RA) or risk-seeking (RS). For RARL, the daily average monetary benefit is \$37.01 while there is 0.55% of the time the reserved energy is lower than the contingency reserve; on the other hand for RSRL, the daily average monetary benefit is \$77.14 while there is 20.55% of the time the reserved energy is lower than the contingency reserve. It indicates that MGO can trade off the system reliability with higher monetary benefit.

The MPCLC achieves the best performance considering that it achieves a comparable daily monetary benefit as per the risk-seeking reinforcement learning but with higher system reliability. However, we acknowledge that the utility function of RL could be modified and fine-tuned to achieve a better result. In addition, the transaction decision produced by the MPC linear programming is continuous while that of by the reinforcement learning is discrete. It definitely contributes to a certain degree of impact to the ultimate result. In the future work, a policy based reinforcement learning can be studied and replace the current value based reinforcement learning to obtain the continuous decision. Besides, the current work only focus on microgrid to main grid which can be extended further to microgrid to microgrid to analyze how microgrid can interact with one another.

7 Appendix

The relevant Python scripts for respective algorithms are accessible with the following link:

- **Forecasting models:** `https://github.com/ChongAih/Energy-Management-and-Economic-Evaluation-of-Grid-Conected-Microgrid-Operation/tree/master/Comparison%20of%20forecasting%20technique%20(Load%2C%20PV%2C%20Price)/Phase%201%20-%20Comparison%20of%20forecasting%20technique`
- **Model predictive control linear programming with forecast future knowledge:** `https://github.com/ChongAih/Energy-Management-and-Economic-Evaluation-of-Grid-Conected-Microgrid-Operation/tree/master/Energy%20Transaction%20Using%20Model%20Prediction%20Control%20Linear%20programming`
- **Reinforcement learning with no future knowledge:** `https://github.com/ChongAih/Energy-Management-and-Economic-Evaluation-of-Grid-Conected-Microgrid-Operation/tree/master/Energy%20Transaction%20Using%20Reinforement%20Learning%20Without%20Future%20Knowledge`
- **Graphical user interface:** `https://github.com/ChongAih/Energy-Management-and-Economic-Evaluation-of-Grid-Conected-Microgrid-Operation/tree/master/Model%20deployment%20and%20visualization`

References

[1] M. F. Zia, E. Elbouchikhi, and M. Benbouzid, “Microgrids energy management systems: A critical review on methods, solutions, and prospects,” *Applied Energy*, vol. 222, pp. 1033 – 1055, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261918306676

[2] Attention in rnns. https://medium.com/datadriveninvestor/attention-in-rnns-321fbcd64f05.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[4] How do transformers work in nlp? a guide to the latest state-of-the-art models. https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models/.

[5] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y. Wang, and X. Yan, “Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting,” *CoRR*, vol. abs/1907.00235, 2019. [Online]. Available: http://arxiv.org/abs/1907.00235

[6] J. Wu, J. Yan, H. Jia, N. Hatziargyriou, N. Djilali, and H. Sun, “Integrated energy systems,” *Applied Energy*, vol. 167, pp. 155 – 157, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261916302124

[7] H. N, *Microgrids: Architectures and Control*, 2007.

[8] G. Comodi, A. Giantomassi, M. Severini, S. Squartini, F. Ferracuti, A. Fonti, D. Nardi Cesarini, M. Morodo, and F. Polonara, “Multi-apartment residential microgrid with electrical and thermal storage devices: Experimental analysis and simulation of energy management strategies,” *Applied Energy*, vol. 137, pp. 854 – 866, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S030626191400751X

[9] J. Shen, C. Jiang, Y. Liu, and J. Qian, “A microgrid energy management system with demand response for providing grid peak shaving,” *Electric Power Components and Systems*, vol. 44, no. 8, pp. 843–852, 2016. [Online]. Available: https://doi.org/10.1080/15325008.2016.1138344

[10] A. C. Luna, L. Meng, N. L. Diaz, M. Graells, J. C. Vasquez, and J. M. Guerrero, “Online energy management systems for microgrids: Experimental validation and assessment framework,” *IEEE Transactions on Power Electronics*, vol. 33, no. 3, pp. 2201–2215, 2018.

[11] H. Kanchev, D. Lu, F. Colas, V. Lazarov, and B. Francois, “Energy management and operational planning of a microgrid with a pv-based active generator for smart grid applications,” *IEEE Transactions on Industrial Electronics*, vol. 58, no. 10, pp. 4583–4592, 2011.

[12] A. Chaouachi, R. M. Kamel, R. Andoulsi, and K. Nagasaka, “Multiobjective intelligent energy management for a microgrid,” *IEEE Transactions on Industrial Electronics*, vol. 60, no. 4, pp. 1688–1699, 2013.

[13] T. Wang, X. He, and T. Deng, “Neural networks for power management optimal strategy in hybrid microgrid,” *Neural Computing and Applications*, vol. 31, no. 7, pp. 2635–2647, Jul 2019. [Online]. Available: https://doi.org/10.1007/s00521-017-3219-x

[14] M. E. Gamez Urias, E. N. Sanchez, and L. J. Ricalde, “Electrical microgrid optimization via a new recurrent neural network,” *IEEE Systems Journal*, vol. 9, no. 3, pp. 945–953, 2015.

[15] E. Kuznetsova, Y.-F. Li, C. Ruiz, E. Zio, G. Ault, and K. Bell, "Reinforcement learning for microgrid energy management," *Energy*, vol. 59, pp. 133 – 146, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360544213004817

[16] T. Bogaraj and J. Kanakaraj, "Intelligent energy management control for independent microgrid," *Sādhanā*, vol. 41, no. 7, pp. 755–769, Jul 2016. [Online]. Available: https://doi.org/10.1007/s12046-016-0515-6

[17] C. Dou and B. Liu, "Multi-agent based hierarchical hybrid control for smart microgrid," *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 771–778, 2013.

[18] A. Ghasemi, "Coordination of pumped-storage unit and irrigation system with intermittent wind generation for intelligent energy management of an agricultural microgrid," *Energy*, vol. 142, pp. 1 – 13, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360544217316675

[19] G. Cau, D. Cocco, M. Petrollese, S. Knudsen Kær, and C. Milan, "Energy management strategy based on short-term generation scheduling for a renewable microgrid using a hydrogen storage system," *Energy Conversion and Management*, vol. 87, pp. 820 – 831, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0196890414007171

[20] B. V. Solanki, K. Bhattacharya, and C. A. Cañizares, "A sustainable energy management system for isolated microgrids," *IEEE Transactions on Sustainable Energy*, vol. 8, no. 4, pp. 1507–1517, 2017.

[21] B. V. Solanki, A. Raghurajan, K. Bhattacharya, and C. A. Cañizares, "Including smart loads for optimal demand response in integrated energy management systems for isolated microgrids," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1739–1748, 2017.

[22] K. Rahbar, J. Xu, and R. Zhang, "Real-time energy storage management for renewable integration in microgrid: An off-line optimization approach," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 124–134, Jan 2015.

[23] T. Chen and W. Su, "Local energy trading behavior modeling with deep reinforcement learning," *IEEE Access*, vol. 6, pp. 62 806–62 814, 2018.

[24] Lecture 14 - model predictive control part 1: The concept. https://web.stanford.edu/class/archive/ee/ee392m/ee392m.1056/Lecture14_MPC.pdf.

[25] Introduction to linear programming. https://web.stanford.edu/group/sisl/k12/optimization/MO-unit3-pdfs/3.4buildingsimplex.pdf.

[26] Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, p. 3104–3112.

[28] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015.

[29] T. Hong, P. Pinson, S. Fan, H. Zareipour, A. Troccoli, and R. Hyndman, "Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond," *International Journal of Forecasting*, vol. 32, no. 3, pp. 896–913, 2016. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:32:y:2016:i:3:p:896-913

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013. [Online]. Available: http://arxiv.org/abs/1312.5602

[31] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay,” *CoRR*, vol. abs/1511.05952, 2015.

[32] Influxdb: Purposely-built time series database. https://www.influxdata.com/.

[33] Grafana: The open observability platform. https://grafana.com/.