State of Energy Prediction in Renewable Energy-driven Mobile Edge Computing using CNN-LSTM Networks

Yu-Jen Ku, Sandalika Sapra, Sabur Baidya, Sujit Dey
Department of Electrical and Computer Engineering, University of California, San Diego
e-mail: {yuku, ssapra, sbaidya, dey}@ucsd.edu

Abstract—**Renewable energy (RE) is a promising solution to save grid power in mobile edge computing (MEC) systems and thus reducing the carbon footprints. However, to effectively operate the RE-based MEC system, a method for predicting the state of energy (SoE) in the battery is essential, not only to prevent the battery from over-charging or over-discharging, but also allowing the MEC applications to adjust their loads in advance based on the energy availability. In this work, we consider RE-powered MEC systems at the Road-side Unit (RSU) and focus on predicting its battery's SoE by using machine learning technique. We developed a real-world RE-powered RSU testbed consisting of edge computing devices, small cell base station, and solar as well as wind power generators. By operating RE-powered RSU for serving real-world computation task offloading demands, we collect the corresponding data sequences of battery's SoE and other observable parameters of the MEC systems that impact the SoE. Using a variant of Long Short-term Memory (LSTM) model with additional convolutional layers, we form a CNN-LSTM model which can predict the SoE accurately with very low prediction error. Our results show that CNN-LSTM outperforms other Recurrent Neural Networks (RNN) based models for predicting intra-hour and hour-ahead SoE.**

I. Introduction

In the era of the Internet-of-Things (IoT), the number of connected devices, performing multi-scale computations to support smart applications, is exponentially increasing. Additionally, the computing tasks become more complex and artificial intelligence (AI) driven, thus consuming more system resources and more energy. However, some of the end devices which have constraints in their computation resources or energy availability, will choose to offload their computations to a more powerful computing servers, e.g. mobile edge computing (MEC) [1] or cloud computing [2] servers. Here we focus on the real-time systems, where computations have low latency requirements. MEC server being one-hop wireless distance away from the users, incurs low round-trip delay, fitting well for real-time applications. However, when several users offload their computations to the edge server, it increases the energy consumption at MEC by a large extent due to high computations, and radio communications through the base station which the MEC server is connected with. Consuming this energy from grid power causes an enormous increase in carbon footprint which is adverse for the environment.

To mitigate this problem, adopting technologies of renewable energy (RE) and storage are increasingly becoming a viable solution to supply power to the MEC systems. In this paper, we investigate a more challenging situation where the MEC system is solely powered by RE. However, effectively implementing this RE-based MEC system needs prediction of the future remaining energy of the battery, i.e. state of energy (SoE) [3], to plan in advance for scheduling alternate energy sources or mode of operations in order to continue the applications or to avoid from over-charging and over-discharging the battery. In this work, we consider SoE instead of state of charge (SoC), which lacks considerations on battery's temperature and voltage [4], because both the battery's temperature and voltage fluctuate with the weather condition and RE generation.

Predicting future SoE at the network edge is nontrivial as it comprises of complex interplay among renewable energy generation patterns, together with varying energy consumption for computation of the applications, and also due to data communications at the base stations. Our goal is to predict the future SoE at the RE-based MEC system, when it is generating RE using solar and wind power, and simultaneously serving multiple applications through edge computing over wireless networks, in terms of observable parameters such as RE generation, environmental parameters (e.g. Temperature, Humidity), MEC's communication as well as computation loads. We aim at predicting remaining SoE without measuring power consumption of the MEC system, so that our method can be applied to any MEC devices which are not capable of measuring its own power consumption.

In this paper, we consider a use case scenario, where MEC systems are used for vehicular real-time applications and mounted at the road side units (RSU). The RSUs also host the small cell base stations that communicate with the vehicle users (VUs) over cellular network. The RSU is fully RE-powered, which we will refer to as RE-powered RSU in the following paragraphs, that supplies energy to the small cell base station and the MEC server. VUs offload some of their tasks to the MEC server thus creating high computation and communication loads at the RSU. We incorporate a machine learning approach based on data collected with a real-world testbed for generating solar and wind power, connected with software-defined radio based small cell for communications, and MEC devices. We predict the future SoE with respect to dynamic vehicular loads for a traffic scenario based on real-world vehicular traces. We particularly predict intra-hour forecast (e.g. a few hours ahead) for the SoE. The contributions of our paper are the following:

- A real-world testbed including solar and wind power generation, together with edge computing devices and small cell cellular base station.
- A Machine learning based prediction model for the future SoE of the system based on the multi-source green energy generations, and simultaneous computing and communication loads that use the generated energy. The feature set of the model consists of only real-time observable parameters that impact the decision making for consumer applications.
- We show how the historical and the future window sizes impact the prediction of intra-hour and hour-ahead SoE.

We present the detailed description of our testbed and demonstrate the advantage of our prediction model through numerical evaluations.

II. Related Work

Predicting SoE or SoC using deep learning techniques has been widely studied by researchers. However, most studies predict the current SoC of the battery to address the difficulty of accurately estimating SoC using available data measurement. For example, [5] and [6] estimate battery SoC from immediate current, voltage, and temperature measurements using long short-term memory network techniques. In our work, we are predicting the battery's remaining energy in the future, which involves the use of knowledge of future energy load requirement and power generation of the system.

Predicting remaining battery energy for the future has been used in various areas. For example, in [7], the authors establish power consumption models for different components in Electric Vehicles (EV). By using the models to predict EV's total future power consumption for a selected route, [7] predicts the remaining SoC at the destination. On the other hand, [8] characterizes a quadratic model for the SoE in the battery versus operation time for a sensor node in a wireless sensor network. With the predicted remaining energy of a battery, these studies can improve the management and operation of the systems in consideration. These works are focusing on modeling the future's discharging rate or power requirement or remaining battery energy prediction. Therefore, these approaches can not be applied in the scenario considered in this work, as the battery is being charged by RE and discharged by RSU at the same time. To the best of our knowledge, there has not been any study that considered predicting the future SoE of battery for an RE-powered RSU.

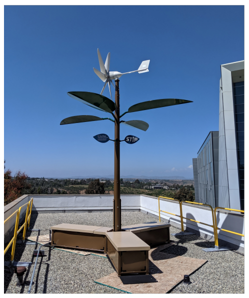
III. System setup

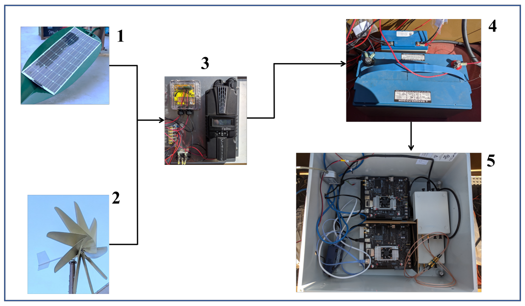
In this work, we focus on the object detection for a dash-cam as a vehicular real-time applications, however, our work can be easily extended to other types of MEC-based applications as well. Due to the potentially limited computation resource of the VU, it can choose to offload the object detection task to the edge computing server mounted at the RE-powered RSU for shorter end-to-end delay performance or saving its energy consumption [9]. We assume the edge computing server is already capable of providing object detection services. Therefore, to offload the task, VU only needs to transmit the images to the edge computing server. After the edge computing server executes the object detection task, RSU transmits the result back to the corresponding VU.

To the best of our knowledge, there is no open access dataset which includes the time series data of RE generation and computation as well as communication loads of an RE-powered RSU providing such offloading capability to VUs. Therefore, in this work, we build a testbed to emulate the functions of a RE-powered RSU at the University of California, San Diego, CA, USA. The external appearance of this testbed is shown in Fig. 1a. We will introduce the different modules of the testbed in the following subsections.

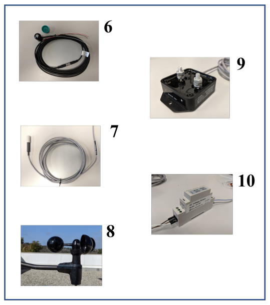
A. *Renewable Energy Harvesting Module*

This testbed is fully powered by the RE harvested from EnergiPlant, a product provided by PrimoEnergy, Inc., which consists of four solar panels and one wind turbine. Each solar panel (i.e. component 1 in Fig. 1b) has length 1.2 m and width 0.55 m, and is able to


(a)



(b)



(c)

Fig. 1: The RE-powered RSU testbed with (a) left, the external look of the testbed, (b) center, the RE generators, batteries, RSU, and energy flow in the testbed, and (c) right, the sensors in the testbed

produce 100 W of energy in maximum. The diameter of the wind turbine (i.e. component 2 in Fig. 1b) is 1.65 m. The rated power of the wind turbine is 1 kW at 350 rpm and the cut in wind speed is 8 mph. The solar and wind generators are connected to the charge controllers (i.e. component 3 in Fig. 1b). Note that in Fig. 1b, the black arrows represent the energy flow in the testbed.

B. RSU Module

The RSU module is shown in component 5 of Fig. 1b and its architecture is depicted in detail in Fig. 2. We emulate the above computation task offloading activity between the RE-powered RSU and VUs by using a setup with three NV Jetson TX2 [10] boards and two NI USRP B210 radios [11]. First, at the VU side, we use an USRP B210, which is controlled by an NV Jetson TX2 board (i.e. Jetson A in Fig. 2), to generate the uplink communication load. On the other hand, we use two NV Jetson TX 2 boards and one USRP B210 to emulate the RSU. At the RSU, we use one of the two NV Jetson TX2 boards (i.e. Jetson B in Fig. 2) and USRP B210 for receiving the uplink communication load and generating the downlink communication load. Note that we use the LTE protocol with the srsLTE software [12], for the communication between the two USRP B200s and the downlink and uplink communication loads are generated by using iperf. For the edge server, we use another NV Jetson TX2 board (i.e. Jetson C in Fig. 2) for executing the object detection tasks. The object detection task used in this work is YOLOv3 [13].

C. Sensors

We have installed solar irradiance (i.e. component 6 in Fig. 1c), temperature (i.e. component 7 in Fig. 1c), humidity (i.e. component 7 in Fig. 1c), and wind speed (i.e. component 8 in Fig. 1c) sensors to capture the environment factors. On the other hand, we measure the power consumed by RE-powered

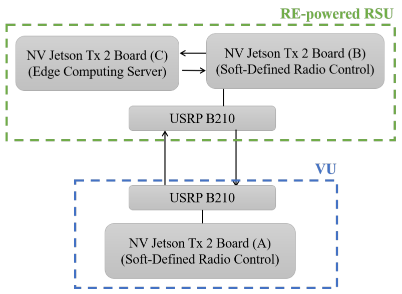


Fig. 2: The architecture of the RE-powered RSU

RSU and the power generated by solar panels and wind turbines using current sensors (i.e. component 9 in Fig. 1c) and voltage sensors (i.e. component 10 in Fig. 1c). The power values are derived by multiplying the current value to the voltage value of every sample point.

D. Battery Module

There are two batteries in this testbed (i.e. component 4 in Fig. 1b). Both of them are 12 V Lead-Acid batteries. We use a 20 Ah Lead-Acid battery to power the RSU module. Note that in the following sections, we are going to predict the SoE of this battery. On the other hand, we installed another 120 Ah Lead-Acid battery to power the sensors and charge controllers, so that this testbed can be solely powered by RE. Both batteries are charged by EnergiPlant, the RE harvesting module. We also implemented a control unit using Raspberry Pi 3 and 5V relays to direct the energy of solar or wind generators to the selected battery, which can be programmed remotely.

E. Data Collection

To establish a dataset to be used in this work, we leverage the above testbed and repeat the following experiment on different days. At the beginning of each experiment, we connect the 20 Ah battery to 2 solar panels and the wind generator. For every minute, we first generate the value of the number of vehicles in the coverage area of this RE-powered RSU by using our MATLAB-based vehicular trace simulator developed in [14]. The simulator generates the current location and speed for each vehicle by using the historical road traffic data of a neighborhood in Brooklyn, New York City, which is collected by New York State Department of Transportation [15]. Among the generated vehicles, we assume that 75% of vehicles are transmitting 720p JPEG images and the rest are transmitting 1080p JPEG images.

Then, we calculate the corresponding uplink and downlink traffic loads, and emulate the data traffic generation over the USRP radios. We also create the same number of YOLOv3 instances as the number of vehicles connected to the edge server. The value of the number of YOLOv3 instances, and uplink as well as downlink traffic loads will update every 1 minute to reflect the realistic vehicular traffic pattern in an urban area.

An experiment is terminated when the battery's voltage is lower than $10V$, which we define as 0% SoE, i.e., there is no energy stored in the battery. From 0% SoE, we trace back the stored energy of each sample by adding the current power generated and deducting the current power consumed from the SoE of the next sample. The equation is as follows,

$$E^i = E^{i-1} + P^i_{gen} * t - P^i_{con} * t \qquad (1)$$

where E^i is the stored energy of the battery, P^i_{gen} is the observed RE power generation, and P^i_{gen} is the observed RSU power consumption at the i_{th} sample point. t is the duration between two consecutive sample points. Note that $E^I = 0$, where I is the index of the last sample of this experiment, namely when the experiment is terminated. On the other hand, we derive the maximum amount of energy that can be stored in this battery is 297356 Joules (J) by starting an experiment with the battery being fully charged. The SoE S^i of each sample i is calculated by,

$$S^i = \frac{E^i}{297356\text{(J)}} * 100\% \qquad (2)$$

During each experiment, for every 2 seconds, we collect the following data as one sample: battery voltage, power generation of solar as well as wind, RSU's power consumption (which will not be used for prediction, but for calculating SoE), solar irradiance, temperature, humidity, wind speed, uplink and downlink data rate, and RSU's Jetsons' CPU loads. Note that the power generation and consumption data are derived from the readings of the corresponding current and voltage sensors in the testbed. In this work, we conducted the above experiment for 7 different days between Dec. 11th 2019 to Jan. 8th 2020 and collected more than 60000 samples.

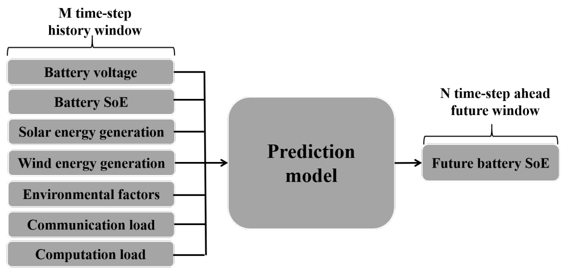


Fig. 3: The overview of the proposed battery SoE prediction mechanism

IV. Model Architecture

Fig. 3 shows an overview of the prediction method in this work. The prediction model will use M time-step history window of battery voltage, battery SoE, solar and wind energy generation, communication and computation loads, as well as environmental factors including outdoor temperature, humidity, solar irradiance, and wind speed. With the above features with history window as input, the prediction model gives the N time-step ahead SoE of the battery as output. Note that although the testbed is capable of measuring the power consumption of the RE-powered RSU, our prediction method doesn't require the knowledge of the RSU's power consumption. Therefore, the proposed prediction method can be applied to other MEC devices which are not capable of measuring its own power consumption.

The prediction method in Fig. 3 can be treated as a multivariate time series forecasting problem and it is predicting multiple time-steps ahead. CNN-LSTM model [16] is proposed to combine effects and advantages of Convolutional Neural Networks (CNN) [17] and Long Short Term Memory (LSTM) [18] models for such multivariate and multiple time-steps ahead time series prediction. Thus, in this work, we investigate the prediction performance of CNN-LSTM on future SoE prediction.

A. *Convolutional Neural Networks (CNN)*

CNN have various applications in image, video and language processing tasks in deep learning, introduced in [17]. Convolution layers, activation and pooling layers make up the architecture of CNNs. Kernel windows slide over input data features performing the convolution function to output resultant feature maps. Each kernel can be thought of as a feature identifier. Trained kernel coefficients, thus, represent these locally present features in the data. A number of kernels are stacked together to increase the depth of the produced feature map. Each kernel produces a feature map of its own, learning some spatially invariant feature in the data. The different stacked kernels in fact learn different features from the same input. The feature maps are passed through activation layers that introduce non-linearity. The non-linear output is then pooled. Pooling layers reduce spatial dimensions while ensuring the network learns features invariant to translation. Pooling clubs together input values to perform a function in order to reduce the size of the feature map. For instance, the result of a max pool operation on an input is the maximum value over a window of values. This maximum would not change if the operation is conducted at a variant location. Pooling also increases robustness to smaller variations.

B. *Long Short Term Memory (LSTM)*

As a revolutionized type of Recurrent Neural Networks (RNN), which are prone to vanishing gradients when solving for long term dependencies in sequences, LSTM units are capable of discovering and learning long-term dependencies. They are architectures that allow networks to remember and hold onto a memory indefinitely by introducing memory cells with input, output, and forget gates on top of the cell state (c_t) of an RNN, introducing also a set of weight matrices for each. The forget gate f_t, a sigmoid output of product of forget gate matrices $[W_f, U_f]$, and input x_t and hidden state h_t, can be used to decide what information to throw away from the cell state. The input gate layer i_t decides which values in the cell state are to be updated.The output gate layer o_t decides which parts of the cell state will be output to the next LSTM layer, or as the answer.

$$f_t = \sigma(W_f * x_t + U_f * h_{t-1} + b_f) \tag{3}$$

$$i_t = \sigma(W_i * x_t + U_i * h_{t-1} + b_i) \tag{4}$$

$$o_t = \sigma(W_o * x_t + U_o * h_{t-1} + b_o) \tag{5}$$

$$c'_t = \sigma(W_c * x_t + U_c * h_{t-1} + b_c) \tag{6}$$

$$c_t = f_t.c_{t-1} + i_t.c'_t \tag{7}$$

$$h_t = o_t.tanh(c_t) \tag{8}$$

LSTMs can maintain a controlled flow of error and information using these gates [18].

C. *CNN-LSTM Neural Network*

In this section, we introduce the CNN-LSTM neural network model implemented for predicting SoE of the RE-powered RSU. Fig. 4 shows an overview of the CNN-LSTM model used in this work. Convolution layers, together with pooling layers, capture local dependencies and statistical invariance in data features. In particular, a 1D convolution layer with 16 filters of kernel size 2 is used on the input data to capture time step-wise information from the multiple input features, and understand local dependencies and invariance across the features in every timed sample. The 1D convolution layer uses tanh activation and Xavier uniform weight initialization. Max pooling, with a kernel size of 2, extracts invariant attributes along every 2 time-steps (= 4s). The output is then fed to an LSTM network with 450 hidden units using tanh. The weights of the matrices for all gates in the LSTM are initialized using Xavier uniform initialization. Each feature, going into the LSTM has a dimension of 16, from the 16 convolution filters. The LSTM treats the time series as a sequence of the 16 dimensional feature vectors. A layer with a dropout probability of 0.1 is introduced after the LSTM output for regularization. The output from the dropout layer is fed to a fully connected layer to output the forecast prediction.

Note that, compared to YOLOv3 application, which consists of 24 CNN layers, the complexity of the above SoE prediction model is very low. Also, the power consumed for the above model will not change over time. Moreover, the frequency of performing SoE prediction is usually much lower, e.g., once in a few seconds at the most, compared to the object detection task which is frame by frame-based. Therefore, the power consumption of executing the proposed SoE prediction model is negligible in the considered MEC system.

V. Performance Evaluation

In this section, we evaluate the performance of CNN-LSTM on the data collected in Section III. The data is processed so as to allow M time-step history and N time-step future windows around each data sample. The resultant data is split 80-20 for training and testing data samples. Each input of the model is in a 2D format of the data features listed in Fig. 3 over a size M window.

For all the deep learning models used in the following experiments, we implement them using Keras library with TensorFlow backend. All experiments are performed on Google Colaboratory Pro resources run on 3 vCPU @ 2.2GHz, with 26.75 GB RAM

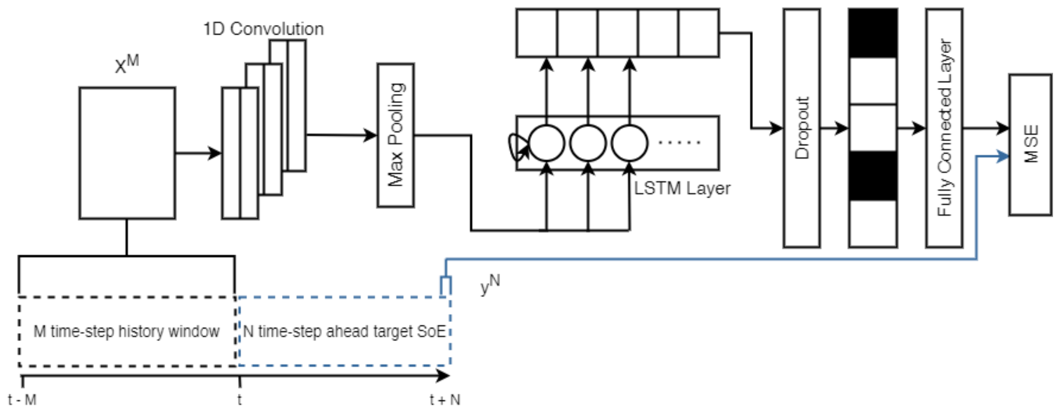


Fig. 4: CNN-LSTM model architecture used for SoE prediction using multivariate data obtained from RE-powered RSU testbed

	RMSE	MAE
Vanilla RNN	1.38	0.98
Vanilla LSTM	0.26	0.20
CNN-LSTM	0.23	0.18

TABLE I: RMSE and MAE performance for predicting hour ahead SoE using 1-hour history window on Vanilla RNN, Vanilla LSTM, and CNN-LSTM models

and 147.15 GB disk space. The TensorFlow sessions utilize NVIDIA Tesla P100 GPU (16 GB).

We evaluate the performance of prediction using the root mean square error (RMSE) and mean absolute error (MAE) metrics. The definitions of these metrics are the following:

$$\mathcal{R}MSE = \sqrt{\frac{\sum_{i=1}^{n}(S_{predict}^{i} - S_{actual}^{i})^2}{n}} \tag{9}$$

$$\mathcal{M}AE = \frac{\sum_{i=1}^{n}|S_{predict}^{i} - S_{actual}^{i}|}{n} \tag{10}$$

A. Comparison with Other Deep Learning Models

We compare the performance of CNN-LSTM as described in Fig. 4 with a Vanilla RNN (i.e. one RNN layer with 450 units followed by a layer with dropout probability of 0.1), and a Vanilla LSTM model (i.e. one LSTM layer of 450 units followed by a dropout probability of 0.1). The resultant vector from the dropout layer of the above models will be fed into a fully connected layer to output the SoE prediction values. The performance of the above models to predict 1-hour ahead SoE with 1-hour history window (i.e. $M = 1, N = 1$) are listed in Table I.

Table I reports results from 3-fold cross validation comparison of the three models. The models are run for 75 epochs with an early stopping patience of 15 epochs. The validation loss threshold set for early stopping is of the order 1e-5. Every model is run using Adam optimizer to reduce the mean square error (MSE) loss between predictions and test values. The performance of Vanilla RNN is the worst among these three models. For example, Vanilla LSTM and CNN-LSTM models perform 81% and 83% better than Vanilla RNN, in terms of RMSE, respectively. This is because there exist long-term dependencies in predicting future SoE, which involve trends in power demand and generation, that Vanilla RNN faces problems recognizing. On the other hand, the performance of CNN-LSTM is the best among these models. Its RMSE and MAE are 11.5% and 10% better than Vanilla LSTM, respectively. Since the data is multivariate, an additional convolution layer on the LSTM layer helps in recognizing spatially invariant and significant information across variables in the history window.

B. SoE prediction Visualization

We then visualize the prediction performance by comparing the temporal sequence data of observed SoE and CNN-LSTM's predicted SoE in an example dataset collected on Jan. 7, 2020. The result for the SoE from 11AM to 4 PM is reported in Fig. 5a, where the blue curve represents the actual SoE and the red dash curve shows the predicted SoE with 1 hour history window and 1 hour ahead future window. e.g. for SoE prediction of 11 AM, the history window is 9 AM to 10AM. The SoE prediction matches the actual SoE in large scale. The corresponding RMSE, MAE are 0.23, 0.18, respectively. We also provide Fig. 5b, which zooms in the area surrounded by the green box of Fig. 5a and shows the detailed result of the SoE prediction. To elaborate the reason why SoE curve flattens after 1:30PM, we include

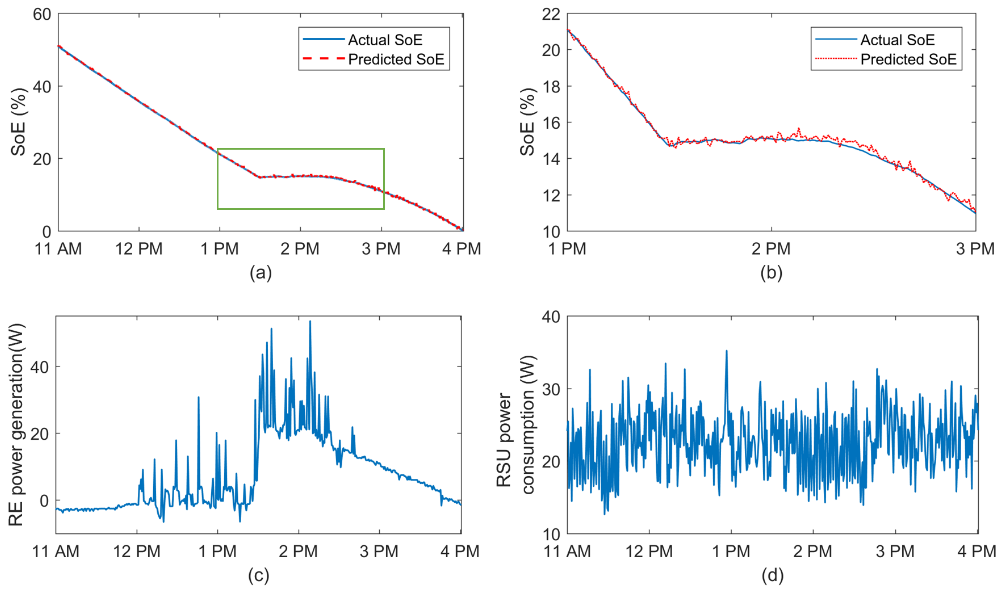


Fig. 5: Visualization of a prediction example on data collected on Jan. 7, 2020, with (a) the 1-hour ahead SoE prediction by the CNN-LSTM model compared with actual SoE, (b) specific plot of (a) from 1 PM to 3 PM, and the corresponding (c) RE generation, and (d) RSU's power consumption

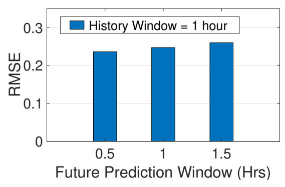
the corresponding RE power generation and RE-powered RSU power consumption data in Fig. 5c and Fig. 5d, respectively. Due to the space limitation, we combined the solar and wind power generation data as the RE power generation in Fig. 5c. The SoE curve flattens because of a large increase of RE power generation after 1:30PM while the RSU's power consumption remains the same. This is due to the location of our testbed, the solar unit is blocked by the nearby buildings until noon during the winter. Note that we plot power consumption in Fig. 5d for explaining the SoE behavior, during the prediction, our model doesn't have the knowledge of the RSU's power consumption.

C. Impact of Different History and Future Windows

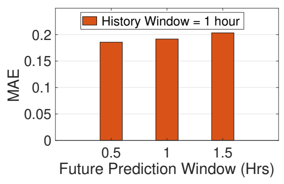
We also conducted experiments with different history and future window sizes for SoE prediction with CNN-LSTM as shown in Fig. 6. We segregated the full training dataset for 5-fold cross validation with respect to sliding window that can support the longest window size for the cases we considered, and then used that same partitioning for all the cases for fair comparison. First, we compare the impact of different future prediction window sizes for a fixed 1 hour history window. Fig. 6a shows the RMSE value gradually increases from 0.23 with 30 mins ahead prediction to 0.24 with 1 hour ahead, and finally, to 0.26 with 1.5 hours ahead prediction. Same trend can be observed in Fig. 6b for the MAE values that, for fixed history, the MAE gradually increase while predicting for longer future window ahead. Then we measure the prediction performance for a fixed future window size and show the impact of choosing different history window sizes. Fig. 6c shows that if we predict 1 hour ahead with 30 mins history, the RMSE value is 0.25. However as we gradually increase the history window, the prediction performance gets better for fixed 1 hour future window ahead. With 1.5 hours history, the 1 hour ahead prediction gives RMSE 0.23. In this case as well, the MAE values show the consistent trend with the RMSE, and it shows decrease in MAE with increasing history window size as shown in Fig. 6d.

VI. Conclusion

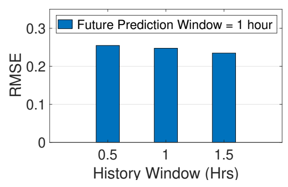
In this paper, we developed a machine learning based model for predicting the intra-hour and hour-ahead SoE in a renewable energy-driven MEC system. The prediction model involves the effects of solar and wind energy generation, as well as the usage of the energy by computing and communication loads at the MEC. Through real-world implementation of a testbed consisting of renewable energy generators, along with software defined radio based small cell base station and edge computing devices that serve mobile users, we conducted experiments over several days. The collected data are used to train for learning



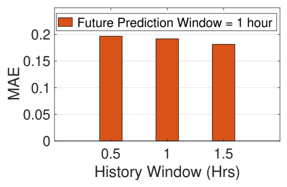
(a)



(b)



(c)



(d)

Fig. 6: Variation in RMSE and MAE values for SoE prediction with CNN-LSTM model with different combinations of history and future window sizes

future SoE in terms of real-time observable features. We show that using CNN-LSTM based model can predict the future SoE with high accuracy and it outperforms other models. We also show the impact of historical window and future window size on prediction performance. In future, this prediction model can be very useful in predictive and proactive decision making, for distribution of energy and loads in different dynamic scenarios.

Acknowledgment

This material is partially supported by the National Science Foundation under Grant No. CNS-1619184, the UC San Diego Center for Wireless Communications, and the Smart Transportation Innovation Program (STIP). The authors would also like to thank PrimoWind, Inc. for providing EnergiPlant, the compound of solar and wind power generators, to this work.

References

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2017.

[2] M. Aazam, I. Khan, A. A. Alsaffar, and E.-N. Huh, "Cloud of things: Integrating internet of things and cloud computing and the issues involved," in *Proceedings of 2014 11th International Bhurban Conference on Applied Sciences & Technology (IBCAST) Islamabad, Pakistan, 14th-18th January, 2014*. IEEE, 2014, pp. 414–419.

[3] K. Mamadou, E. Lemaire, A. Delaille, D. Riu, S. Hing, and Y. Bultel, "Definition of a state-of-energy indicator (soe) for electrochemical storage devices: application for energetic availability forecasting," *Journal of The Electrochemical Society*, vol. 159, no. 8, p. A1298, 2012.

[4] R. Hickey and T. M. Jahns, "Direct comparison of state-of-charge and state-of-energy metrics for li-ion battery energy storage," in *2019 IEEE Energy Conversion Congress and Exposition (ECCE)*. IEEE, pp. 2466–2470.

[5] F. Yang, X. Song, F. Xu, and K.-L. Tsui, "State-of-charge estimation of lithium-ion batteries via long short-term memory network," *IEEE Access*, vol. 7, pp. 53 792–53 799, 2019.

[6] T. Mamo and F.-K. Wang, "Long short-term memory with attention mechanism for state of charge estimation of lithium-ion batteries," *IEEE Access*, 2020.

[7] K. Sarrafan, K. M. Muttaqi, D. Sutanto, and G. E. Town, "An intelligent driver alerting system for real-time range indicator embedded in electric vehicles," *IEEE Transactions on Industry Applications*, vol. 53, no. 3, pp. 1751–1760, 2017.

[8] D. Antolín, N. Medrano, and B. Calvo, "Analysis of the operating life for battery-operated wireless sensor nodes," in *IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2013, pp. 3883–3886.

[9] S. Baidya, Y. Ku, H. Zhao, J. Zhao, and S. Dey, "Vehicular and edge computing for emerging connected and autonomous vehicle applications," in *Proceedings of the 57th Annual Design Automation Conference 2020 (DAC '20)*, 2020.

[10] Nvidia Co., CA, USA. Datasheet v1.2, *NVIDIA Jetson TX2 Series System-on-Module*, 2014, (Accessed: Apr. 9, 2020). [Online]. Available: https://developer.nvidia.com/embedded/jetson-tx2

[11] Ettus Research, TX, USA, *USRP B210*. [Online]. Available: https://www.ettus.com/all-products/ub210-kit/

[12] Software Radio Systems, Cork, Ireland, *srsLTE: Your own mobile network*, 2016, ((Accessed: Jul. 10, 2020)). [Online]. Available: https://www.srslte.com/

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[14] Y.-J. Ku, P.-H. Chiang, and S. Dey, "Quality of service optimization for vehicular edge computing with solar-powered road side units," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2018, pp. 1–10.

[15] Department of Transportation, New York, NY, USA, *NYS Traffic Data Viewer*, (Accessed: Mar. 6, 2018). [Online]. Available: https://www.dot.ny.gov/tdv

[16] K. Yan, X. Wang, Y. Du, N. Jin, H. Huang, and H. Zhou, "Multi-step short-term power consumption forecasting with a hybrid deep learning strategy," *Energies*, vol. 11, p. 3089, 11 2018.

[17] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*. Springer, 1999, pp. 319–345.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.