IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 15, 2021, accepted February 15, 2021, date of publication March 2, 2021, date of current version March 9, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3063066

An Ensemble Approach for Multi-Step Ahead Energy Forecasting of Household Communities

AIDA MEHDIPOUR PIRBAZARI[1], (Graduate Student Member, IEEE), EKANKI SHARMA[2], (Student Member, IEEE), ANTORWEEP CHAKRAVORTY[1], (Member, IEEE), WILFRIED ELMENREICH[2], (Senior Member, IEEE), AND CHUNMING RONG[1], (Senior Member, IEEE)

[1]Department of Electrical and Computer Engineering, University of Stavanger, 4036 Stavanger, Norway
[2]Institute of Networked and Embedded Systems, University of Klagenfurt, 9020 Klagenfurt am Wörthersee, Austria

Corresponding author: Aida Mehdipour Pirbazari (aida.mehdipourpirbazari@uis.no)

This work was supported by the Research Council of Norway under Project 267967.

ABSTRACT This paper addresses the estimation of household communities' overall energy usage and solar energy production, considering different prediction horizons. Forecasting the electricity demand and energy generation of communities can help enrich the information available to energy grid operators to better plan their short-term supply. Moreover, households will increasingly need to know more about their usage and generation patterns to make wiser decisions on their appliance usage and energy-trading programs. The main issues to address here are the volatility of load consumption induced by the consumption behaviour and variability in solar output influenced by solar cells specifications, several meteorological variables, and contextual factors such as time and calendar information. To address these issues, we propose a predicting approach that first considers the highly influential factors and, second, benefits from an ensemble learning method where one Gradient Boosted Regression Tree algorithm is combined with several Sequence-to-Sequence LSTM networks. We conducted experiments on a public dataset provided by the Ausgrid Australian electricity distributor collected over three years. The proposed model's prediction performance was compared to those by contributing learners and by conventional ensembles. The obtained results have demonstrated the potential of the proposed predictor to improve short-term multi-step forecasting by providing more stable forecasts and more accurate estimations under different day types and meteorological conditions.

INDEX TERMS Ensemble learning, gradient boosted regression tree, household communities, load forecasting, Sequence-To-Sequence LSTM, solar output forecasting.

I. INTRODUCTION

The shift towards low carbon and sustainable energy production is gaining momentum to support the increasing energy demand. Due to this transition, the installed PV generation capacity is expected to increase by more than 21.9 TW by 2050 [1]. Photovoltaics nowadays enables the generation of localized electricity among residential consumers at a lower cost than that of the power grid. Costs are even smaller if energy storage devices are used. Therefore, self-consumption, the consumption from self-produced electricity, is expected to grow among households. Moreover, in residential buildings, energy use is on the increase by using more electric vehicles and high-demand appliances. In such an environment, forecasting energy demand and supply from micro-generation sources become necessary to tackle the instability induced by the integration of PV to the power grid and reduce the uncertainty of demand [2].

For electricity suppliers, forecasting demand and micro-generation provide useful information to achieve demand and supply equilibrium, serve peak demands, and maintain reliable grid operation. From the customers' point of view, energy forecasts through a smart Energy Management System (EMS) will enable them to make smarter decisions on managing their use, increasing self-consumption, trading energy, and reducing electricity bills. Intelligent Energy management in buildings will lead to a decrease in the electricity intake from the power grid, which in turn lowers the total operating costs [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Kaitai Liang.

Energy forecasts are made with various time scales corresponding to a particular decision-making activity. Very short-term (from a few minutes to a few hours ahead) is generally used for flow control and real-time dispatch; short-term (from a few hours to a few weeks ahead) for adjusting generation and demand and electricity trading; medium-term and long-term (from a few months to a few years ahead) for PV plant planning, power maintenance, etc. [4].

Generally, forecasting energy consumption with short-term to medium-term horizons at smaller scales such as a residential building or community level is quite challenging due to several demographic and economic factors which influence the load with different degrees. These factors typically include population, size and structure of the building, number of residents, number of appliances under usage, heating, ventilation, air conditioning system, and weather data (humidity, wind speed, temperature, precipitation, etc.). A comprehensive study of primary features that influence electricity energy demand is conducted in [5].

Similarly, the PV power output is difficult to predict with short-term horizons due to its dependency on uncertain meteorological factors, such as solar irradiance, atmospheric temperature, module temperature, wind pressure, wind direction, and humidity. This causes the power output of a PV system to change dynamically. A recent correlation study reported in [6] shows that solar irradiance has the highest correlation with PV power output compared to other weather parameters. The result is validated for various weather conditions in [7] and [8].

The energy estimation can be grouped into two categories according to forecasting steps: one step-forecasting, which estimates future demand or supply one step ahead in time, and multi-step forecasting that predicts multiple time steps into the future. Several architectures are proposed in the literature for one to multi-step ahead energy forecasting at building levels. They are broadly categorised into three categories: physical, data-driven (statistical or computational intelligent) and hybrid methods [9] and [10]. Physical approaches, also known as analytical methods, rely on the mathematical modelling of the building under study. Data-driven methods, in contrast, focus on statistical analysis performed on historical time-series data with different input variables. Hybrid approaches incorporate both physical and data-driven methods to exploit the benefits of each approach.

Several studies have analysed the potential of data-driven and hybrid methods for electric load forecasting of buildings and cities at multi-step ahead. The approaches based on Artificial Neural Network (ANN) [11], [12] and Support Vector Machines (SVMs) are successfully applied for energy analysis of buildings [13]. Moreover, the deep learning approaches have been utilized for energy consumption prediction at multi-step ahead. A deep learning method based on 2D Convolutional Neural Network (CNN) to forecast one-day ahead load with the fifteen-minute resolution is investigated in [14]. The prediction accuracy of Auto-Regressive Integrated Moving Average (ARIMA), Long-Short-Term-Memory (LSTM), and Recurrent Neural Networks (RNN) models are compared in [15]. The results showed the effectiveness of the LSTM in comparison with ARIMA for multi-step electric load forecasting. A variant of LSTM based on Multi-Channel with time location (TL-MCLSTM) is proposed for multi-step short-term consumption forecasting in [16]. The results showed that their proposed method outperformed the compared methods, including LSTM and CNN-LSTM. Yang *et al.* in [17] investigated the potential of a hybrid model for multi-step load forecasting based on Extreme Learning Machine (ELM), Recurrent Neural Network (RNN), SVM, and Multi-Objective Particle Swarm Optimization algorithm (MOPSO). The experiments on different cities showed that the optimization technique can improve the performance of the hybrid method and the combination technique can improve the prediction accuracy. Another ensemble approach based on Generalized Recurrent Neural Network (GRNN) and SVM is proposed in [18] to predict the one-week ahead electricity demand of state loads. The experimental findings indicate that the proposed approach is highly effective in predicting accuracy and model robustness.

The studies related to multi-step ahead PV production forecasting also reveal the strength of using Deep Learning (DL) and hybrid models. Various structures of LSTM networks in [19], and [20] have been proposed for solar power forecasting. Lee *et al.* [21], and Alzahrani *et al.* [22] have demonstrated the superior performance of deep models over conventional techniques for solar irradiance estimation. The authors in [10] have shown that the ensemble approaches in PV output forecasting increase the precision and efficiency of models compared with individual models by integrating linear and non-linear techniques. A hybrid learning algorithm incorporating Self-Organizing maps (SO), Support Vector Regression (SVR), and Particle Swarm Optimization (PSO) are also presented in [23] to forecast hourly solar irradiance at city levels. It is found that the combined technique outperforms conventional forecasting models. Another comparative study in [24] shows the superiority of a blended model against individual algorithms including SVR, Random Forests (RF), Deep Neural Networks (DNN), and Extreme Gradient Boosting (XGB) for day-ahead PV output forecasting. In [25] the tree-based ensemble methods based on extra trees (ET) and random forests (RF) also demonstrate satisfactory results compared with support vector regression (SVR) as the widely used machine learning method.

Despite several studies proposed in the literature dedicated to short-term and multi-step forecasting, a few have investigated the potential of deep-learning-based hybrid techniques using various input variables. Presumably, no study has focused on forecasting electricity consumption and PV production across one dataset at the local level. Furthermore, most of the existing studies related to PV production forecasting have been conducted on a proprietary dataset. The limited availability of these datasets does not allow a fair comparison between the results obtained using different

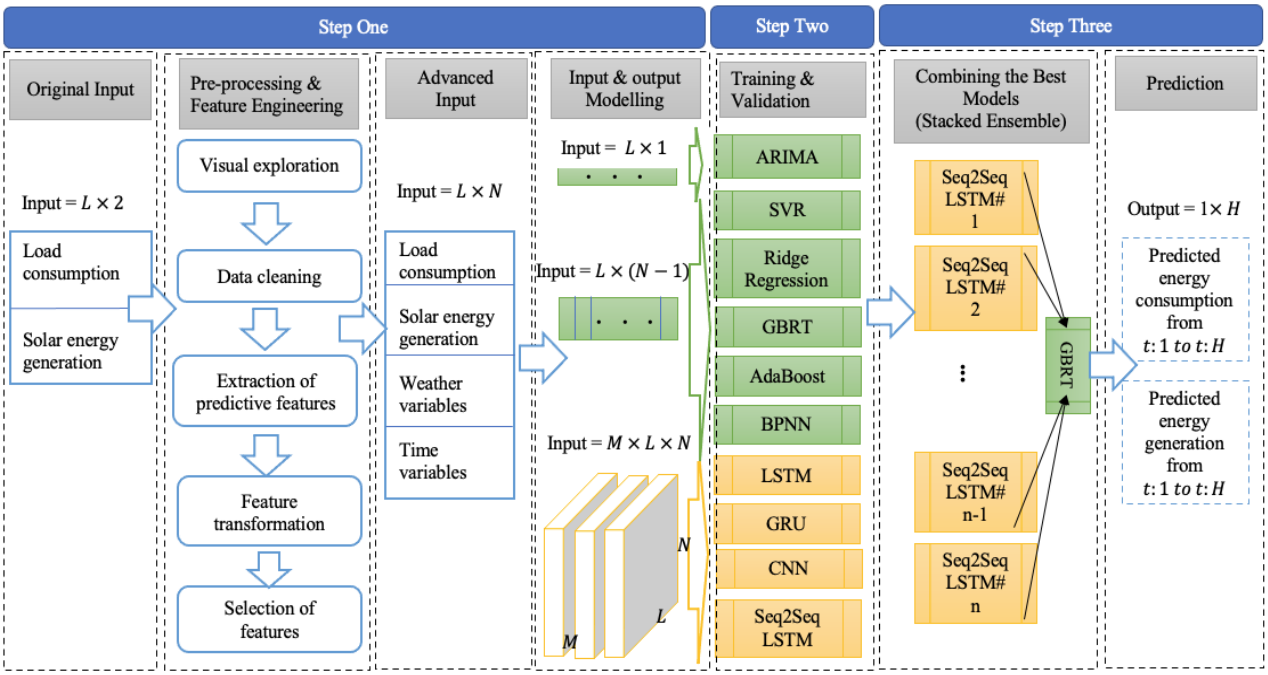

FIGURE 1. Framework for prediction of two targets: total energy consumption and energy generation. Original input data has the dimension of $L \times 2$ where L is the time lag and 2 refers to the number of target variables. The advance input consists of N input variables, including energy consumption, energy generation, meteorological and time factors with the same number of lags L. For the models in green rectangles, except for ARIMA fitted to uni-variate data, the multivariate input was flattened to $L \times (N-1)$ array to meet the requirements of the input layer. In contrast, for the models represented with yellow rectangles, the input is transformed to $(M \times L \times N)$ tensor where M represents the batch size. The output dimension is 24-horizon ahead ($H = 24$). For the green models that do not support the prediction of two target variables in one-step forward, one regressor is fitted per target and for most of them that naturally do not support multi-step prediction, one regressor is fitted per time step. For the others, one algorithm is trained to predict multi-steps ahead of two output variables.

forecasting model architectures. Hence configuring the forecasting models on a specific dataset is not optimal for a similar problem or dataset. This also makes it difficult to reproduce the result. To close the research gap, this paper presents four main contributions.

1) An ensemble approach with two levels is proposed to develop forecasting models for energy consumption and energy generation of household communities at multi-steps ahead. In the first level, multiple forecasting algorithms as base learners predict both target outputs in one-step forward. In the second level, the predictions for each target are used to train a meta learner aimed at generating multi-step predictions separately for each target.
2) To create a diverse ensemble, one promising algorithm from the category of deep neural networks is used for the base learners, and one from the class of conventional ML algorithms are employed for the meta learner. Instead of finetuning one deep network, multiple deep networks with different parameter settings were trained on the dataset, and their forecasts were combined to create a more robust estimate.
3) Three influential factors are considered as input variables to the forecasting models: time variables, meteorological data, and historical electricity consumption and PV power output. Before model development, two feature selection techniques and two machine learning algorithms are used to select the optimal subset of input variables.
4) A publicly available dataset with actual observations from an Australian electricity distributor is selected as a case study with both electricity and solar PV output measurements from 300 houses. The forecasting methods are evaluated from the perspective of accuracy and prediction stability.

The paper is organised as follows: Section II presents and discusses detailed forecasting framework steps. Section III describes the case study, followed by forecasting experiments and results in section IV. Section V concludes the study.

II. FORECASTING FRAMEWORK

Fig. 1 depicts the framework for multi-step ahead energy forecasting, consisting of three main steps. Step One; data preprocessing, Step Two; model development and selection of the most accurate models, and Step Three; development and evaluation of an ensemble model based on the results of the previous step.

The first step is further classified into five main tasks: visual exploration, data cleaning, feature extraction and transformation, feature selection, and input/output modelling. During the first step, we aim to better understand the data, improve the data quality, determine predictive features, select

the most useful ones and convert data into the appropriate format for forecasting models. In the second step, several commonly used algorithms in time series forecasting e.g., ARIMA, SVR, LSTM, etc. are trained and evaluated on large sets of training and validation data. The aim is to shortlist the most promising models for the energy forecasting problem. In the third step, the resulted algorithms from Step Two with the lowest prediction error on average i.e., Seq2Seq LSTM and GBRT, are combined to create an ensemble model. The trained ensemble technique is then applied to predict several household communities' energy consumption and generation as test sets. The following subsections present the details of each step.

A. STEP ONE

1) VISUAL EXPLORATION

Visual exploration allows us to understand the dataset more effectively. It is also useful to recognize patterns and trends within the data more quickly. In this study, we provide several plots and statistics to serve these purposes. The visual exploration results further promote our decisions on subsequent steps such as data cleaning, feature extraction, and feature engineering.

2) DATA CLEANING

Typically, machine learning algorithms cannot perform effectively with missing features. The raw smart meter data may contain missing values due to transmission error or smart meter failures, which would degrade the data quality. Data cleaning helps in managing missed values in the time series of electricity load and solar output. To fix missed values, imputation, moving average (MA), and inference-based approaches can be used.

This study fills the missing values from the surrounding measurements with a variant of MA known as an exponential weighted moving average (EWMA) [26]. This technique solves the problem by computing the arithmetical mean of data surrounding a missing value while placing a higher weight on the latest data. As both electricity load and solar output continuously vary, two closer measurements are more similar. Thus, applying the EMWA technique can be useful for replacing missing values.

3) FEATURE EXTRACTION

The performance of a forecasting model mainly depends on the input variables as the predictive features. As mentioned in Section I. previous studies have shown several influential factors on accurate forecasting of household energy consumption and solar cells' output. The most important and common factors between the two targets include historical load measurements, weather conditions, time variables, and customer socio-economic factors.

In this analysis, we created a set of candidate features based on two references: the literature study and data exploration results. The majority of the candidate features are influential on both prediction targets such as outdoor temperature and historical load. While some are more effective for predicting only one target, such as 'solar zenith angle' for solar output estimation and 'weekends' or 'holidays' on electricity consumption prediction. Notably, the candidate set of features will be discussed in detail in Section III, Subsection B. The subset of features as the final input to the forecasting models will be introduced in Section III, Subsection C.

4) FEATURE TRANSFORMATION

Most machine learning algorithms perform well with numerical features instead of categorical features. In this work, the categorical attributes are transformed into numerical attributes using a commonly used method as One-hot encoding. This technique generates a binary feature for each subclass of categorical features. For instance, it converts a feature with two sub-classes into two binary features. Furthermore, some numerical features with the current scale or value are not informative enough as input to the forecasting models. Therefore, they are transformed into an appropriate format before using in the model development process. The attributes and functions used in the feature transformation process will be discussed in Section III, Subsection C.

5) FEATURE SELECTION

Selecting the best combination of the variables having a high correlation with energy consumption and production can improve the performance of the forecasting algorithm. However, an input space with redundancy and many inter-correlated features typically decreases the accuracy of the prediction model and contributes more to the over-fitting problem. To avoid overfitting, two feature selection methods, which have been proposed in the literature, including Pearson Correlation Coefficient (PCC) [27], [28] and Recursive Feature Elimination Technique (RFE) [29]–[31] were used to reduce the dimension of input space.

The PCC measures the linear correlation between two variables x and y as Equation (1):

$$\rho(x, y) = \frac{Cov(x, y)}{\sigma(x), \sigma(y)} \tag{1}$$

where Cov is the covariance; $\sigma(x)$ and $\sigma(y)$ are the standard deviations of x and y. If we consider each candidate variable as x and each prediction target as y, then the x variables whose correlation with the y target exceed a predefined threshold can be selected as most related features. However, The PCC method cannot capture nonlinear relationships between the x and y variables. Therefore, the RFE method combined with a training algorithm was used to discover the variables with high prediction ability and even with nonlinear relations with the targets.

The RFE[1] ranks features to evaluate their importance according to a specific criterion. It also uses the model

[1] Available at https://scikit-learn.org/stable/modules/feature_selection.html#rfe

accuracy to determines the features contributing most to the prediction task in a recursive way. The determination process starts with training a prediction algorithm on the original feature set. It then continues with measuring the feature importance and removal of the less relevant ones. This procedure is repeated until it reaches the desired number of features to select. The results of the feature selection process are discussed in Section III, Subsection C.

6) INPUT AND OUTPUT MODELLING

As mentioned in Section I, the task of energy load forecasting involves many influential factors. Each factor can be dependent on both its precedent values and the values of other factors. For instance, household energy usage and solar output are strongly correlated to their historical values and to air temperature values at the same time. To learn the potential correlations, it is useful for the learning algorithm to receive this information as a multivariate time series which contains multiple variable values at each observation time step. Therefore, in this analysis, we provide a description of multivariate energy forecasting. The aim is to use historical time series data to predict the future temporal values of multiple energy variables. The model input not only contains the historical energy factors but also includes other predictive attributes. This problem is formulated as Equation (2):

$$D \xrightarrow{M(D)} (Y_1, Y_2) \tag{2}$$

where $M(D)$ refers to the learning model which aims to predict the next H values of energy consumption and generation of households from time step t as two time series data $Y_1 = [y_{1,t}, y_{1,t+1}, \ldots, y_{1,t+H}]$ and $Y_2 = [y_{2,t}, y_{2,t+1}, \ldots, y_{2,t+H}]$ given a history multivariate time series dataset $D = (x_{i,j}|i = 1, 2, \ldots, N; j = t - L, \ldots, t - 1, t)$ where N denotes the number of input variables (features) and L represents the window length of history data.

B. STEP TWO

In this step, many forecasting models from different categories (e.g., Autoregressive, linear, SVM, Bagging, and Boosting ensembles based on Decision trees, Neural networks, and deep learning) are trained and evaluated on a subset of training data using standard and suggested parameters in the literature. The models involved are briefly presented in the remainder of this section. The experimental settings and prediction results of given techniques will be discussed in Section IV.

1) PERSISTENCE

One of the easiest ways of forecasting a time series's future behavior is the so-called persistence model. Persistence in the context of energy forecasting assumes that the future values of the demand or supply are determined under the basis that the conditions stay constant between the current time and the future time. For long prediction periods, however, this strategy lacks the skill of forecasting. Persistence was only assessed here for comparative analysis.

2) AUTOREGRESSIVE INTEGRATED MOVING AVERAGE (ARIMA)

ARIMA belongs to the category of statistical models for forecasting time series data. In ARIMA, the generated forecasts are treated as a linear function of the most recent observations and past random errors. Mathematical details are provided in [32]. This technique is usually unable to capture non-linear relationships between the time series components and is often applied to univariate time series data. Sample applications of the hybrid models with ARIMA applied to short-term time series forecasting can be found in [33] and [34].

3) RIDGE REGRESSION

Similar to the Linear Regression algorithm, Ridge regression assumes a linear relationship between input variables and the output. However, it makes the model simpler by adding L2 penalty to the loss function during training. L2 penalty has the effect of reducing coefficient values of those inputs that have less contribution to the forecasting task. It is calculated based on the sum of the squared coefficient values. In our research, Ridge regression [35] was selected to evaluate and compare the potential predictive ability of a regularized linear model against non-linear statistical models.

4) SUPPORT VECTOR REGRESSION (SVR)

SVR is an extension of a Support Vector Machine (SVM) used for regression problems. The SVR is based on statistical learning theory and structural risk minimization. In this method, instead of minimizing the training error, the generalization error is reduced. The generalization functionality optimization is achieved by mapping the initial input space through non-linear kernel functions to a high-dimensional feature space. A mathematical explanation of SVR is provided in [36]. The SVR models are successfully applied to electrical load forecasting [18], [37], [38] as well as renewable energy prediction [39]–[41].

5) ADABOOST

AdaBoost is an ensemble method that combines many weak learners into a strong learner. The most common algorithm used with AdaBoost as one-level Decision Tree (DT) algorithm. In this method, the Decision Trees are sequentially added and trained as weak learners. This process repeats until a predefined number of learners has been created, or there is no further reduction in the training error. In an AdaBoost used for regression tasks, final predictions are made by calculating the weighted median prediction of the learners in the ensemble. The detailed process is given in [42]. The benefit of AdaBoost over SVM is the ability to identify only those features with more predictive capacity during training. This ability would potentially lead to enhanced execution time due to the lower dimensionality of input space.

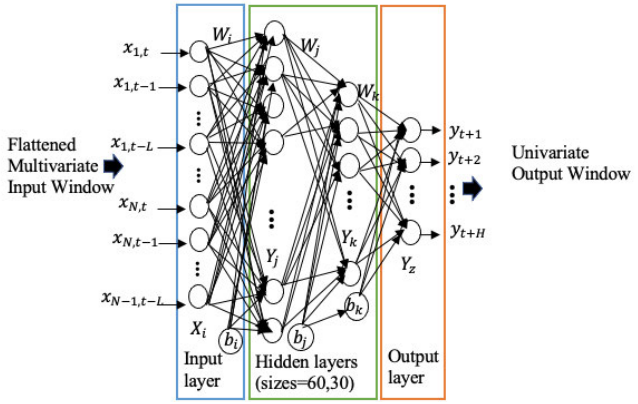

FIGURE 2. BPNN architecture for multi-step prediction.

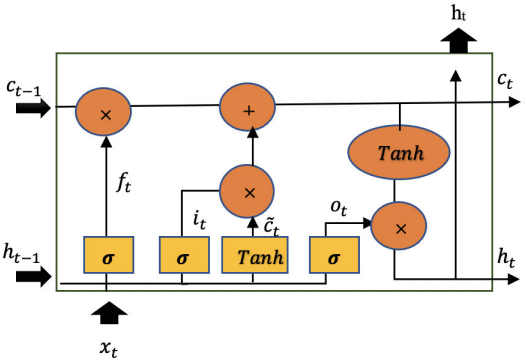

FIGURE 3. Structure of a basic LSTM cell.

6) GRADIENT BOOSTED REGRESSION TREE (GBRT)

GBRT is another type of ensemble algorithm developed based on Boosting and Decision Tree algorithms. In this approach, the model loss is determined using a gradient descent technique when each weak learner is introduced to the GBRT ensemble. This process adds a tree to the model that decreases the loss. To improve the performance of the ensemble, each new learner's output is then added to the output of the sequences of the generated tree. The details of the algorithm is discussed in [43].

7) BACK PROPAGATION NEURAL NETWORK (BPNN)

BPNN is a type of artificial neural network (ANN) that uses a backpropagation algorithm [44] for training the network. The architecture of a BPNN model includes an input layer, one or more hidden layers and an output layer. The non-linear activation function in the hidden layer(s) can capture the complex relationship between variables in the input and output layers. The flow of signals in a conventional BPNN is from inputs to outputs. Thus the network architecture is called the Feed Forward network. The ability to estimate any continuous function, the high generalization ability, and imposing no restrictions on the input variables have made the ANNs considerably useful for forecasting time series, specifically where the data volatility is high, as for example in load data. A review of using ANNs for building electrical energy consumption forecasting and photovoltaic power generation is provided in [45] and [46].

Fig. 2 depicts the architecture of BPNN, which was developed for our problem. To satisfy a BPNN network's input requirements, the time series data with $(N-1)$ variables from previous t time steps is first framed as sliding windows with the size of L. Reducing one variable from the input with the size of N indicates that the BPNN that is trained to predict only future load consumption values removes the solar energy generation variable from its input. Accordingly, the network which estimates solar output ignores electricity load from the input set.

Each two-dimensional input window is then transformed to a flattened vector of X_i and is fed to the network where $X_i = (x_{1,t}, x_{1,t-1}, \ldots, x_{1,t-L}, \ldots, x_{N-1,t}, \ldots, x_{N-1,t-L})$ and $i = (1, 2, \ldots, (N-1) \times L)$. Next, the network computes an output vector of Y for the next time steps from $t : t+1$ to $t : t+H$, where H represent the forecast horizon. Each element of Y vector at time step t is then calculated by Equation (3):

$$y_t = b_k + \sum_{k=1}^{n_2} w_k f_2 \Big(\sum_{j=1}^{n_1} w_{jk} f_1 \Big(\sum_{i=1}^{p} w_{ij} + b_i \Big) + b_j \Big) \qquad (3)$$

where y_t is the output y of the BPNN at time t; f_1 and f_2 are the non-linear functions of the neurons in the first and second hidden layers; n_1 and n_2 are the number of neurons in the hidden layers; w_{ij} are the weights of neuron j connecting the input with the first hidden layer; and w_{jk} are the weights of neuron k connecting the neuron j in the first hidden layer with the neuron k in the second hidden layer; w_k are the weights connecting the output of the neuron k with the output neurons and b_i, b_j, b_k denote bias vectors in the input layer and two hidden layers respectively. Note that, in the given problem, one BPNN was trained per target for multi-step prediction. Both networks accept the same multivariate input data; however, one is trained to estimate energy consumption, and one is trained to predict energy production.

8) LONG-SHORT TERM MEMORY NETWORK (LSTM)

LSTM, proposed initially by Hochreiter *et al.* [47], is an artificial recurrent neural network (RNN) [48] architecture that is well suited to time series prediction. There are feedback connections in the LSTM to update the state of neurons with previous inputs, in contrast to conventional feed-forward neural networks. Moreover, unlike typical RNNs, they benefit from long-term memory cells to resolve the disadvantage of unstable gradients while learning series with long-term dependencies. There are four main connected layers in the basic LSTM cell, as shown in Fig. 3.

The main layer known as control state computes the long-term state c_t by analyzing the current input vector x_t and previous short-term state h_{t-1}. The other three layers are gate controller layers known as f_t for the forget gate, i_t for the input gate and o_t for the output gate. Gate operations, such as removal, writing and reading are performed to change the LSTM cell's states and its output at each time step.

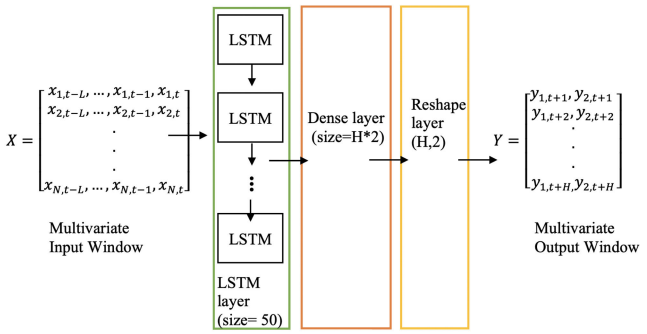

FIGURE 4. Structure of LSTM network for multivariate multi-step prediction.

The LSTM computations are shown through Equation (4) to Equation (9):

$$i_t = \sigma(W_{xi}^T.x_t + W_{hi}^T.h_{t-1} + b_i) \quad (4)$$
$$f_t = \sigma(W_{xf}^T.x_t + W_{hf}^T.h_{t-1} + b_f) \quad (5)$$
$$o_t = \sigma(W_{xo}^T.x_t + W_{ho}^T.h_{t-1} + b_o) \quad (6)$$
$$\tilde{c_t} = \tanh(W_{xc}^T.x_t + W_{hc}^T.h_{t-1} + b_c) \quad (7)$$
$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c_t} \quad (8)$$
$$h_t = o_t \otimes \tanh c_t \quad (9)$$

where σ denotes the logistic activation function; W_{xi}^T, W_{xj}^T, W_{xo}^T and W_{xc}^T are the weight matrices of each of the four layers for their connection to the input vector x_t; W_{hi}^T, W_{hj}^T, W_{ho}^T and W_{hc}^T are the weight matrices of each of the four layers for their connection to the previous short-term state h_{t-1}; b_i, b_f, b_o, b_c are the bias terms for each of the four layers; c_t is the long-term state at time t, and h_t is the output of the LSTM cell. In short, the input gate decides which parts of input at time t, should be added to the long-term state c_t; the forget gate stores the important part in c_t as long as it is needed, and output gate decides which parts of c_t should be read and output at a current time step. The decisions made by gate controllers are implemented through sigmoid activation functions whose outputs range from zero to one. Feeding output values close to zero to element-wise multiplication operation ($\otimes$) makes the gates close and cell's states unchanged, while producing values close to one, makes them open and change the states.

The LSTM network which was developed for this study is illustrated in Fig. 4. As shown, it includes four layers: an input layer, an LSTM layer with a certain number of hidden cells, a dense layer and a reshape layer. The input layer includes $N \times L$ neurons, where N denotes the number of features and L denotes the number of temporal lags. The LSTM layer is used to grasp the internal representation of input data by capturing the deep temporal dependencies within the multivariate time series. The dense layer is responsible for forecasting the future values of the two targets based on the hidden state of the last LSTM cell in the first layer. Since each neuron in the dense layer is responsible for producing one output, the number of neurons in the dense layer is set to as twice as H steps ahead, corresponding to the two targets' forecasting requirements. The reshape layer is finally used to transform the output layer to separate vectors for each target.

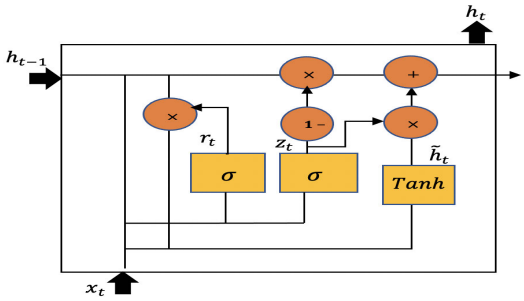

FIGURE 5. Structure of a basic GRU cell.

9) GATED RECURRENT UNIT (GRU) NETWORK

GRU, proposed by Cho et.al [49], is another version of LSTM with the same principles of processing long-term sequences but with a more compact structure. Compared to LSTM, it controls the information flow with fewer gates and parameters. GRU is thus trained faster and can be considered more effective in terms of simplified architecture. GRU has also successfully applied in both short-term residential load [50], and photovoltaic forecasting [51]. The architecture of a GRU cell is illustrated in Fig. 5.

In the GRU architecture, both state vectors are merged into a single vector h_t and a single gate both controls the input gate and forget gate. There is no output gate instead, the full state vector is output at every time step. However, there is a new gate controller which decides which part of the previous state will be available for the main layer. The equations from number 10 to 13 summarize the computation process of a GRU cell:

$$z_t = \sigma(W_{xz}^T.x_t + W_{hz}^T.h_{t-1} + b_z) \quad (10)$$
$$r_t = \sigma(W_{xr}^T.x_t + W_{hr}^T.h_{t-1} + .b_r) \quad (11)$$
$$\tilde{h_t} = \tanh(W_{xg}^T.x_t + W_{hg}^T.(r_t \otimes h_{t-1}) + b_h) \quad (12)$$
$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \tilde{h_t} \quad (13)$$

The GRU network architecture implemented for our study is similar to the one of LSTM network, which was illustrated in Fig. 4. The only difference is replacing LSTM cells with GRU cells in the recurrent layer.

10) CONVOLUTIONAL NEURAL NETWORK (CNN)

CNNs are a branch of neural networks initially designed for such areas as speech recognition [52] and image classification [53]. They have also been useful in predicting energy time-series data [54]. CNNs, compared to fully connected networks, are less complicated as they use fewer parameters to learn. They also do not require extensive feature engineering to extract and generalize features from the input space automatically.

Every convolutional network has three main components: (1) Convolution through one or multiple layers, where the

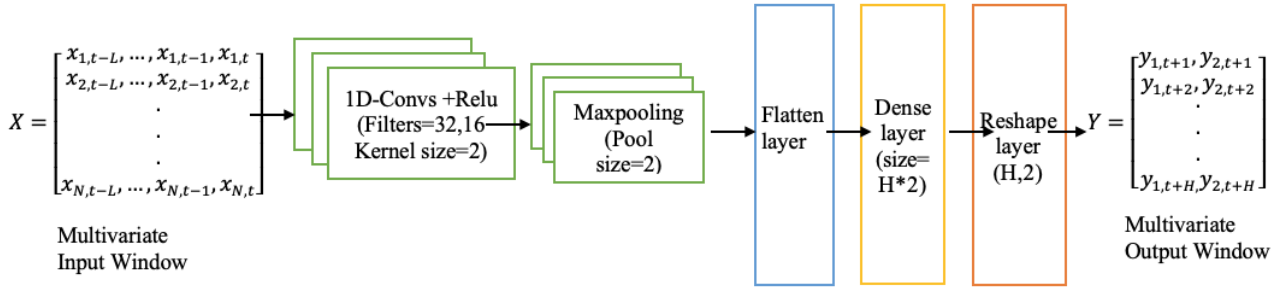

FIGURE 6. Structure of CNN network for multivariate multi-step prediction.

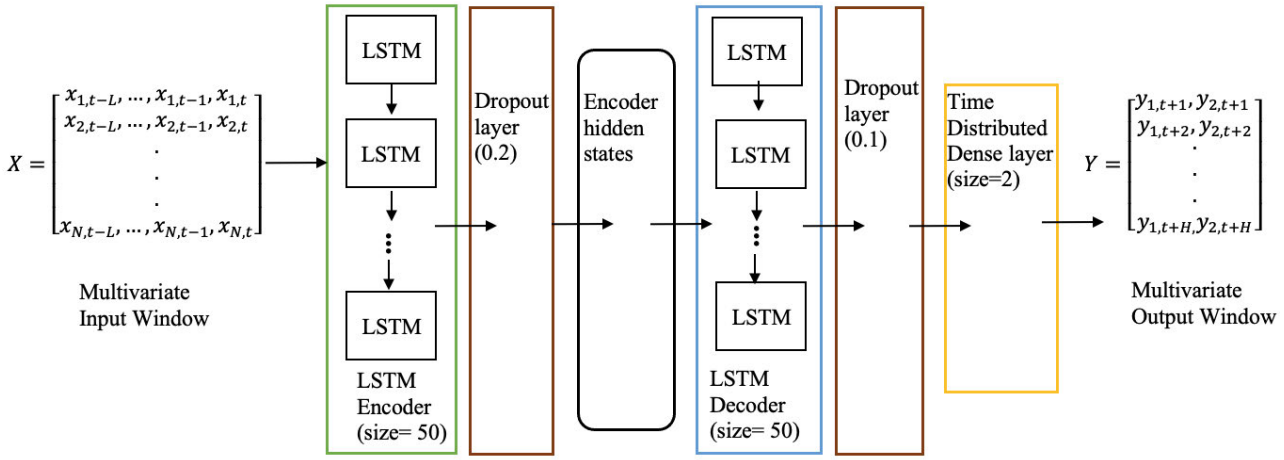

FIGURE 7. Structure of Seq2Seq LSTM network for multivariate multi-step prediction.

features are extracted from input through filters, a non-linear transfer function and feature maps. Feature maps allow neurons in each convolution layer to be connected to neurons located within a small rectangle in the previous layer. This architecture enables the network to concentrate on low-level features in the first hidden layer and assemble them into higher-level features in the next hidden layers. (2) Pooling reduces the dimensionality of feature maps while maintaining the relevant input information. (3) Fully connected layer that creates final non-linear combinations of features for making predictions by the network.

The architecture of the CNN network implemented for this study is depicted in Fig. 6. Six components are included: (1) an input layer that is the same as the one fed to LSTM and GRU networks. (2) The Convolutional layers that perform convolution operations on the multiple time series of the preceding layer with 32 and 16 filters including kernel size of two, followed by Relu layer; (3) a maximum pooling layer (as the most common type of pooling layer) that aggregates the inputs so that only the maximum value in each kernel passes through the next layer and the other inputs are discarded; (4) A flattening layer which transforms the 2-dimensional output to 1-d output; (5) a dense layer where neurons are connected to all the neurons in the previous layer. The neurons in the dense layer are responsible for producing forecasts at multiple steps ahead and (6) the reshape layer used to transform the output shape to the desired shape.

11) SEQUENCE-TO-SEQUENCE LSTM (SEQ2SEQ LSTM)

A Sequence-To-Sequence network (also called Encoder-Decoder) is a subclass of neural networks used to map sequences to sequences. Encoder-Decoder networks have been proposed to implement machine translation systems. The source language sentences are fed to the encoder, and the decoder's destination language sentences are interpreted.

A Seq2Seq network that utilizes LSTM cells in both encoder and decoder layers was first proposed by I.Sutskever *et al.* [55] for language translation. This network structure was further extended to time-series forecasting activities, especially targeting multi-step ahead forecasting.

The Seq2Seq architecture developed for our problem is adopted from this architecture, which performs multi-step forecasting of two targets based on multivariate input time series. As shown in Fig. 7, this network has two main components: one LSTM layer as the encoder and one LSTM layer as the decoder.

First, the input sequence is shown to the network one window at a time. Next, the LSTM encoder learns the relationship between time steps in the input. The output of the encoder shown as 'hidden states' layer in the architecture,

is a vector v_t that contains the internal representation of the input series. The decoder converts this vector further into two target sequences as the multistep forward prediction values. The probability of each target sequence is then computed as Equation (14):

$$p(y_{j,t+1}, y_{j,t+2}, \ldots, y_{j,t+H|X_1,X_2,\ldots,X_L}) = \prod_{t=1}^{H} p(y_{j,t} \mid v_t, y_{j,1}, y_{j,2}, \ldots, y_{j,t-1}) \quad (14)$$

where y_j denotes the jth target variable for $j = (1, 2)$ and $(X_1, X_2, \ldots, X_L)$ is a time series of multiple input series framed as a 2d-window; where X_i represents the ith column of this window and refers to the features values at ith time-step; and $(y_{j,t+1}, y_{j,t+2}, \ldots, y_{j,t+H})$ denote the forecasting values of H steps ahead of jth target value. Note that the number of temporal lags or window length L can be equal or different from the lookup size in the output sequences H.

C. STEP THREE

In this stage, we adopt an ensemble learning approach to create a strong forecasting model based on the algorithms evaluated in the previous step. When we combine the predictions of a group of predictors, the forecast accuracy is typically higher than that of the best individual predictor. The technique which utilizes the group of predictors is called ensemble learning. Ensemble learning can be performed in different ways.

One popular approach is called bagging, where predictors with the same training algorithms are trained on different random subsets of the training set. The sampling is performed with replacement and allows training instances to be sampled several times for the same predictor. After training of all predictors, the prediction is made for a new instance by simply aggregating the predictions of all estimators. As a result, the ensemble will have a lower variance than individual estimators. However, in the context of load forecasting, the bagging method with a random sampling technique cannot be optimal because of the inherent autocorrelation within the observations.

Another common approach is boosting, where the predictors are sequentially trained, and each tries to correct its predecessor. The main idea of boosting is building a strong learner based on many weak learners. The major downside to sequential learning is that it can not be parallelized because each predictor's training takes place after training and assessing the previous predictor.

A more advanced ensemble technique is called stacking. In this case, several predictors are trained on a subset of training data and make predictions on another subset of training data (called held-out set). A blender or meta learner then is trained on the forecasts of base predictors. In practice, the meta learner can be any learning algorithm such as Linear Regression or Decision Tree. A successful meta-learner effectively learns the optimal weights to combine the base learners and, as a result, produces more accurate predictions compared to the individual learners. Stacking is thus aimed at both reducing variance and improving the accuracy of forecasts. A detailed guide to ensemble learning is provided in [56].

In this work, we employ the stacking ensemble approach using the two most promising algorithms from Step Two. One algorithm with multiple variations is used to create the first-layer predictors (base learners), and one is used to merge the base learners' forecasts. We adopted and extended the development of base learners from [57] for processing multivariate input and output energy data. The training process and forecasting results of the ensemble model will be provided in Section III Subsection H.

III. CASE STUDY

A. DATASET DESCRIPTION AND DATA PARTITIONING

The original data was obtained from a publicly available dataset known as Ausgrid solar home electricity data [58]. It consists of half-hourly electricity consumption and generation of 300 Australian houses with rooftop solar systems from 2010 to 2013. Based on the recorded postal codes, two cities were recognized as the place of data collection: Newcastle and Sydney, each including 150 house profiles.

For this study, several aggregated profiles out of the original dataset were created to serve energy forecasting at low aggregated levels, such as small household communities. More precisely, 150 individual profiles from each city were initially converted into three equal-sized groups, each including 50 members. The individual readings of energy consumption and generation from each group were summed up to create six aggregated load profiles in total over two cities. We considered each group as a small community and named them A, B, C for Sydney and D, E, F for Newcastle. To provide smoother profiles and be consistent with the frequency level of external datasets such as weather data, the 30-minute series was downsampled to hourly series using the summation technique over one hour. Fig. 8 and Fig. 9 depict hourly electricity consumption and solar energy output of the six communities between 1 July 2010 and 30 June 2013 respectively.

As shown in the figures, the dataset was partitioned into different subsets with a total ratio of 77 % for training and 23 % for testing. The training subsets were then concatenated together, covering all six communities' hourly energy data over the first two periods (2010-11 and 2011-12) and the ones of A and D over the third period (2012-13). The test subsets which were used for evaluation of the baseline models include energy consumption and generation of four communities over the third period. More precisely, the observations of Community B and C from Sydney and Community E and F from Newcastle between 2012 and 2013 (The data on the right side of red dashed lines in Fig. 8 and Fig. 9). Each test subset was further divided into meta train and meta test sets with a ratio of 70 % and 30 % for building and evaluation of the ensemble method.

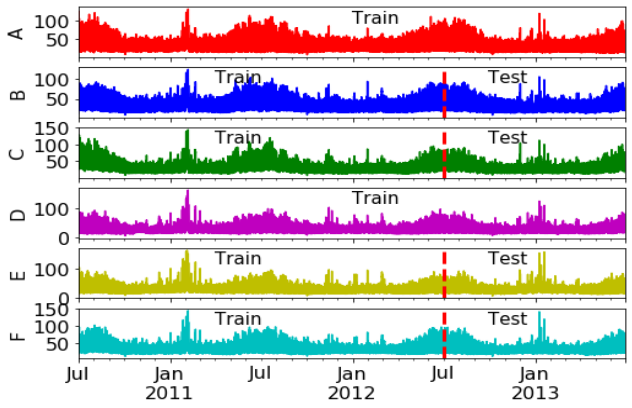

FIGURE 8. Hourly total electricity consumption of six communities over three years.

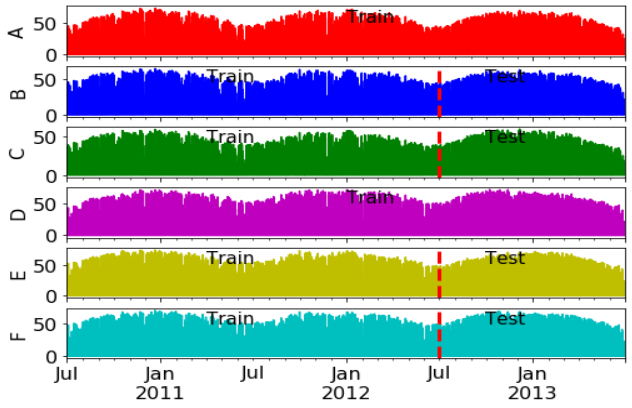

FIGURE 9. Hourly total solar output of six communities over three years.

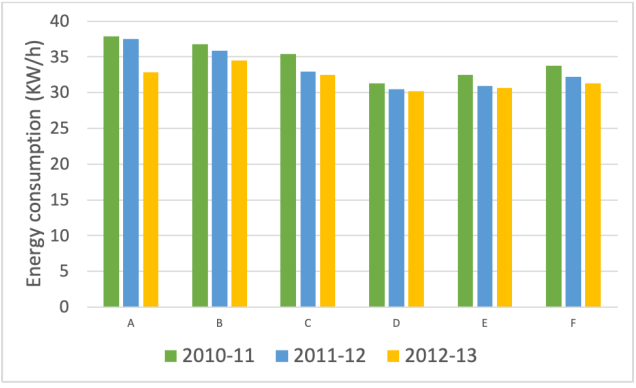

FIGURE 10. Mean hourly demand of six communities over the year.

B. DATA EXPLORATION

Fig. 10 and Fig. 11 show the average values of hourly energy demand and supply in Kilowatt-hours by the six communities (A to F) over the year.

Overall, the households in Sydney (A, B, and C) consumed more electricity than the consumers in Newcastle (D, E, and F) in all three periods. Above 37 KW/h energy was spent in Community A between 2010 and 2011. This amount was the highest among other groups in different periods. Furthermore, both cities experienced a decreasing trend in energy

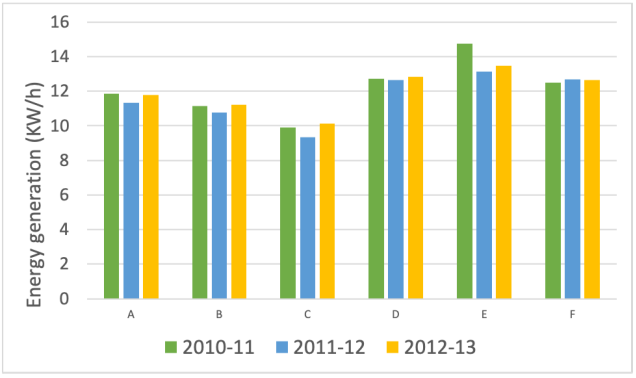

FIGURE 11. Mean hourly generation of six communities over the year.

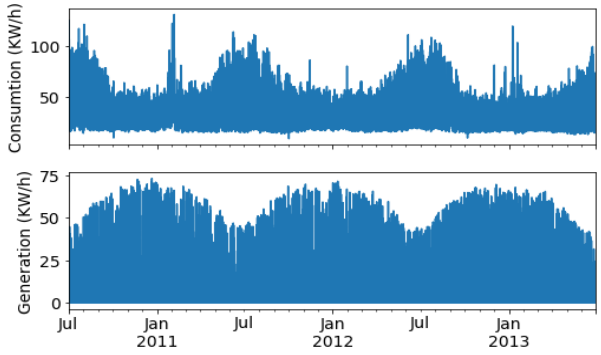

FIGURE 12. Hourly energy consumption and solar output of community a over three years.

consumption from 2010 to 2013 (Fig. 10). In terms of solar output, however, the people in Newcastle, on average, produced more energy in different periods; 13 KW/h as opposed to 10 KW/h (Fig. 11).

Fig. 12. shows the consumption behaviour and production pattern of Community A as an example. There are up and down patterns in both consumption and generation profiles, as seen in the graphs, mainly replicated throughout the entire three-year cycle. We can also see that both profiles follow the time-of-year pattern in each period but in reversed directions. More precisely, during the cold months in Australia (May-September), with more usage of heating systems, the peak hourly electricity demands increases and reaches around 100 to 120 KW/h, whereas the solar energy generation decreases at least by 20 percent (from 50 to 40 KW/h) during the same period with lack of sunshine. Similarly, the hourly electricity demand over non-cold periods (October-April) is reduced by half and reaches 50 KW/h in most intervals, whereas the energy generation grows continuously when in January reaches its highest amount; 60 KW/h.

The effect of time variables such as the month of year and day of the week on the load data is more visible in Fig. 13 and Fig. 14, where the mean aggregated loads of the houses (in Community A) vary by month and day with different degrees. According to Fig. 13, peaks in mean demand are seen around winter months (June, July, August) and troughs around March, April and October. The day-of-week pattern,

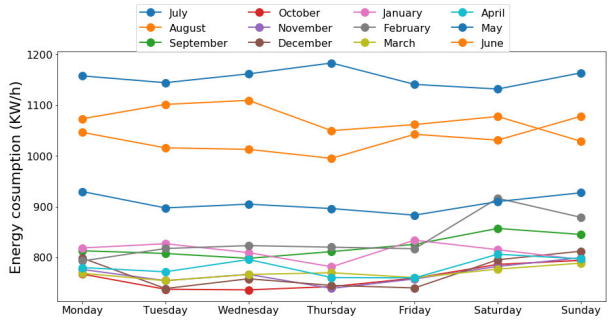

FIGURE 13. Average energy consumption by month and day.

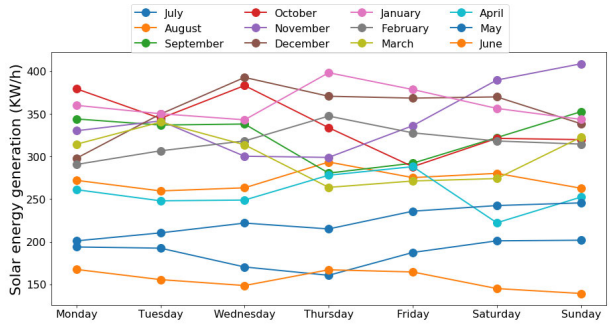

FIGURE 14. Average energy generation by month and day.

however, is similar for all months except for July when the consumption reduces in the weekends. On the other hand, Fig. 14 indicates the maximum energy output levels in the summer months, such as December and January. Similar to the consumption plot, there is no regular pattern based on days of the week for solar energy generation.

The analysis of time variables reveals the effect of outside air temperature and, most likely, other meteorological factors on household consumption patterns and solar cell outputs. As stated in the introduction section, weather parameters in the issue of energy forecasting have also been demonstrated in the literature. Related meteorological data was collected for our analysis via the Australian Government Bureau of Meteorology [59] and added to the aggregated load dataset.

Fig. 15 and Fig. 16 provide two examples of weather data analysis on the load profiles. Fig. 15 shows the demand during non-working hours plotted against outside temperature on both weekdays and weekends. The non-linear relationship indicates the importance of current air temperature in load demand prediction. The usage of air-conditioning systems for temperatures above 20 °C slightly lifts the demand, while for lower temperatures around 10 °C, heating systems increase the demand considerably.

In Fig. 16, however, we can see the positive correlation between air temperature and solar output (during daytime). It also shows how various amounts of Global Horizontal Irradiance affect solar energy production. During hot days where the temperature is mostly above 20°, large concentrations of GHI are detected. Since the number of sunny days during the observing period is smaller, the overall amount of energy

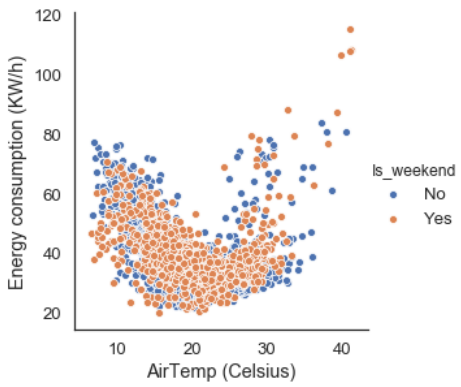

FIGURE 15. Aggregated load demand against air temperature.

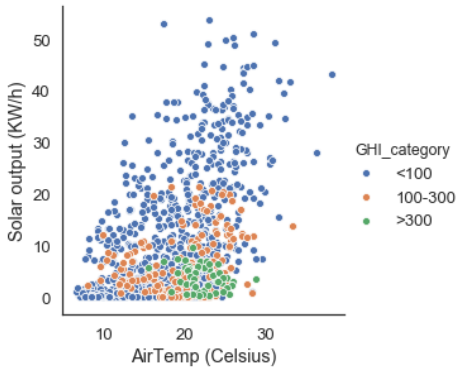

FIGURE 16. Aggregated solar output against air temperature.

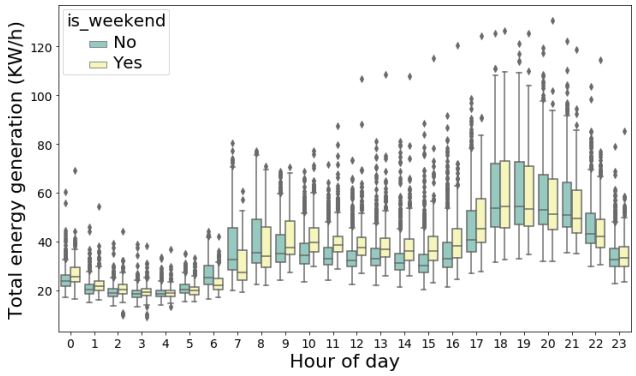

FIGURE 17. The hourly consumption distribution.

provided by solar panels is lower than that produced by those days with a lower or moderate temperature.

In Fig. 17 and Fig18 we also looked at how demand and generation vary with the time of day and the weekend. Here, Hour 0 corresponds to 12 am to 1 am, Hour 1 corresponds to 1 am to 2 am, and so on. It can be seen that there are significant differences between daytime and night-time patterns of the two plots. As expected, the household peak consumption occurs during afternoon and evening while the generation peak happens around noon. Moreover, in both graphs, the weekend effect is relatively large, specifically during the daytime between 9.00 and 17.00.

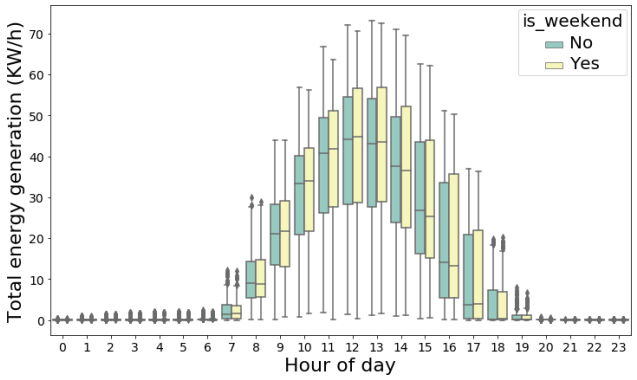

FIGURE 18. The hourly generation distribution.

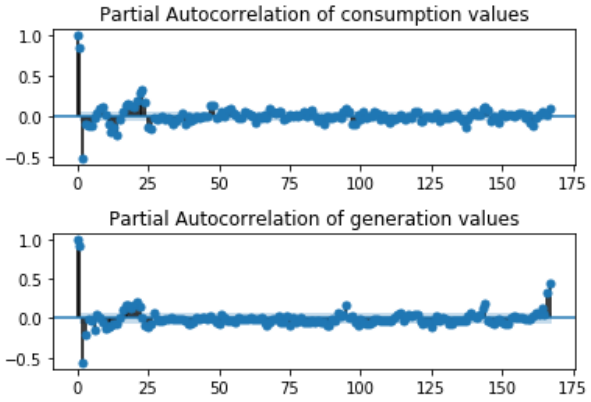

FIGURE 19. Partial auto-correlation plots.

Future energy values can also be strongly correlated with the amount of energy spent or produced over previous steps in time. Fig. 19 illustrates the Partial Autocorrelation Correlation plots of the time series of hourly energy consumption and generation. The maximum lag step used for calculating PACs is set to $24 \times 7 = 168$ hours (i.e. one week) and the confidence interval is set to 85% shown by the light blue lines. We can see that in both graphs that PACs' absolute values mostly exceed the significance level up to 24 lags. This implies that the energy values at 24 previous hours are highly correlated with the energy values at the current hour.

C. DATA CLEANING AND FEATURE ENGINEERING

In the initial dataset, irregular profiles with a significant amount of incomplete records had been already removed. For our use, three additional cleaning activities were carried out on the dataset: One was to delete one consumer data from the Sydney dataset due to a large number of missed measurements for the three-year study period; The second was filling in missing values for the days with short gaps e.g. one to six hours. The third was to correct the record values indicating zero aggregated energy consumption, implying a situation that would be almost impossible in real-world scenarios. As mentioned in Section II, Part (2); we used EMA to replace the missing measurements with valid values. Note that zero aggregated solar production values are not assumed missing values since sun rays will be zero at night time. Therefore, through training and testing of the models, they remained unchanged.

As mentioned in Section II, Part (3); we created a candidate set of predictive features: historical energy load measurements, time and calendar variables, along with meteorological features. Historical load variables refer to energy consumption and production of households at previous hours. The initial number of historical values is set to 24, as described in Section III and demonstrated in Fig. 19. Time and calendar variables include 'Hour of the day', 'Time of the year', 'Is weekend', and 'Is holiday'. The weather data include the following variables: Cloud Opacity %, Diffuse Horizontal Irradiance (DHI) in W/m2, Direct Normal Irradiance (DNI) in W/m2, Global Horizontal Irradiance (GHI) in W/m2, Solar Zenith angle (Zenith) in degree, Air Temperature in °C, Wind Speed in m/s, Wind Direction in °, Relative Humidity in %, and Precipitable Water in kg/m2.

Through the feature transformation process, to create more meaningful inputs, the values of 'Hour of the day' and 'Time of the year' attributes were converted to integer values produced by *Sin* and *Cos* functions. This transformation represents the daily and yearly periodicity of load profiles in a more effective way. Additionally, the two categorical attributes representing weekends and holidays were formulated as binary features with the One-hot encoding method.

The meteorological features,'Wind Direction' and 'Wind speed' were converted to more meaningful variables for the forecasting algorithms. Generally, wind direction in units of degrees is not considered as informative model input. For instance, 360° and 0° are close to each other and wrap around smoothly. If the wind is not blowing, then the direction should not matter. Therefore, we converted the 'Wind Direction' and 'Wind Speed' attributes to a wind vector with X and Y coordinates according to the following equations:

$$Wind\ (in\ radian) = WindDirection \times pi/180 \tag{15}$$

$$Wind\ X = WindSpeed \times \cos(Wind(in\ radian)) \tag{16}$$

$$Wind\ Y = WindSpeed \times \sin(Wind(in\ radian)) \tag{17}$$

Among the three features related to the sun radiation reaching the earth's surface, 'GHI' is of particular interest to photovoltaic installations and based on this formula $DNI \times \cos\theta + DHI$, it includes both 'DNI' and 'DHI' attributes; where θ refers to the angle of incidence of the beam. It is conceived that GHI can convey most information about its components to the model. Thus, to prevent the detrimental impact of feature redundancy on the performance of the predictive models, GHI remained in the feature set, and the other two were skipped.

As mentioned in Section II, Part (5); we selected a subset of appropriate features from the candidate set through feature selection methods. The original feature set consists of 17 variables: 15 non-load variables, weather and time parameters

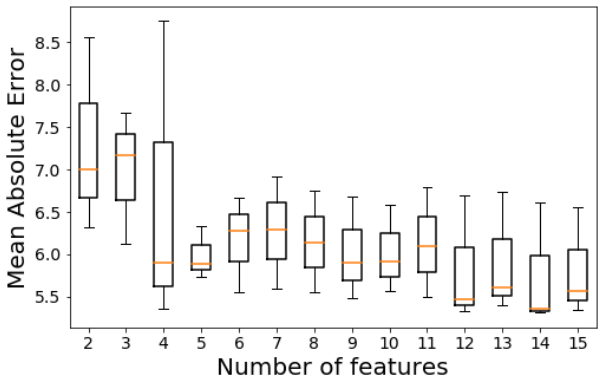

FIGURE 20. Error distribution by number of features for consumption estimation.

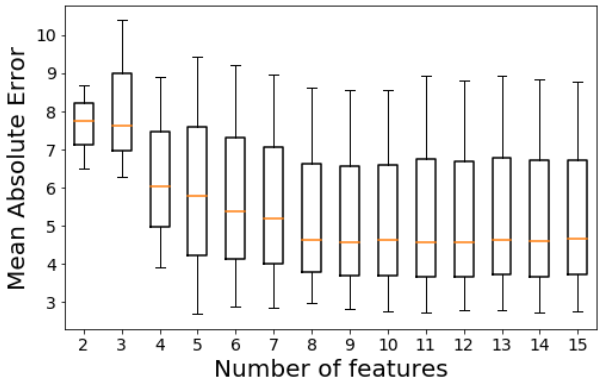

FIGURE 21. Error distribution by number of features for production estimation.

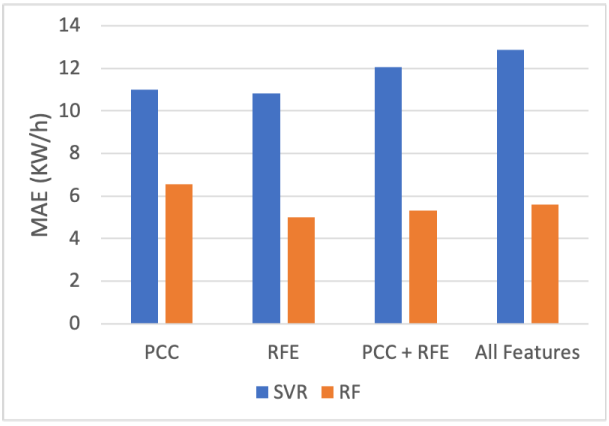

FIGURE 22. Average MAE using four feature sets for energy production.

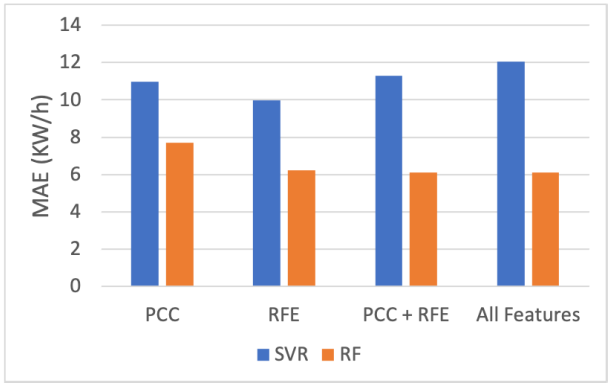

FIGURE 23. Average MAE using four feature sets for energy consumption.

along with two load variables, electricity consumption, and solar generation. Since we want to filter out non-load features from the original candidate set, we evaluate the feature selection methods using only non-load variables. We used one-year of training data with an hourly resolution to do the FS experiments.

The PCC method was designed to select the best n variables from the 15 predictors according to the correlation values higher than a threshold of $+/-0.3$. For the RFE method, the Random Forest algorithm was utilized to identify m number of best attributes out of 15. In practice, the value of m is not known in advance. Therefore, in the first step, different values for the number of features were evaluated using the training data and a K-fold cross-validation technique for time series data with K equal to three. The temporal order of data is complied with in this cross-validation technique so that the model is tested on observations that have not been used as training data.

Fig. 20 and Fig. 21 demonstrate the distribution of mean absolute error (MAE) values for each configured number of input features (The MAE metric is further defined in Equation (18)). We can see that performance improves as the number of features increases. However, the reduction in MAE values continues until reaching a certain number in both graphs. By growing the input space, the median values (shown by orange lines in the boxplots) fluctuate and show no significant improvements. It implies mostly up to eight or nine variables for both response variables can be relevant and influential.

In the next step, we used two conventional prediction algorithms known as support vector regression (SVR) and Random Forest (RF) to evaluate the prediction performance resulted by (1) the two FS techniques, (2) the combination of both, and (3) with all features (without FS approach). Fig. 22 and Fig. 23 show the average MAE results over three folds regarding predicting the targets at one step ahead.

As shown, all feature selection methods show better MAE performance when using the RF model. The lowest MAE error for both SVR and RF is obtained by the RFE method for solar output prediction (Fig. 22). However, when it comes to consumption prediction, there is no unique variable set that produces the highest accuracy for both models (Fig. 23). As a result, the RFE-driven features which resulted in low MAE errors for both targets and by both predictive algorithms were chosen as the final feature set. Table 1 lists the final set; 12 non-lead variables out of 15.

IV. FORECASTING EXPERIMENTS AND RESULTS

In this section, we describe the experiments conducted in Step Two and Step Three and analyze the forecasting results.

TABLE 1. List of non-load features.

Feature name	Value
Global Horizontal Irradiance	W/m2
Solar Zenith angle	Degree
Relative Humidity	Percentage
Precipitable Water	Kg/m2
Cloud Opacity	Percentage
Air Temperature	Celsius
Wind vector	$Wind\,Speed * Cos(Wind\,Direction)$
	$Wind\,Speed * Sin(Wind\,Direction)$
Time of day	$Sin(2 * \pi)/24 * hour of day$
	$Cos(2 * \pi)/24 * hour of day$
Time of year	$Sin(2 * \pi)/365 * 24 * hour\ of\ day$
	$Cos(2 * \pi)/365 * 24 * hour\ of\ day$

A. IMPLEMENTATION ENVIRONMENT

All models have been implemented using the Scikit-learn open-source machine learning library and the Keras framework for deep learning. The experimental hardware environment is based on a 3.1 GHz Intel (R) Core i5 CPU and 16 GB of memory.

B. DATA SCALING AND INPUT REQUIREMENTS

Since both conventional and deep neural networks are sensitive to the input scale and are more efficiently trained with normalized data, the data was normalized to the range [0, 1] by applying the Min-Max function. All predictive algorithms except for Persistence and ARIMA used the normalized input comprising the final set of variables; 12 non-load features presented in Table 1 along with one or two load (energy) features. ARIMA and Persistence only used historical energy data to predict future values of the two targets.

For the Ridge regression, SVR, AdaBoost, GBRT and BPNN, each input window explained in Part (6) of Section II, was transformed into a flattened format of $L * (N - 1)$, where L is equal to 24 lags and $N - 1$ is equal to 13 as the total number of variables (12 non-load variables selected through FS process and 1 load variable). As mentioned in Section II, Part (7), reducing one load variable from the input with the size of N indicates that the model that predicts future values of one target removes the values of other target from its input.

All deep models, on the other hand, are fed with 2-D input windows containing all $N = 14$ variables. Since they are capable of producing two outputs at the same time they are fed with both load varibles in addition of other non-load features. As indicated in Fig. 19, the length of input windows L was initialized as 24 for all the experimental models in Step Two. At the same time, this value was tuned as the hyperparameter of the final forecasting algorithm in Step Three. The forecast horizon H as the length of output windows was set to 24 for all experiments.

C. MULTI-STEP AND MULTIVARIATE FORECASTING STRATEGY

To predict the observations at multiple time steps, we applied two strategies depending on the training model: (1) direct multi-step forecast strategy for the models which naturally do not support multi-output regression including Ridge regression, SVR, AdaBoost, and GBRT; For each forecast time step, one model is developed (e.g. 24 models for 24 steps ahead). Furthermore, since the given models are not adopted for predicting multivariate time series, a separate model was trained and evaluated for each prediction target; one for energy consumption estimation and another for energy generation prediction. For this reason, the number of variables in their corresponding input was reduced from N to N − 1 where only one load variable corresponding to the target remained in the input array, and the other one was removed.

(2) Multiple output strategy for the models capable of performing multi-output regression including BPNN, CNN, LSTM, and Seq2Seq LSTM. This strategy involves the development of one model capable of predicting the entire forecast sequence at once. Unlike BPNN that considers the inputs as independent variables, the rest of ANN-based models, due to their learning procedure, can learn the dependency structure between inputs and outputs as well as between outputs. Therefore, they become more complex and are expected to perform better with sufficient training data. Moreover, as the deep models can produce multivariate outputs, individual deep models were trained to produce the two targets' time series at once.

D. PARAMETER SETTINGS

We mostly used the default values considered in the Scikit-learn library for the conventional ML models in terms of model parameter configuration. However, we modified the values of some of the parameters specified in Table 2, according to the input and output specifications and the complexity of the prediction task.

For the recurrent deep models, the same parameters of neural network architecture configuration (Table 2) were used; for the LSTM and GRU 50 neural unit in one hidden layer and for the Seq2Seq LSTM, the same number of neurons was used in both encoder and decoder each including one LSTM layer. For BPNN, to have a fair comparison with deep models, we increased the number of hidden layers to two with 60 and 30 neural units besides increasing the number of training epochs from 80 to 100. For all ANN-based models, 'Relu' [60] was applied as the activation function of hidden layers, mean square error (MSE) was used as the loss function, and 'Adam' function [61] was set as the model optimizer. The batch size sets to 32, the learning rate sets to 0.001, and the drop out rate sets to 0.1 (excluding BPNN and CNN, which did not use the dropout layer).

E. EVALUATION METRICS

To evaluate the prediction results of the trained algorithms, two commonly used error metrics in time series forecasting are used; Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). Lower values of error metrics indicate a more accurate prediction. The MAE measures the difference between predicted and real values on average and ignores whether the prediction values are greater than or smaller than the actual values. The RMSE, in contrast, penalizes large

TABLE 2. Parameter settings of experimental models.

Model name	Parameter	Value
ARIMA	p,q,r	2,2,0
Ridge Regression	Regularization parameter(Alpha)	1.0
SVR	C, Gamma, Epsilon, Kernel fucntion	100, 0.1, 0.1, RBF
AdaBoost,GBRT	Maximum depth of the tree, number of trees, Min samples split , Min sample leaf	3, 150, 2, 2
BPNN	Default hidden layer, Units in hidden layer, Training epochs, Batch size, Loss function, Optimizer, Activation function of hidden layer	2, (60,30), 100, 32, mse, Adam, Relu
CNN	Conv-layer, Kernel size, Filter size, Training epochs, Batch size, Loss function, Optimizer	2, 2, 32, 80, 32, mse, Adam
LSTM,GRU	Default hidden layer, Units in hidden layer, Training epochs Batch size, Drop out, Loss function, Optimizer, Activation function of hidden layer	1, 50, 80, 32, 0.1, mse, Adam, Relu
Seq2Seq LSTM	Default hidden layer (Encoder), Default hidden layer (Decoder), Units in hidden layers, Training epochs, Batch size, Drop out, Loss function, Optimizer, Activation function of hidden layers	1, 1, 50, 80, 32, 0.1, mse, Adam, Relu

errors before averaging them by computing the square error. These two metrics are defined as follows:

$$MAE = \frac{\sum_{t=1}^{N} |\hat{y}_i - y_i|}{N} \tag{18}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N}} \tag{19}$$

where y_i and $\hat{y}_i$ denote actual and predicted output at time step t and N denotes the number of sample observations in the testing period.

Since the focus of this work is on multi-step forecasting, we also added another metric. We named it as *SDE* to evaluate the consistency of the errors throughout the whole forecast horizon. The *SDE* metric computes the standard deviation of mean error values in each time step throughout the forecast horizon and is defined as follows:

$$SDE = \sqrt{\frac{\sum(\overline{E_i} - \mu)}{N}} \tag{20}$$

where $\overline{E_i}$ denotes the average error values over timestep i in the forecast horizon, μ denotes the mean of $\overline{E_i}$ and N denotes the total number of steps in the forecast horizon i.e 24. The lower value of *SDE* indicates lower variations and consequently, more stability at multi-step ahead forecasting.

F. EXPERIMENTS IN STEP TWO

To build and evaluate all models in Step Two, a sub-sample of training data (60%) was considered sufficient equal to nine yearly periods of hourly observations from different communities. A cross-validation methodology known as blocked cross-validation was applied instead of a standard train-test split to avoid overfitting and measuring each model's performance more robustly.

In this technique, which is designed for time-series data, the sample training set is split into n non-overlapping subsets. At the first iteration, the first subset is further divided into two-folds on the condition that the validation set is always ahead of the training set. For the next iteration, the next subset is again divided into two folds, and the iterations continue until n times. As a result, the temporal dependencies between observations are preserved during testing, and also no leakage from future data is introduced to the model. In each iteration, the model will not observe and memorize patterns from an iteration to the next. In this study, the number of iterations was set to three and the division rates for each subset are set to 80% for train fold and 20% for validation fold. This split method is depicted in Fig. 24 for more clarification.

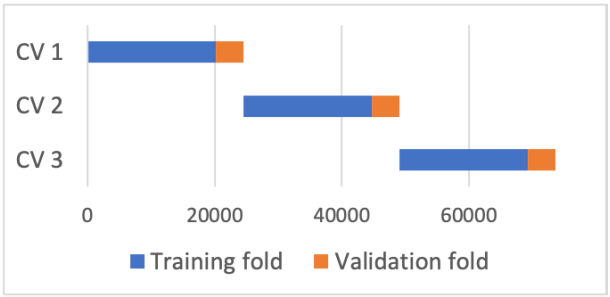

FIGURE 24. Blocked cross-validation with three splits.

The vertical axis refers to the number of cross-validation iterations, whereas the horizontal axis represents the size of training data on an hourly basis. The training folds are depicted in blue, and the folds used for validation are depicted in orange. The dataset has not shuffled, and the chronological order is preserved along the horizontal axis.

G. RESULTS ANALYSIS OF STEP TWO

Table 3 and Table 4 provide multi-step a head forecasting results of different models using the cross-validation technique for two prediction targets. In each table, the three columns on the left report average RMSE errors in KW/h over 24 time steps per validation fold. The last column on the right provides the average RMSE errors over the three folds with standard deviation values.

The experimental results show that different learning algorithms outperform the Persistence technique showing at least 40% to at most 80% improvement in prediction accuracy. We can see the Seq2Seq LSTM followed by GBRT outperform the other techniques in terms of energy consumption estimation.

Regarding energy production forecasting, the lowest prediction error is achieved by the GBRT and GRU. Further analysis reveals that RMSE of deep neural network models

TABLE 3. Average RMSE using the blocked cross-validation technique for energy consumption prediction.

Model	Fold 1	Fold 2	Fold 3	Over three folds
Persistence	24.09	30.29	34.86	29.75 +/-4.41
ARIMA	16.9	19.09	18.94	18 +/- 0.89
Ridge Regression	6.60	6.42	6.84	6.62 +/- 0.17
SVR	7.97	8.06	9.13	8.93 +/- 0.52
AdaBoost	11.02	10.43	11.13	10.86 +/ -0.30
GBRT	5.80	5.67	6.32	**5.93** +/- 0.28
BPNN	6.57	6.74	6.62	6.64+/-0.34
CNN	5.88	6.30	6.30	6.16 +/- 0.19
LSTM	5.78	5.85	6.43	6.02 +/- 0.52
GRU	5.76	6.08	6.41	6.09 +/- 0.26
Seq2Seq LSTM	5.92	5.85	6.30	**5.91** +/- 0.29

TABLE 4. Average RMSE using the blocked cross-validation technique for energy production prediction.

Model	Fold 1	Fold 2	Fold 3	Over three folds
Persistence	34.99	30.56	34.58	33.8 +/- 1.99
ARIMA	12.03	11.55	12.4	12+/-0.34
Ridge Regression	6.43	5.66	6.72	6.27+/-0.44
SVR	7.68	4.44	7.15	6.42 +/- 1.42
AdaBoost	9.41	7.72	10.75	9.29 +/ -1.23
GBRT	5.77	4.88	5.97	**5.41** +/- 0.65
BPNN	6.34	5.38	6.32	6.01+/- 0.44
CNN	6.24	5.01	5.99	5.74 +/ -0.52
LSTM	6.71	4.20	5.87	5.59 +/- 0.93
GRU	6.46	4.12	5.76	**5.45** +/- 0.98
Seq2Seq LSTM	6.86	3.90	5.85	**5.54** +/ -0.29

TABLE 5. Total training time for 24-steps ahead prediction of two targets.

Model	Total training time (Seconds)
ARIMA	1100 *2 = 2200
Ridge Regression	3*2 = 6
SVR	171 *2 = 342
AdaBoost	1480 *2 = 2960
GBRT	670 *2 = 1340
BPNN	38 *2 = 76
CNN	90
LSTM	420
GRU	400
Seq2Seq LSTM	830

(e.g. Seq2Seq LSTM, LSTM, CNN) are similar and lower than that of shallow neural network (BPNN) as well as most conventional learning models (e.g. ARIMA, Ridge Regression, and SVR). Among the deep models, the Seq2Seq LSTM yields low average RMSE errors (less than 5 KW/h) with a low standard deviation (0.29) for both demand and PV output forecasting indicating more accurate and robust performance against other deep models.

To further investigate each method's effectiveness in terms of computational cost, we have computed the average training time over three folds. Each training fold covers around two years and three months of hourly data and occupies about 2.6 Mega byte of the system memory. Note that for the models that do not support multivariate regression, the training time is multiplied by two to represent the required training time to forecast the two energy targets. The persistence model is ignored in this evaluation since it does not pass any training phase. Table 5 presents the results.

ARIMA and Adaboost not only produce inaccurate hourly forecasts but also require long training times. Ridge regression followed by BPNN is substantially faster among all methods. SVR compared to deep models needs higher training time. While GBRT algorithms appear to have good precision (according to Table 4), relative to other methods, they are slower. Among deep neural networks, CNNs are the fastest, followed by GRU and LSTM networks.

In conclusion, among the candidate techniques, Seq2Seq LSTM and GBRT were chosen as the most promising models for building the ensemble model in the next step. Despite being slower than other algorithms, these models have shown higher forecasting accuracy for both predictive targets. To accelerate the training procedure, GPU-based computing can be adopted in real-world scenarios.

H. EXPERIMENTS OF STEP THREE

1) ENSEMBLE SETTING

Having access to an extensive training set (14 years of hourly data from 6 communities) inspired us to use the deep Seq2Seq LSTM models as the first-layer predictors or base learners and the GBRT algorithm as the meta learner to capture non-linear relations between base predictions. To build an ensemble model with large diversity, multiple Seq2Seq-LSTM networks were developed and parameterized differently. The parameters that contributed to the ensemble's diversity include learning rate, number of hidden layers in Encoder, number of hidden neurons in Encoder and Decoder, type of layer in Encoder, and the length of the input sequence (windows) W.

For evaluation, we chose $W = [24, 24 * 2, 24 * 3]$ as the length of input windows, and for each of the other parameters, a set of values was considered from a candidate set V. As a result, we ended up with $W * V$ Seq2Seq LSTM models for each given parameter. For our experiments, the V vector for each given parameter is set to values as the following: learning rate: {0.01,0.001}, the number of hidden layers in Encoder: {1,2}, the number of hidden neurons in Encoder and Decoder: {60,90} and the type of Encoder layer: {LSTM, BILSTM}. As default values, we chose ADAM as an optimizer, mean squared error as the loss function, batch size of 64 and 20 iterations as the number of training epochs per network.

For the stacking purpose, the forecasts from the first-layer predictors are treated as input features for the GBRT. As we want to predict future values of two variables (energy consumption and generation), and the GBRT does not support multivariate output regression, one GBRT model was trained per target based on the corresponding predictions from the first layer. The performance of meta learners was then evaluated against real observations available to us within the test set.

All four test sets (mentioned in Section III, Subsection A) were used to evaluate and demonstrate the prediction capabilities of the ensemble approach. Each test set representing a community load for one year was divided into two subsets with 70% and 30% ratios known as a meta train set and a meta test set.

TABLE 6. Average error metrics over multiple steps ahead for energy consumption prediction.

Average MAE												
	Test set 1			Test set 2			Test set 3			Test set 4		
Model/Horizon	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)
Best base learner (Seq2Seq LSTM)	4.15	4.09	4.10	3.77	3.85	3.87	3.22	3.27	3.23	3.70	3.85	3.74
Averaged ensemble	4.06	4.31	4.33	3.69	3.90	3.92	4.24	4.17	4.20	3.82	3.90	3.88
Stacked ensemble (Seq2Seq LSTM + Ridge)	3.63	3.80	3.76	3.49	3.66	3.65	**3.01**	3.09	3.13	3.49	3.60	3.61
Stacked ensemble (Seq2Seq LSTM + GBRT)	**3.59**	**3.73**	**3.72**	**3.42**	**3.57**	**3.59**	3.03	**3.01**	**3.05**	**3.31**	**3.34**	**3.28**
Average RMSE												
Best base Learner (Seq2Seq LSTM)	5.61	5.59	5.72	5.06	5.31	5.32	4.18	4.26	4.23	4.81	4.98	4.89
Averaged Ensemble	5.58	5.96	6.09	5.10	5.45	5.60	5.40	5.29	5.33	4.97	5.05	5.07
Stacked Ensemble (Seq2Seq LSTM + Ridge)	4.83	5.06	**5.04**	4.65	4.90	**4.92**	**3.97**	4.03	4.12	4.55	4.65	4.68
Stacked Ensemble (Seq2Seq LSTM + GBRT)	**4.81**	**5.01**	5.05	**4.61**	**4.85**	4.94	4.05	**3.97**	**3.96**	**4.37**	**4.40**	**4.38**

TABLE 7. Average error metrics over multiple steps ahead for energy generation prediction.

Average MAE												
	Test set 1			Test set 2			Test set 3			Test set 4		
Model/Horizon	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)	(1-8)	(8-16)	(16-24)
Best base learner (Seq2Seq LSTM)	1.43	1.67	1.88	1.01	1.15	1.34	1.41	1.60	1.74	1.24	1.42	1.56
Averaged ensemble	1.46	1.79	2.13	1.03	1.30	1.60	4.05	4.15	4.38	1.42	1.70	2.11
Stacked ensemble (Seq2Seq LSTM + Ridge)	1.13	1.33	1.55	0.88	1.05	**1.20**	1.19	1.31	1.55	1.02	1.15	1.35
Stacked ensemble (Seq2Seq LSTM + GBRT)	**1.04**	**1.29**	**1.52**	**0.87**	**1.04**	1.24	**1.12**	**1.26**	**1.46**	**0.98**	**1.13**	**1.33**
Average RMSE												
Best base Learner (Seq2Seq LSTM)	2.65	3.19	3.74	2.01	2.35	2.77	2.63	3.09	3.36	2.36	2.75	3.05
Averaged Ensemble	2.74	3.33	3.95	1.98	2.47	2.99	7.50	7.60	8.04	2.62	3.09	3.94
Stacked Ensemble (Seq2Seq LSTM + Ridge)	2.33	**2.83**	**3.32**	**1.83**	2.19	**2.55**	2.24	**2.57**	3.09	2.02	**2.33**	**2.79**
Stacked Ensemble (Seq2Seq LSTM + GBRT)	**2.28**	2.86	3.39	1.87	2.26	2.69	**2.22**	2.59	**3.06**	**2.01**	**2.33**	2.82

The meta train set was used to build a training set for the meta learner (GBRT) so that all trained LSTM base learners in the first step were tested on this set. The Meta test set was then used to evaluate the performance of the meta learner against three other techniques:

- The Seq2Seq LSTM network that yields the lowest prediction error among the individual learners (called the best learner).
- The ensemble that computes the average of the base learners' predictions for each step in the forecasting horizon.
- Another stacked ensemble of Seq2Seq LSTM networks which applies Ridge regression as the meta learner to capture linear relations between the first layer's forecasts.

It is worth mention that none of the meta test sets has been seen by any learning algorithm during training.

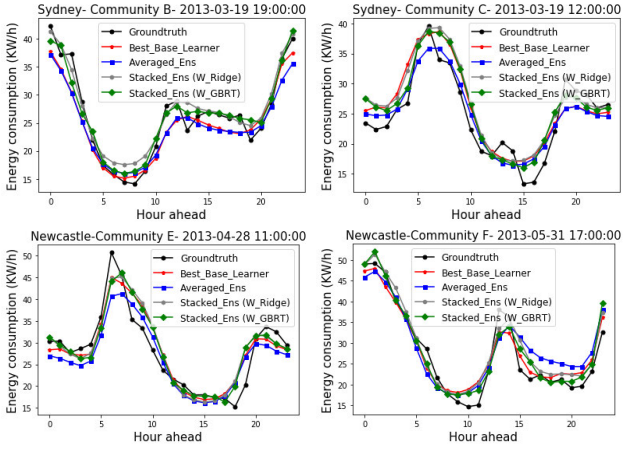

FIGURE 25. Comparison of 24-h ahead load consumption forecasting results.

2) RESULT ANALYSIS OF STEP THREE

Fig. 25 and Fig. 26 provide example comparisons between the actual (Ground-truth) and predicted energy load curves using various approaches in four test communities. We can see that in all graphs, on some test steps, the forecast precision decreases due to irregular fluctuations in hourly load, especially for load demand prediction. Nevertheless, in most cases, all models, specifically the two ensemble approaches, could effectively follow the usage and generation patterns for both forecasting targets.

Table 6 and Table 7 summarise the forecasting results of different methods. For each test set, the average MAE and

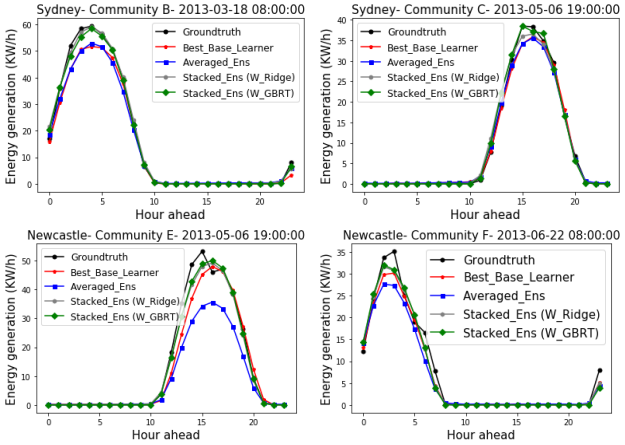

FIGURE 26. **Comparison of 24-h ahead energy generation forecasting results.**

RMSE are calculated for three horizons: short-term; from 1 to 8 hours ahead, medium-term; from 8 to 16 hours ahead, and long-term; from 16 to 24 hours ahead.

It is observed that the stacked ensemble with GBRT and Ridge algorithms have the lowest average MAE and RMSE overall forecast horizons and across all test sets for both energy targets. The prediction accuracy of electricity consumption and PV power output can be effectively improved by using ensembles of Seq2Seq LSTM networks. After using a conventional ML algorithm on top of them, the forecast accuracy can be further improved since more diverse first-layers' forecasts can provide more relevant information for model training. This implies that the ensembles with GBRT and Ridge regression can model regular and irregular patterns of future energy values more effectively.

In contrast, the performance of the averaged ensemble compared to the other three algorithms is not highly accurate and stable across different test sets. For instance, on Test set 3, the predictive errors for solar output prediction are far greater than those obtained by other algorithms. The explanation is that unlike stacked ensembles, the forecasting performance of the Averaged model is equally determined by all contributors' forecasts. With even one poor base estimator, performance will degrade significantly.

Fig. 27 and Fig. 28 illustrate the prediction performance of each algorithm on average from 1 to 24 steps ahead per test set for the two targets. They confirm the results of previous experiments. Regarding energy consummation prediction, the averaged ensemble produces higher MAE (4.27, 3.86, 4.19, and 3.88) compared to the best base learner (4,14, 3.84, 3.25, and 3.77).

In terms of RMSE, the forecast error of the Averaged model (5.93, 5.41, 5.32, and 5.05) is also higher in comparison with the best LSTM network in the ensemble (5.68, 5.25, 4.23, and 4.91). The proposed forecast framework (Seq2Seq LSTM + GBRT) produces an average MAE of 3.39 and RMSE of 4.55 over four test sets, which are lower than those of all other models.

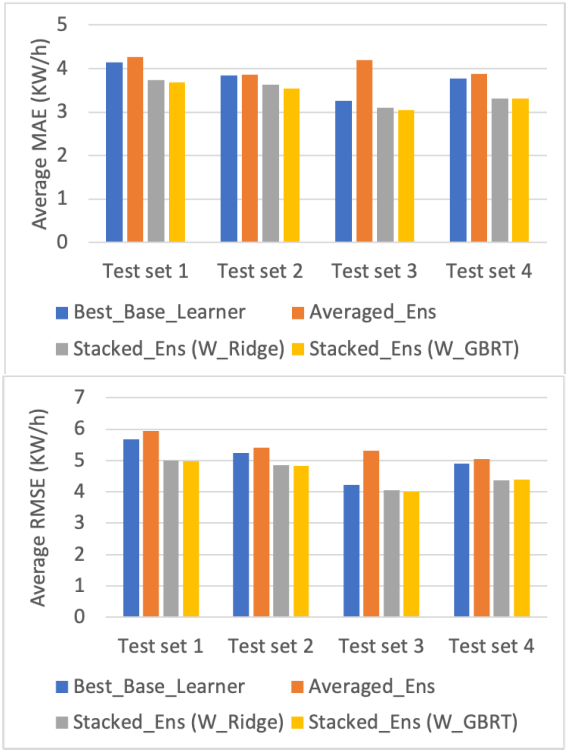

FIGURE 27. **Average error metrics for energy consumption prediction from 1 to 24 steps ahead.**

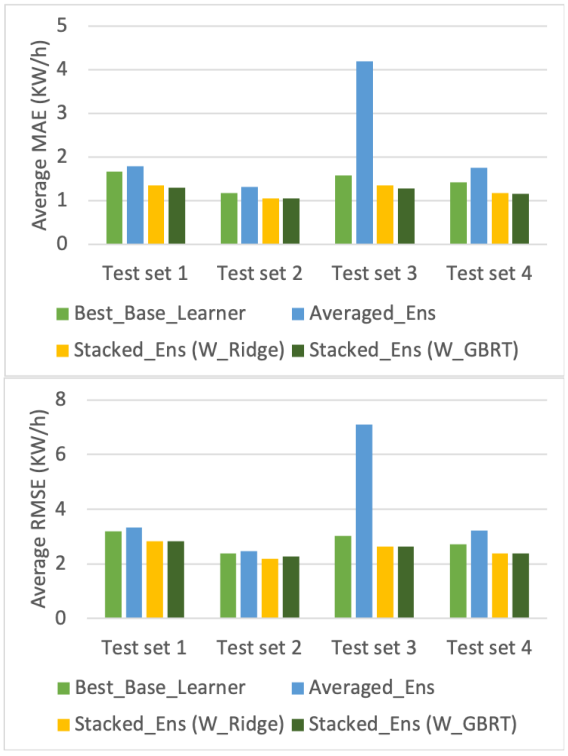

FIGURE 28. **Average error metrics for energy generation prediction from 1 to 24 steps ahead.**

Similar forecast accuracy is recorded for the PV power output of the communities. The best LSTM network gives

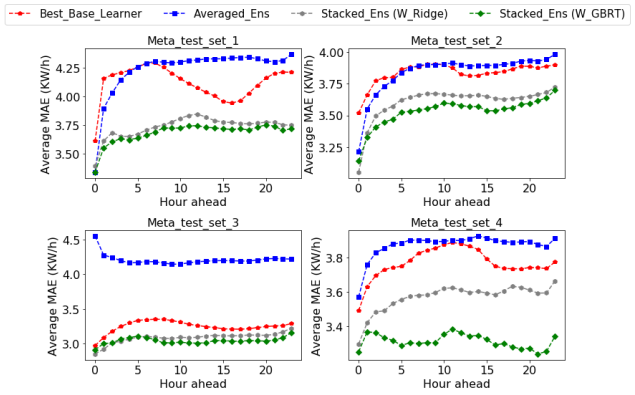

FIGURE 29. Average MAE for energy consumption prediction over each time step on four meta test sets.

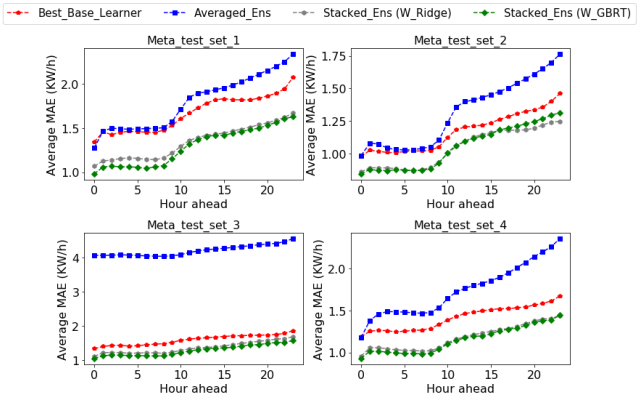

FIGURE 30. Average MAE for energy production prediction over each time step on four meta test sets.

more accurate results than the averaged ensemble forecast but lower compared to the stacked models. The stacked ensemble with GBRT slightly outperforms the Ridge-based stacked ensemble with 2.4% reduction in MAE score on average across the test sets. However, it significantly produces more accurate forecasts than the best learner and averaged ensemble, showing on average 17% and 47% reduction in MAE as well as 10% and 37% reduction in RMSE.

To evaluate the consistency of errors throughout 24 time steps, we calculated the average MAE values of the prediction results at each time step for different models.

Fig. 29 and Fig. 30 illustrate the results. We can see that the forecasting errors of all models fluctuate smoothly along the forecast horizon up to 10 steps and then increase with varying degrees for different models. The degree of the overall variation in terms of MAE was computed by *SDE* metric.

The SDE Results are presented in Table 8 and Table 9. The proposed ensemble shows higher consistency in multi-step ahead forecasting of energy consumption across all test sets with, on average, a 0.06 variation rate. Regarding solar output estimations, higher variation values are recorded for all models. On average, the best base learners followed by the two ensembles reach satisfactory *SDE* scores of 0.15, 0.16. and 0.17, respectively. However, as expected, the Averaged

TABLE 8. Comparison of SDE scores over 24 steps ahead for consumption estimation.

Model	Test set 1	Test set 2	Test set 3	Test set 4
Best base Learner (Seq2Seq LSTM)	0.14	0.08	0.08	0.08
Averaged ensemble	0.21	0.16	0.07	0.07
Stacked Ensemble (Seq2Seq LSTM + Ridge)	0.09	0.13	007	0.07
Stacked Ensemble (Seq2Seq LSTM + GBRT)	**0.08**	**0.11**	**0.049**	**0.03**

TABLE 9. Comparison of SDE scores over 24 steps ahead for production estimation.

Model	Test set 1	Test set 2	Test set 3	Test set 4
Best base Learner (Seq2Seq LSTM)	0.2	**0.14**	**0.14**	**0.14**
Averaged ensemble	0.3	0.25	0.15	0.3
Stacked Ensemble (Seq2Seq LSTM + Ridge)	**0.18**	**0.14**	0.16	**0.14**
Stacked Ensemble (Seq2Seq LSTM + GBRT)	0.20	0.16	0.15	0.15

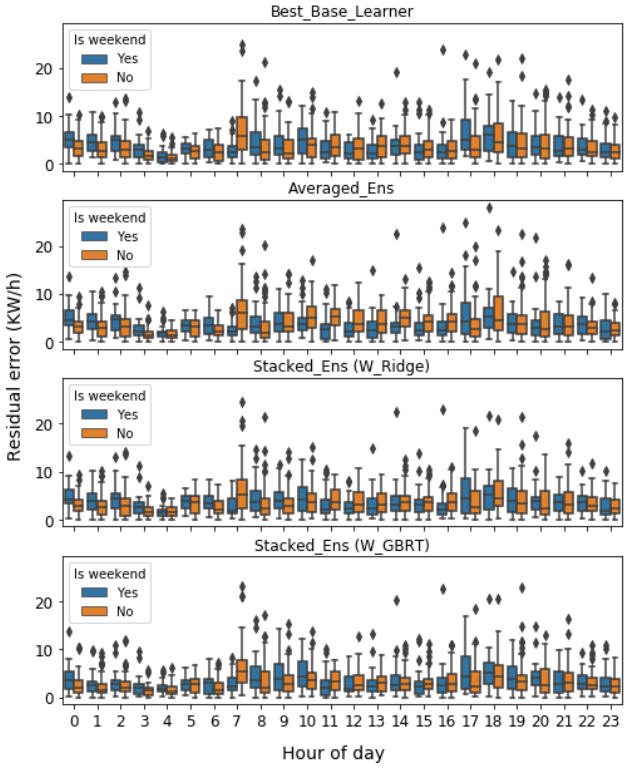

FIGURE 31. Error distribution of day-ahead load demand forecasting over weekdays and weekends.

Ensemble with high *SDE* values shows poor forecasting stability against the other techniques in most cases.

To further verify the superiority of the stacked models, the day-ahead (24 hours ahead) prediction results are analyzed under different situations. For load demand estimation, weekdays and weekends are considered, whereas, for solar power output prediction, two typical weather conditions are evaluated: Cloudy and non-cloudy days.

To identify cloudy days, we used the cloud visual opacity index, which describes how much sunlight the clouds let some sunlight through. The days with cloud opacity values less than

TABLE 10. Average residual error for day ahead forecasting of energy consumption based on type of day.

	Test set 1		Test set 2		Test set 3		Test set 4	
Model/Day type	Weekday	Weekend	Weekday	Weekend	Weekday	Weekend	Weekday	Weekend
Best base learner (Seq2Seq LSTM)	3.81	5.2	3.48	4.97	3.24	3.38	3.68	3.98
Averaged ensemble	3.77	5.8	3.43	5.38	4.29	4.01	3.95	3.79
Stacked ensemble (Seq2Seq LSTM + Ridge)	3.51	4.34	3.33	**4.72**	3.15	3.38	3.60	3.80
Stacked ensemble (Seq2Seq LSTM + GBRT)	**3.50**	**4.27**	**3.28**	4.75	**3.10**	**3.29**	**3.25**	**3.56**

TABLE 11. Average residual error for day ahead forecasting of energy generation for cloudy and non-cloudy days.

	Test set 1		Test set 2		Test set 3		Test set 4	
Model/Day type	Cloudy	Non-Cloudy	Cloudy	Non-Cloudy	Cloudy	Non-Cloudy	Cloudy	Non-Cloudy
Best base learner (Seq2Seq LSTM)	1.86	2.14	1.32	1.50	1.77	1.88	1.61	1.69
Averaged ensemble	2.08	2.41	1.63	1.80	3.32	4.94	2.34	2.36
Stacked ensemble (Seq2Seq LSTM + Ridge)	1.58	1.69	**1.08**	**1.30**	1.71	1.66	1.50	**1.41**
Stacked ensemble (Seq2Seq LSTM + GBRT)	**1.52**	**1.65**	**1.08**	1.38	**1.61**	**1.54**	**1.48**	1.43

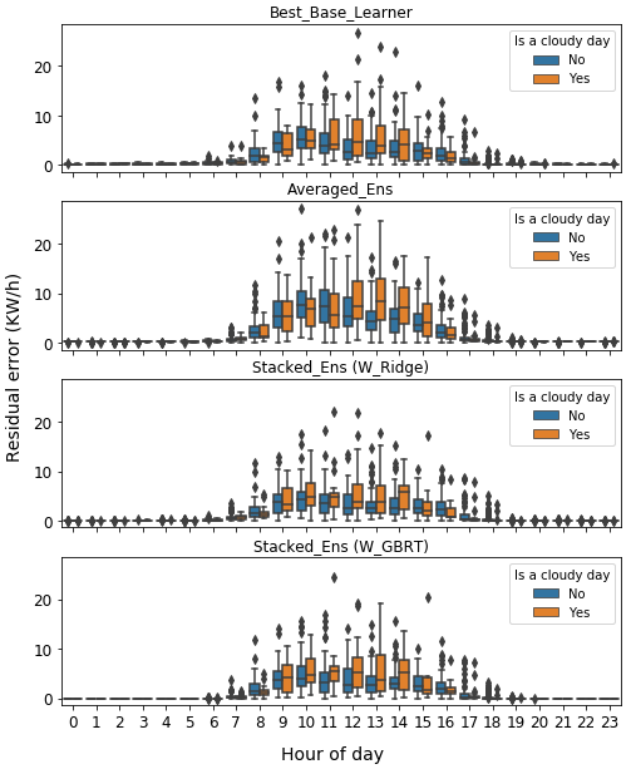

FIGURE 32. Error distribution of day-ahead solar output forecasting over cloudy and and non-cloudy days.

42 are considered as clear or partially cloudy days, while the days with higher index values are categorized as cloudy days.

Fig. 31 depicts the distribution of residual error of the proposed framework along with other comparative models regarding electricity consumption prediction of one community in Sydney. The residual error represents the difference between predicted and real values.

As expected, the median error values of all models mostly increase during peak hours between 7.00 to 10.00 in the morning and from 17.00 to 21.00 in the afternoon. Moreover, for all frameworks, smaller boxes of weekdays compared to weekends indicate fewer variations in forecasting results and, therefore, more predictability of usage patterns on weekdays.

Forecast residual error comparison indicates that the stacked predictors generate less error in comparison with the best individual predictor and Averaged ensemble, demonstrating the potential benefit of the stacking ensemble technique in the day-ahead load demand forecast application.

Table 10 summarizes the prediction errors of different models in different communities. The proposed framework is more accurate than the comparative forecast models by producing at most 3.50 KW/h error for weekdays and 4.75 KW/h for weekends.

Fig. 32 highlights the box forecast error plot of all models for day-ahead estimation of PV power output in cloudy and non-cloudy days in the same test community. It can be observed that the PV output prediction of each model is relatively less accurate on cloudy days, and this is primarily because the PV power curve on cloudy weather is less steady and more volatile.

Compared to stacked models, the averaged ensemble predictor followed by the best learner gives higher forecast errors during cloudy days. Among all, the proposed ensemble with GBRT has produced the most accurate results with fewer outliers.

In Table 11, the results are summarized for all models and test sets. As shown, the proposed method can mostly capture the regular and non-regular patterns of PV power output at a satisfactory level on cloudy and non-cloudy days across different communities.

V. CONCLUSION

In this paper, we proposed a framework for multi-hour ahead load forecasting and solar energy generation estimation of household communities. The framework introduces a process

in which an ensemble model is developed based on extensive evaluations of baseline forecasting algorithms. The ensemble model applies deep recurrent neural networks as base learners and a tree-based ensemble algorithm as meta learner. It also incorporates multivariate time series data, including energy, time, and weather variables as predictive features to address the volatility of load series.

The proposed method offers several advantages over existing techniques. Firstly applying an ensemble learning strategy enables the model to provide more robust and accurate results than individual predictive methods. Secondly, deep recurrent neural networks as strong predictive algorithms for time series prediction tasks, provide the model with highly accurate base estimations. Next, since the ensemble model is not reliant on the structure of one particular deep network, it can generalize better to new data sets than individual neural networks that are heavily tuned for a given dataset. Finally, unlike the boosting approach that involves sequential learning, the applied stacking strategy offers the ability to separately train base learners, thereby reducing training time in distributed computational environments. However, the main limitation of the proposed approach can be the lack of appropriate historical data for proper training of deep base models.

In future work, the performance of the ensemble technique and all contributing algorithms can be evaluated through sensitivity analysis where we examine the impacts of different input features or input size on the prediction task. The presented forecast framework could also be applied for other types of time series data such as wind and electricity price as long as a sufficient amount of data (typically one to a few years of hourly observations) for training the deep networks are available.

REFERENCES

[1] A. Jäger-Waldau, "Snapshot of photovoltaics," *Energies*, vol. 12, no. 5, p. 769, Feb. 2019.

[2] E. Sharma, "Energy forecasting based on predictive data mining techniques in smart energy grids," *Energy Informat.*, vol. 1, no. 1, p. 44, Oct. 2018, doi: 10.1186/s42162-018-0048-9.

[3] D. Sembroiz, D. Careglio, S. Ricciardi, and U. Fiore, "Planning and operational energy optimization solutions for smart buildings," *Inf. Sci.*, vol. 476, pp. 439–452, Feb. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025518304444

[4] C. Wan, J. Zhao, Y. Song, Z. Xu, J. Lin, and Z. Hu, "Photovoltaic and solar power forecasting for smart grid energy management," *CSEE J. Power Energy Syst.*, vol. 1, no. 4, pp. 38–46, Dec. 2015.

[5] Z. Wang, J. Li, S. Zhu, J. Zhao, S. Deng, S. Zhong, H. Yin, H. Li, Y. Qi, and Z. Gan, "A review of load forecasting of the distributed energy system," *IOP Conf. Earth Environ. Sci.*, vol. 237, Mar. 2019, Art. no. 042019, doi: 10.1088/1755-1315/237/4/042019.

[6] E. Sharma, M. Mussetta, and W. Elmenreich, "Investigating the impact of data quality on the energy yield forecast using data mining techniques," in *Proc. IEEE PES Innov. Smart Grid Technol. Eur. (ISGT-Europe)*, Oct. 2020, pp. 599–603.

[7] J. Zhong, L. Liu, Q. Sun, and X. Wang, "Prediction of photovoltaic power generation based on general regression and back propagation neural network," *Energy Procedia*, vol. 152, pp. 1224–1229, Oct. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876610218307173

[8] L. Liu, Y. Zhao, D. Chang, J. Xie, Z. Ma, Q. Sun, H. Yin, and R. Wennersten, "Prediction of short-term PV power output and uncertainty analysis," *Appl. Energy*, vol. 228, pp. 700–711, Oct. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261918309826

[9] J. Zhang, Y.-M. Wei, D. Li, Z. Tan, and J. Zhou, "Short term electricity load forecasting using a hybrid model," *Energy*, vol. 158, pp. 774–781, Sep. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S036054421831065X

[10] S. Sobri, S. Koohi-Kamali, and N. A. Rahim, "Solar photovoltaic generation forecasting methods: A review," *Energy Convers. Manage.*, vol. 156, pp. 459–497, Jan. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0196890417310622

[11] S. L. Wong, K. K. W. Wan, and T. N. T. Lam, "Artificial neural networks for energy analysis of office buildings with daylighting," *Appl. Energy*, vol. 87, no. 2, pp. 551–557, Feb. 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261909002669

[12] B. B. Ekici and U. T. Aksoy, "Prediction of building energy consumption by using artificial neural networks," *Adv. Eng. Softw.*, vol. 40, no. 5, pp. 356–362, May 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0965997808001105

[13] A. Yang, W. Li, and X. Yang, "Short-term electricity load forecasting based on feature selection and least squares support vector machines," *Knowl.-Based Syst.*, vol. 163, pp. 159–173, Jan. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705118304234

[14] N. Singh, C. Vyjayanthi, and C. Modi, "Multi-step short-term electric load forecasting using 2D convolutional neural networks," in *Proc. IEEE-HYDCON*, Sep. 2020, pp. 1–5.

[15] S. Masum, Y. Liu, and J. Chiverton, "Multi-step time series forecasting of electric load using machine learning models," in *Proc. ICAISC*, 2018, pp. 148–159.

[16] X. Shao and C. S. Kim, "Multi-step short-term power consumption forecasting using multi-channel LSTM with time location considering customer behavior," *IEEE Access*, vol. 8, pp. 125263–125273, 2020.

[17] Y. Yang, Z. Shang, Y. Chen, and Y. Chen, "Multi-objective particle swarm optimization algorithm for multi-step electric load forecasting," *Energies*, vol. 13, no. 3, p. 532, Jan. 2020. [Online]. Available: https://www.mdpi.com/1996-1073/13/3/532

[18] W. Li, X. Yang, H. Li, and L. Su, "Hybrid forecasting approach based on GRNN neural network and SVR machine for electricity demand forecasting," *Energies*, vol. 10, no. 1, p. 44, Jan. 2017. [Online]. Available: https://www.mdpi.com/1996-1073/10/1/44

[19] J. Zhang, Y. Chi, and L. Xiao, "Solar power generation forecast based on LSTM," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 869–872.

[20] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecasting—An approach using autoencoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 002858–002865.

[21] H. Lee and B.-T. Lee, "Bayesian deep learning-based confidence-aware solar irradiance forecasting system," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2018, pp. 1233–1238.

[22] A. Alzahrani, P. Shamsi, C. Dagli, and M. Ferdowsi, "Solar irradiance forecasting using deep neural networks," *Procedia Comput. Sci.*, vol. 114, pp. 304–313, Jan. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050917318392

[23] Z. Dong, D. Yang, T. Reindl, and W. M. Walsh, "A novel hybrid approach based on self-organizing maps, support vector regression and particle swarm optimization to forecast solar irradiance," *Energy*, vol. 82, pp. 570–577, Mar. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544215000900

[24] J. Antonanzas, R. Urraca, A. Pernía-Espinoza, A. Aldama, L. A. Fernández-Jiménez, F. J. Martínez-de-Pisón, "Single and blended models for day-ahead photovoltaic power forecasting," in *Hybrid Artificial Intelligent Systems*, F. J. M. de Pisón, R. Urraca, H. Quintián, and E. Corchado, Eds. Cham, Switzerland: Springer, 2017, pp. 427–434.

[25] M. W. Ahmad, M. Mourshed, and Y. Rezgui, "Tree-based ensemble methods for predicting PV power generation and their comparison with support vector regression," *Energy*, vol. 164, pp. 465–474, Dec. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544218317432

[26] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer, 2009.

[27] Y. Liu, W. Wang, and N. Ghadimi, "Electricity load forecasting by an improved forecast engine for building level consumers," *Energy*, vol. 139, pp. 18–30, Nov. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544217313348

[28] W. Kim, Y. Han, K. J. Kim, and K.-W. Song, "Electricity load forecasting using advanced feature selection and optimal deep learning model for the variable refrigerant flow systems," *Energy Rep.*, vol. 6, pp. 2604–2618, Nov. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484720313317

[29] C. Fan, F. Xiao, and S. Wang, "Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques," *Appl. Energy*, vol. 127, pp. 1–10, Aug. 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261914003596

[30] A. M. Pirbazari, A. Chakravorty, and C. Rong, "Evaluating feature selection methods for short-term load forecasting," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2019, pp. 1–8.

[31] N. Ayub, N. Javaid, S. Mujeeb, M. Zahid, W. Z. Khan, and M. U. Khattak, "Electricity load forecasting in smart grids using support vector machine," in *Advanced Information Networking and Applications*, L. Barolli, M. Takizawa, F. Xhafa, and T. Enokido, Eds. Springer, 2020.

[32] J. Durbin and S. J. Koopman, *Time Series Analysis by State Space Methods*. Oxford, U.K.: Oxford Univ. Press, 2012.

[33] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231201007020

[34] H. Nie, G. Liu, X. Liu, and Y. Wang, "Hybrid of ARIMA and SVMs for short-term load forecasting," *Energy Procedia*, vol. 16, pp. 1455–1460, Jan. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876610212002391

[35] M. Kuhn and K. Johnson, *Applied Predictive Modeling*, vol. 26. Springer, 2013.

[36] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[37] W.-C. Hong, "Electric load forecasting by support vector model," *Appl. Math. Model.*, vol. 33, no. 5, pp. 2444–2454, May 2009.

[38] Z. Tan, J. Zhang, Y. He, Y. Zhang, G. Xiong, and Y. Liu, "Short-term load forecasting based on integration of SVR and stacking," *IEEE Access*, vol. 8, pp. 227719–227728, 2020.

[39] M. Abuella and B. Chowdhury, "Solar power forecasting using support vector regression," 2017, *arXiv:1703.09851*. [Online]. Available: https://arxiv.org/abs/1703.09851

[40] O. Kramer and F. Gieseke, "Short-term wind energy forecasting using support vector regression," in *Proc. Soft Comput. Models Ind. Environ. Appl., 6th Int. Conf. (SOCO)*, E. Corchado, V. Snášel, J. Sedano, A. E. Hassanien, J. L. Calvo, and D. Ślęzak, Eds. Berlin, Germany: Springer, 2011, pp. 271–280.

[41] Y. Zhang, H. Sun, and Y. Guo, "Wind power prediction based on PSO-SVR and grey combination model," *IEEE Access*, vol. 7, pp. 136254–136267, 2019.

[42] D. K. Barrow and S. F. Crone, "A comparison of AdaBoost algorithms for time series forecast combination," *Int. J. Forecasting*, vol. 32, no. 4, pp. 1103–1119, Oct. 2016.

[43] J. H. Friedman, "Stochastic gradient boosting," *Comput. Statist. Data Anal.*, vol. 38, no. 4, pp. 367–378, Feb. 2002.

[44] J. Li, J.-H. Cheng, J.-Y. Shi, and F. Huang, "Brief introduction of back propagation (BP) neural network algorithm and its improvement," in *Advances in Computer Science and Information Engineering*, D. Jin and S. Lin, Eds. Berlin, Germany: Springer, 2012, pp. 553–558.

[45] A. S. Ahmad, M. Y. Hassan, M. P. Abdullah, H. A. Rahman, F. Hussin, H. Abdullah, and R. Saidur, "A review on applications of ANN and SVM for building electrical energy consumption forecasting," *Renew. Sustain. Energy Rev.*, vol. 33, pp. 102–109, May 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032114000914

[46] U. K. Das, K. S. Tey, M. Seyedmahmoudian, S. Mekhilef, M. Y. I. Idris, W. Van Deventer, B. Horan, and A. Stojcevski, "Forecasting of photovoltaic power generation and model optimization: A review," *Renew. Sustain. Energy Rev.*, vol. 81, pp. 912–928, Jan. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032114000914

[47] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: https://www.mitpressjournals.org/doi/abs/10.1162/neco.1997.9.8.1735

[48] A. Géron, *Hands-On Machine Learning With Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[49] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*. [Online]. Available: https://arxiv.org/abs/1409.1259

[50] M. Sajjad, Z. A. Khan, A. Ullah, T. Hussain, W. Ullah, M. Y. Lee, and S. W. Baik, "A novel CNN-GRU-based hybrid approach for short-term residential load forecasting," *IEEE Access*, vol. 8, pp. 143759–143768, 2020.

[51] Y. Wang, W. Liao, and Y. Chang, "Gated recurrent unit network-based short-term photovoltaic forecasting," *Energies*, vol. 11, no. 8, p. 2163, Aug. 2018, doi: 10.3390/en11082163.

[52] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 10, pp. 1533–1545, Oct. 2014.

[53] M. Hussain, J. J. Bird, and D. R. Faria, "A study on cnn transfer learning for image classification," in *Advances in Computational Intelligence Systems*, A. Lotfi, H. Bouchachia, A. Gegov, C. Langensiepen, and M. McGinnity, Eds. Springer, 2019, pp. 191–202.

[54] I. Koprinska, D. Wu, and Z. Wang, "Convolutional neural networks for energy time series forecasting," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[55] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 3104–3112.

[56] J. S. Armstrong, "Combining forecasts," in *Principles of Forecasting*. Springer, 2001. pp. 417–439.

[57] S. Krstanovic and H. Paulheim, "Ensembles of recurrent neural networks for robust time series forecasting," in *Proc. Int. Conf. Innov. Techn. Appl. Artif. Intell.* Springer, 2017, pp. 34–46.

[58] Ausgrid. *The Largest Distributor of Electricity on Australias East Coast*. Accessed: Sep. 2020. [Online]. Available: https://www.ausgrid.com.au/Industry/Our-Research/Data-to-share/Solar-home-electricity-data

[59] Australian Government Bureau of Meteorology. *Bureau of Meterology*. Accessed: Sep. 2020. [Online]. Available: https://www.bom.gov.au/

[60] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 1–8.

[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

AIDA MEHDIPOUR PIRBAZARI (Graduate Student Member, IEEE) received the M.S. degree in computer science from the University of Stavanger (UiS), Stavanger, Norway, in 2017, where she is currently pursuing the Ph.D. degree in computer science. In 2017, she was a Researcher at the European Triangulum project, UiS. From 2018 to 2020, she was a Teaching Assistant in data science courses. Her Ph.D. thesis is on predictive analytics for maintaining power stability in smart communities. Her research and interests are in the field of smart grids, data mining, information retrieval, predictive analytics, and deep AI-based machine learning models.

EKANKI SHARMA (Student Member, IEEE) received the master's degree in electronics, electrotechnics, automation and signal processing (EEATS) from the University of Grenoble Alpes, Grenoble, France, with a specialization in the domain of systems, control & IT (MiSCIT). She is currently pursuing the Ph.D. degree in information and communications engineering focusing on energy informatics with the University of Klagenfurt, Austria. Since 2017, she has been working as a Research and Teaching Assistant in the Smart Grids Group with Prof. Wilfried Elmenreich. The research activities focus on the integration of renewable energy resources in the power grid and developing efficient ways to deliver variable renewable energy to the grid. Primarily, it includes building data-driven prediction models and uncertainty quantification for renewable energy systems and design, modeling, and analysis of future energy applications.

ANTORWEEP CHAKRAVORTY (Member, IEEE) received the Ph.D. degree with a thesis on privacy-preserving big data analytics at the University of Stavanger, Norway, in 2015. He is currently an Associate Professor with the University of Stavanger. His current research and development work is in the field of applied blockchains, big data, large scale machine learning, and data privacy. He has an interest in real-world problems, especially the development of privacy enabled data-driven services in smart energy, healthcare, and smart city domains.

WILFRIED ELMENREICH (Senior Member, IEEE) studied computer science at the Vienna University of Technology and received the venia docendi for technical computer science, in 2008. In 2007, he moved to the Alpen-Adria-Universität Klagenfurt as a Senior Researcher. After a visiting professorship at the University of Passau Elmenreich in 2013, he followed the call to the University of Klagenfurt. He is currently a Professor of Smart Grids with the Institute of Networked and Embedded Systems, Alpen-Adria-Universität Klagenfurt. He is a member of the Senate at the Alpen-Adria-Universität Klagenfurt, the Counselor of the IEEE Student Branch, and is involved in the master program on game studies and engineering. He is the author of several books and has published over 200 articles in the field of networked and embedded systems. He researches intelligent energy systems, self-organizing systems, and technical applications of swarm intelligence.

CHUNMING RONG (Senior Member, IEEE) is currently the Professor and Head of the Center for IP-based Service Innovation (CIPSI), University of Stavanger (UiS), Norway. He is also the Co-Chair of the IEEE Blockchain, the Chair of IEEE Cloud Computing. He is the adjunct Chief Scientist leading Big-Data Initiative at IRIS. He was the Vice President (2015–2016) of the CSA Norway Chapter. His research work focuses on data science, cloud computing, security, and privacy. He is honoured as a member of the Norwegian Academy of Technological Sciences (NTVA) since 2011. He is the Founder and Steering Chair of the IEEE CloudCom Conference and Workshop Series. He is the Steering Chair and Associate Editor of the IEEE TRANSACTIONS ON CLOUD COMPUTING (TCC), and the Co-Editor-in-Chief of the *Journal of Cloud Computing* (ISSN: 2192-113X) by Springer. He has extensive experience in managing large-scale Research and Development projects funded by both industry and funding agencies, both in Norway and the EU.