Computing and Informatics, Vol. 42, 2023, 126–156, doi: 10.31577/cai_2023_1_126

POWER CONSUMPTION FORECASTING BY HYBRID DEEP ARCHITECTURES WITH DATA FUSION

Serkan Özen, Volkan Atalay

Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
e-mail: {serkan.ozen, vatalay}@metu.edu.tr

Abstract. Many of the deep learning solutions for time-series forecasting reported in the literature include complex neural networks that may not be directly employed by the practitioner in the field. In this study, we demonstrate how the standard deep neural network types, convolutional neural network (CNN) and long short-term memory (LSTM) network can be applied in the field of time-series forecasting. This study consists of two parts. The first part is to compare CNN and LSTM models with classical methods like Random Forest (RF) and ARIMA for the univariate electric power consumption task. The second part is to use the best performing model from the first part in the hybrid model and perform data fusion with the newly built hybrid model for the electric power consumption forecasting task. CNN and LSTM models outperform traditional methods when their performances are evaluated on the univariate electric power consumption data of Illinois, USA. We also illustrate the use of hybrid deep learning models composed of standard CNN and LSTM for data fusion with the aim of time-series forecasting. When the hybrid models are applied to the fused data of the electric power consumption data and the multivariate weather data of Illinois, USA, the forecasting performance is improved compared to that when only univariate data is used.

Keywords: Time-series forecasting, data fusion, deep learning, hybrid models

1 INTRODUCTION

Time series is used essentially in any domain of applied science and engineering that comprises temporal measurements. Time-series forecasting is to forecast future values based on previously observed values by using a model. Time-series forecasting has recently become more popular because of the radical rise in the amount of available temporal data across various domains such as weather, climate, finance, health, and agriculture. Linear models such as autoregressive, moving average and their combinations are frequently used in univariate time series forecasting. However, most of the problems defined in terms of time series involve several variables. Furthermore, these problems originate from complex systems that can be better analyzed by non-linear models. The recent success of applying deep learning models in the fields such as natural language understanding, speech recognition, image analysis, and bio-informatics [1, 2, 3, 4, 5, 6] has motivated researchers in the field of time-series forecasting to make use of these non-linear models [7, 8, 9, 10]. Hence, it has been demonstrated that it is possible to learn temporal information and dynamics through deep neural networks in a data-driven manner. However, many of the deep learning solutions reported in the literature include complex deep learning models which may not form a convenient alternative for a practitioner in the field.

Data fusion is the process of integrating multiple data sources and the aim is to generate more effective and accurate forecasting than that provided by any individual data source. The literature on data fusion is vast and rich [11]. However, the use of deep learning for data fusion has not been exploited sufficiently. An important aspect of deep learning is that hybrid models can be easily constructed by combining various types of networks. This aspect coincides very well with the concept of data fusion in the sense that a single deep neural network (DNN) model can be used to process each data source and even an additional deep neural network model can be selected for the fusion phase which would overall lead naturally to a hybrid model. Alternatively, in a hybrid deep neural network model, an initial network can be used to extract features and a successive network may perform the regression task. In this way, featurization or feature engineering that requires domain knowledge and expertise are alleviated and commissioned to the neural network. Of course, there is a cost: an appropriate neural network model has to be selected along with its hyper-parameters and the overall system should be fine-tuned as well.

In this study, we aim to demonstrate the use of standard and established neural network types such as convolutional neural networks (CNN) and long short-term memory (LSTM) networks for time-series forecasting and the use of hybrid deep learning models for data fusion. CNN and LSTM networks are already available in the popular open-source programming frameworks.

We first conduct an experimental study to compare the performance of some traditional forecasting models, machine learning forecasting techniques and the two deep neural networks, CNN and LSTM networks. For demonstration purposes, univariate electric power consumption data and multivariate weather data from Chicago, Illinois, USA between 2005 and 2011 are employed as data sources. We then illustrate the use of hybrid deep learning models for data fusion with the aim of time-series forecasting and we present the results and compare the performance of various models.

2 RELATED WORK

In this section, we review time-series forecasting studies that used deep neural networks (DNN) and that were related particularly to energy consumption. In recent years, DNNs have been extensively applied to time series forecasting. We start with the studies using a single DNN with a univariate data type and then proceed with the studies having multivariate data type, employing hybrid neural network models and furthermore fusing different types of data.

Individual Household Electric Power Consumption (IHEC) dataset contains measurements of electric power consumption in one household in Paris, France with a one-minute sampling rate over a period of almost four years and it is a very popular multivariate dataset from UCI Machine Learning Repository [12]. Various types of deep neural networks have been applied to the IHEC dataset and in many of these studies, the results have been compared with other deep neural network types and traditional machine learning algorithms as well [13, 14, 15, 16]. In general, results showed that deep neural networks obtained better performance. Particularly, Marino et al. used two long short-term memory (LSTM) network-based models in which the first network had the traditional LSTM network model while the second one was LSTM network-based Sequence to Sequence model (Seq2Seq) [15]. Seq2Seq consisted of two consecutive LSTM networks where the first one is for encoding the input and the other is for decoding. Although both Seq2Seq and the traditional LSTM network performed well on hourly resolution datasets, Seq2Seq has outperformed the traditional LSTM network on one-minute resolution data. Amarasinghe et al. compared convolutional neural network (CNN) with other DNN types on the same IHEC dataset [14]. The challenge was to forecast the electric power consumption of the household for the next 60 hours in which CNN outperformed the multi-layer perceptron (MLP) neural network and support vector machine (SVM), but fell well below of Seq2Seq model. On the IHEC dataset, Kim and Cho have shown that the hybrid CNN + LSTM network model outperformed models composed of single CNN, single LSTM network and single model-based traditional regressors [16]. Building Energy Consumption (BEC) dataset is another energy-related dataset on which various deep neural networks have been shown to outperform the traditional machine learning methods and shallow neural networks [17, 18]. It was also possible to improve the CNN to obtain better performance [19].

Several studies using hybrid models constructed with multiple DNNs reported having better performance than when a single DNN is employed. This seems to be logical since components of a hybrid model can handle different tasks of learning. Usually, the first part of a hybrid model performs the feature extraction task and the second part handles the prediction task. Similarly, Yan et al. used a hybrid CNN + LSTM network model on UK Domestic Appliance-Level Electricity (UK-DALE) dataset [20] for forecasting household electricity consumption and this hybrid CNN + LSTM network model had better performance than that of traditional Autoregressive Integrated Moving Average (ARIMA) model, single LSTM network, and single CNN as well [21]. Ospina et al. applied a hybrid model composed of an LSTM network and deep MLP [22] on photo-voltaic power generation data (PVGD) [23]. In this study, wavelet transform was applied to the input signal, then fed to LSTM networks and LSTM network outputs were subsequently given to deep MLP with an extra temperature variable. The final prediction was observed as the output of deep MLP. The hybrid LSTM network-deep MLP model outperformed single support vector regression (SVR), single shallow MLP and single LSTM models too.

Data fusion is popularly used for time-series forecasting. For example, Kong et al. have shown that when appliance-based electricity consumption data (AMPds) [24] were fused by the LSTM network, it had better performance than that when only household level electricity data was used for forecasting home electric power consumption [25]. In a similar manner, the LSTM network was applied to forecast household heat gain (HHG) in buildings by fusing self-collected data from multiple resources such as electric power consumption of office devices, lighting, occupant count and wi-fi counts [26]. Results showed that the LSTM network with data fusion had lower error values than those of the baseline American Society of Heating and Air-Conditioning Engineers model.

A data fusion architecture was proposed [27] to forecast Spanish Electricity Market [28] price. The first stage of this architecture is composed of 3 models which were MLP neural network, adaptive neuro-fuzzy inference system and autoregressive moving average. The output of each model was the electricity price value and these were fed to an ordered weighted average model to get the final forecasted value. Results showed that the fusion architecture outperformed each one of the single models.

As a popular data fusion approach, traffic and weather data were fused in a study to forecast traffic density [29]. Forecasting was performed by considering only traffic flow data and also by using fusing traffic flow and weather data. Data used in this study were collected from stations in San Francisco, Bay area. It was shown that the fusion model outperformed the single data model. In addition, the deep belief network outperformed MLP neural network and ARIMA models for traffic density forecast in this study.

In a recent study carried out by Kong et al. [30], a multivariate ensemble method based on a dynamic transfer model is proposed for air pollution prediction. This model consists of two parts. The first part is an autoregressive dynamic transfer model and the second part is an ensemble of this model. The proposed model showed a good performance. Sanhudo et al. [31] have used the k-medoids clustering algorithm to rectify missing or erroneous values. After this process, ANN and SVM are used for temperature prediction. ANN has outperformed the SVM in prediction accuracy. These studies form good examples of the usage of hybrid models for the task of time-series forecasting.

3 METHODS

Time-series forecasting problem can be defined as predicting future values of historical time-series data. Let $X = X_1, X_2, \ldots, X_t$. A model is trained on this historical data in order to predict $X_{t+1}, X_{t+2}, \ldots, X_{t+k}$ where k is called prediction bound. Let prediction vector be $Y = Y_{t+1}, Y_{t+2}, \ldots, Y_{t+k}$. Then, the task of the model is to minimize the error $E = \sum_{i=t+1}^{t+k} |Y_i - X_i|$. Here, if $X \in \mathbb{R}^j$ where $j = 1$, the data is called univariate, otherwise it is called multivariate. For this task, we have designed hybrid deep models together with a data fusion approach. We addressed the problem of electric power consumption forecasting under two configurations.

Configuration 1. In the first configuration, only the previous electric power consumption data is given as input. Therefore, the input is univariate data. Remark that the output is also the electric power consumption data. Within this configuration, we used a prediction architecture composed of a single model. Single models that we have used are the ARIMA method, Random Forest, CNN and LSTM network. ARIMA and Random Forest are selected as traditional prediction methods and they are considered baseline methods. CNN and LSTM networks are chosen as single deep neural network models for this configuration.

Configuration 2. In the second configuration, we have fused weather information with the electric power consumption data as the input. In this way, we had multivariate data at the input and the output is always the electric power consumption data. In this configuration, we have employed hybrid deep neural network models composed of combinations of CNN and LSTM network models as the prediction method.

3.1 Baseline Methods for Univariate Input Data

Autoregressive Integrated Moving Average (ARIMA) model is a generalization of an autoregressive moving average model. ARIMA is expected to model the linear relationship between the variables of a data [32]. It is the combined form of Autoregressive, Integrated and Moving Average models and it has three hyperparameters which are p, d and q. p is used to model autoregressive part $AR(p)$ where p is the degree of autoregression; that is the number of past time steps to be included to forecast the current value of the series. An integrated model, $I(d)$, is used to remove seasonal trends where d is the number of times that the seasonal differencing will be applied to the time series data.

The last component of ARIMA is the Moving Average model, $MA(q)$. In the Moving Average model, the current value of the series is predicted using past error values of the series. As a result, these three models are combined to form $ARIMA(p, d, q)$ and as the ARIMA model is trained on data, it learns the model parameters. There are several problems with ARIMA when it is used for time-series forecasting tasks. First of all, ARIMA needs some pre-processing work on data. Data has to be free of seasonality and trend. Another drawback is that ARIMA is good at revealing the linear relationship between past lags and current instances of data. It is not good at revealing complex and non-linear relationships hidden inside the dataset. Lastly, ARIMA works on univariate time-series data. When there is a case of multivariate time-series forecasting, ARIMA can not extract the relationship between these multiple series.

Random forest (RF) can be used for time-series forecasting tasks. RF is a set of decision trees where the mission of each decision tree is to learn a different part of the training dataset. At each node of the tree, there is a comparison of a variable with a random value or a comparison of a set of variables with a set of random values in the case of the multivariate dataset. Hence, it can be said that each decision tree is responsible for learning different relationships between variables of the dataset. Lastly, the average prediction of decision trees becomes the final prediction result of RF. Like ARIMA, RF has also problems in revealing the complex relationships between input variables.

All these problems with the classical approaches made the deep neural networks popular for the time-series forecasting task. These networks are able to extract non-linear complex relationships hidden in both univariate and multivariate datasets by optimizing various hyperparameters.

3.2 Single Deep Neural Network Models

Long Short-Term Memory network (LSTM) is a type of recurrent neural network used in deep learning. An LSTM network has memory blocks that are connected through layers instead of neurons. Each memory block acts like a mini-state machine and takes the long-term dependency in the data into consideration using memory gates and the weights of the memory gates are learned during the training procedure.

Convolutional Neural Network (CNN) is a class of deep neural networks in which the hidden layers include layers that perform convolutions. The hidden layers learn to optimize the convolution filters through training for extracting features from input data. A CNN often includes pooling layers in addition to the convolutional layers. Pooling layers decrease the dimensions of data by combining the outputs of a group of nodes in one layer into a single node in the subsequent layer.

3.3 Hybrid DNN Models for Multivariate Input Data

Hybrid models are formed from various combinations of CNN and LSTM networks. Some examples are as follows:

- CNN + LSTM,
- LSTM + LSTM (which is also called the Seq2Seq model),
- CNN + LSTM + CNN,

- LSTM + CNN + LSTM.

The logic behind the hybrid models is the separation of feature extraction and learning the relation between these features. Feature extraction is handled by the first part of the hybrid models. The second part is responsible for the use of these features. In the case of the first two models, the first part of the models, CNN and LSTM, are responsible for feature extraction. In both models, the second LSTM learns how to use these features to make future predictions. The logic behind using CNN and LSTM as feature extractors is that both are good at learning long-term sequential relations in time-series datasets. This way, we want to compare which is more effective when used as a hybrid with the LSTM model.

Furthermore, we want to analyze the effect of an extra model in the hybrid model. We created third and fourth models to test the effect of the extra CNN and LSTM layers respectively. We wanted to question whether adding an extra model would improve the learning efficiency of the hybrid model.

The model for data fusion is presented in Figure 1. Univariate electric power consumption data is processed by Submodel-1 and the forecast horizon is (that is, predictions are obtained for the next) 120 hours. Electric power consumption and weather data are combined and given as input to Submodel-2. The forecast horizon is still 120 hours. The predictions from Submodel-1 and Submodel-2 are then fed into Submodel-3 which generated the final value. Submodel-1 is selected among the univariate models that give the best performance as the result of a series of evaluations as described in Section 5.5. Submodel-2 is selected as one of the CNN+LSTM or LSTM+LSTM or CNN+LSTM+CNN or LSTM+CNN+LSTM, according to their performance given in Section 5.5. Submodel-3 is an MLP neural network that has outputs of Submodel-1 and Submodel-2 as its input. Submodel-3 uses these inputs to decide the final electric power consumption prediction. As a result, it is important to note that there is a single model that consists of 3 submodels. Submodel-1 runs on univariate electric power consumption data. It accepts the past 1 200 hours of electric power consumption data and predicts the next 120 hours of electric power consumption. Submodel-2 takes 1 200 hours of multi-variate data (electric power consumption and eight weather variables) and produces the prediction for the next 120 hours of electric power consumption. Submodel-3 takes these two predictions coming from the outputs of Submodel-1 and Submodel-2 and generates a prediction by means of linear regression. This is the final prediction of electric power consumption. Hence, there is a single model consisting of 3 submodels.

We wanted to compare the performances of the single models and hybrid models and evaluate the effect of data fusion on forecasting performance. We have also investigated the effect of the level of hybrid models on forecasting performance. CNN + LSTM and LSTM + LSTM are common models that are widely used in time-series forecasting [33, 34, 35, 36, 37, 38]. We have additionally tested the performance of CNN+LSTM+CNN and LSTM+CNN+LSTM. The basic question is whether a third model improves forecasting performance or not. The intuition

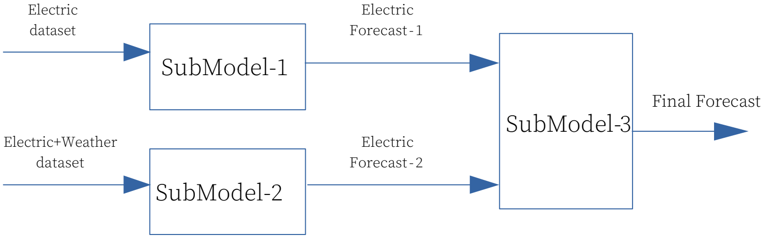


Figure 1. Data fusion architecture used in the models.

behind the three-level hybrid models comes from the question of whether adding an additional model at the start or at the end of the two-level hybrid model increases the forecasting performance or not. The additional model incorporated into the start might help the model to better extract the features. Similarly, the model incorporated at the end of the two-level model might increase the performance of sequence learning.

4 DATA

We used weather data and electric power consumption data from Chicago, Illinois and Pittsburgh, Pennsylvania. Electric power consumption data is extracted from [39] while weather data is taken from [40]. Data is in hourly resolution starting from 2011/1/1 until 2016/12/31 which makes up 52 607 data points in total. All data points are normalized into the interval $[0, 1]$ by min-max normalization. B.6: In order to make the dataset size multiple of 5, because of the 5-fold partition, the last two data points are discarded. Missing data points are just discarded. Electric power consumption data is univariate, meaning that it consists of only one electric power consumption variable in terms of Megawatts as the unit. It contains a region-based aggregation of electric power consumption data from Chicago. Weather data contains a population-weighted average of weather variables and these variables are precipitation, temperature, irradiance surface, irradiance top of atmosphere, snowfall, snow depth, cloud cover, and air density. For data fusion, the electric power consumption variable is incorporated into the weather dataset, making it 9 variables in total. Our motivation to fuse weather and electric power consumption datasets is because their correlation level seems to be good for fusion. Effects of weather variables may have results on electric power consumption, i.e. during hot or cold days, it would be meaningful to see a rise in electric power consumption. Moreover, increasing the number of input variables gives better prediction results as indicated in Section 2. The information regarding the datasets used in the study is given in Table 1. The correlations between the variables of the weather data and the electric power consumption are also investigated and Figure 2 shows the correlation map of the variables of weather data and the variable of electric power consumption data. Correlation analysis is performed using the Spearman method to reveal the monotonic relationship between variables. We avoided using Pearson since it only reveals linear relationships between variables. It is observed that eight variables of weather data and the single variable of electric power consumption data are neither highly correlated nor not correlated. The correlation level is just enough to provide new information. The correlation values in column (or row) nine are between -0.1 and 0.5. This means that weather and electric power consumption variables are neither uncorrelated nor highly correlated. In the former case, it would be meaningless to use uncorrelated data and in the latter case, it would be a redundant variable since they are highly correlated. The datasets can be accessed at `https://github.com/serkanozen/FusionElectricityForecast/`.

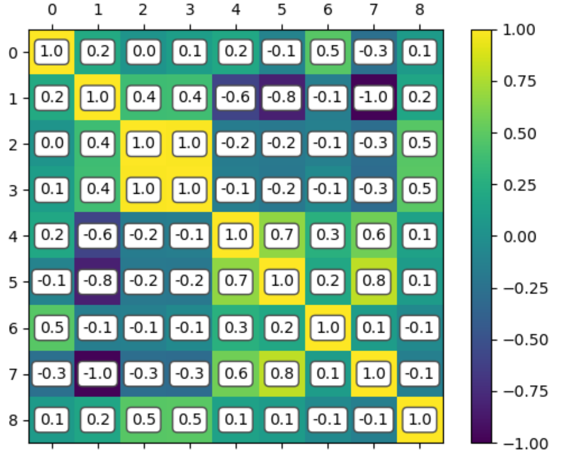


Figure 2. Correlation map of weather data variables and electric power consumption. 0: Precipitation, 1: Temperature, 2: Irradiance surface, 3: Irradiance top of atmosphere, 4: Snowfall, 5: Snow depth, 6: Cloud cover, 7: Air density, 8: Electric power consumption.

Name	# of Variables	Type
Chicago-electric power consumption	1	univariate
Chicago-weather	8	multivariate

Table 1. Properties of datasets used in this study. All datasets cover hourly data from 2011 to 2016 and their sizes are all the same: 52 607 data items.

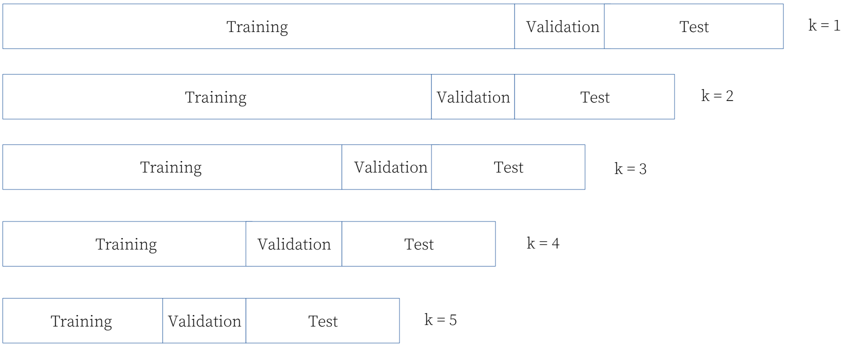


Figure 3. Nested cross-validation method. Training and validation size reduces as the k-fold value increases whereas test size remains the same in all folds. Size of each partition can be seen in Table 2.

5 EXPERIMENTAL SETUP AND RESULTS

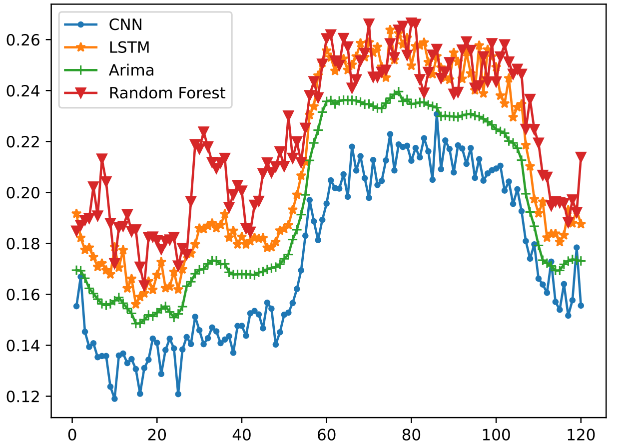
In this section, experiments carried out with hybrid models using a data fusion approach are presented and the results are discussed in detail. Additionally, the hyperparameter tuning used to find optimal parameter configurations for models is discussed.

5.1 Nested Cross Validation

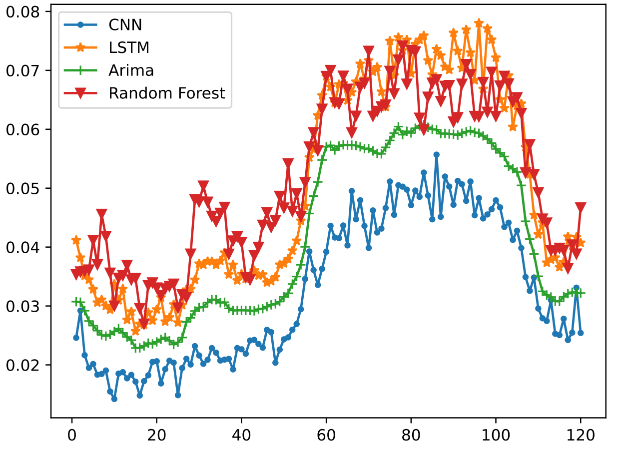
The methodology in k-fold nested cross-validation is that time series data is divided into partitions without a random process since the order of the data of the time series is important. Time series data can not be randomly chosen as in the traditional k-fold cross-validation since it would modify the time-series nature of the data, making it unordered. As it can be seen in Table 2, n^k is the number of data in k^{th} fold of nested cross-validation and $n^k = \frac{S}{5}(5 - k + 1)$ where S is the total dataset size. At every fold of the nested cross-validation, the size of the test set is fixed, $n^k_{test} = n_{inp} + n_{out} + N = 1\,370$ for $k = 1, 2, 3, 4, 5$ where $N = 50$. Then, $\frac{1}{4}$ of the remaining data points is used for validation and $\frac{3}{4}$ is used for training; $n^k_{valid} = \frac{1}{4}(n^k - n^k_{test})$ and $n^k_{train} = \frac{3}{4}(n^k - n^k_{test})$. To demonstrate an example, the above explanations yield dataset partitions that are shown in Figure 3. The sizes of training set n^k_{train}, validation set n^k_{valid} and test set n^k_{test} across different k values (folds) are given in Table 2.

5.2 Error Metrics

The prediction error is evaluated in terms of the three most popular metrics in time series forecasting: Mean Squared Error (MSE), Root Mean Squared Error (RMSE),



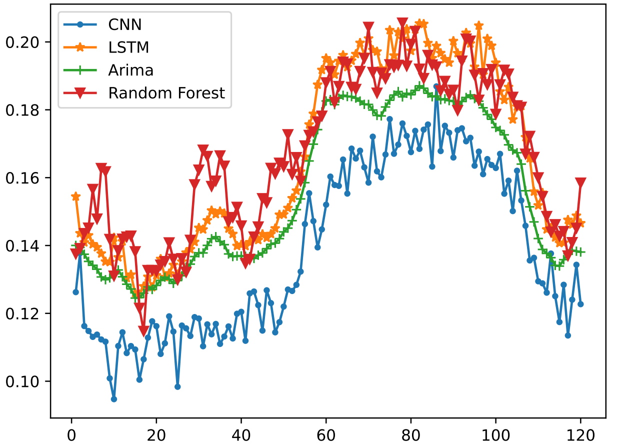
a) Root mean squared error (RMSE)



b) Mean squared error (MSE)

k	n^k	n^k_{train}	n^k_{valid}	n^k_{test}
1	52 605	38 426	12 808	1 370
2	42 084	30 535	10 178	1 370
3	31 563	22 645	7 548	1 370
4	21 042	14 754	4 918	1 370
5	10 521	6 863	2 287	1 370

Table 2. Size (in terms of hour) of the dataset n^k, training set n^k_{train}, validation set n^k_{valid} and test set n^k_{test} across various k values (folds)



c) Mean absolute error (MAE)

Figure 4. Submodel-1 forecasting errors Chicago, Illinois electric power consumption dataset. The horizontal axis represents the forecasted hour, the vertical axis represents the error value.

and Mean Absolute Error (MAE).

$$\mathrm{MSE} = \frac{1}{n}\Sigma_{i=1}^{n}(p_i - a_i)^2, \tag{1}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(p_i - a_i)^2}, \tag{2}$$

$$\mathrm{MAE} = \left(\frac{1}{n}\right)\sum_{i=1}^{n}\left|p_i - a_i\right|, \tag{3}$$

where p_i is the predicted value and a_i is the actual value of data item i.

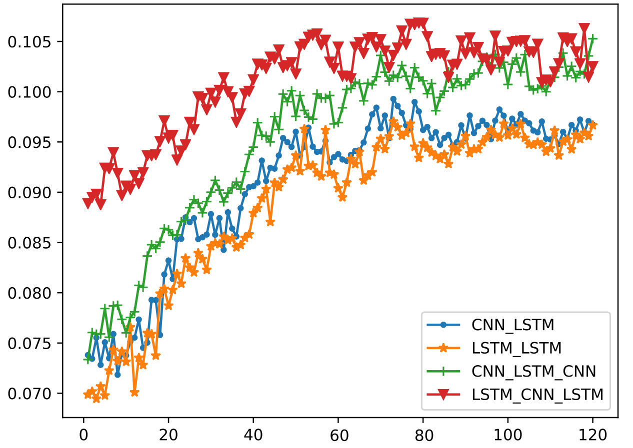
Experiments are performed with a 5-fold nested cross-validation scheme explained in the previous subsection and the average MSE, RMSE and MAE values are calculated on the test dataset. These metrics are selected since they emphasize different aspects of the error. MSE penalizes large errors since it takes the square of the difference between the predicted value and the actual value. RMSE is the square root of the MSE value. MAE on the other hand directly measures the difference between the predicted value and actual value.

5.3 Selection of Hyperparameter Values

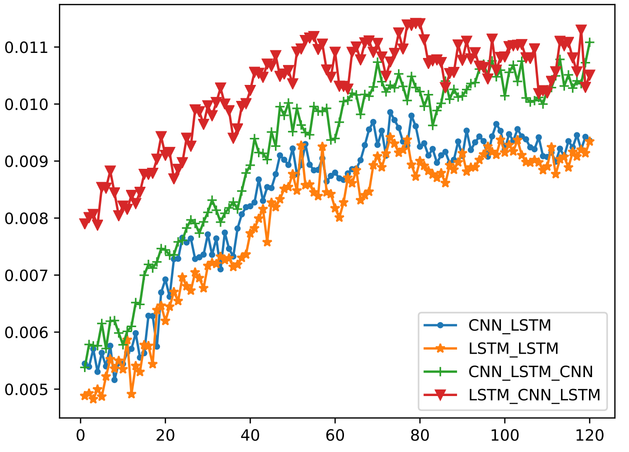
The hyperparameter values of ARIMA, RF, CNN and LSTM network models have to be selected. Since the number of hyperparameter value combinations is very large for CNN and LSTM networks, we limited our search for best parameter values to

Model Name **Model Name**	**Parameter Name**	**Values**	**RMSE Value**
CNN	Layer size	$\{1, \mathbf{3}, 5, 10\}$	$\{0.23, \mathbf{0.15}, 0.19, 0.18\}$
	# of Filters	$\{12, 5, 3\}$, $\{12, 2, 1\}$, $\{\mathbf{8}, \mathbf{5}, \mathbf{3}\}$, $\{5, 5, 3\}$	$\{0.19, 0.16, \mathbf{0.14}, 0.15\}$
	Kernel size	$\{10, 6, 3\}$, $\{8, 6, 3\}$, $\{3, 3, 2\}$, $\{\mathbf{1}, \mathbf{1}, \mathbf{2}\}$	$\{0.29, 0.26, 0.18, \mathbf{0.13}\}$
	Pool size	$\{\mathbf{4}, \mathbf{2}\}$, $\{4, 4\}$, $\{2, 2\}$	$\{\mathbf{0.14}, 0.15, 0.18\}$
	Batch size	$\{32, 256, \mathbf{1\,024}\}$	$\{0.24, 0.19, \mathbf{0.16}\}$
	# of Epochs	$\{5, 10, \mathbf{15}\}$	$\{0.21, 0.16, \mathbf{0.15}\}$
	Input time steps	$\{240, \mathbf{1\,200}, 2\,400\}$	$\{0.25, \mathbf{0.11}, 0.13\}$
LSTM	Layer size	$\{\mathbf{1}, 2, 4, 8\}$	$\{\mathbf{0.22}, 0.24, 0.23, NA\}$
	# of Units	$\{1, \mathbf{3}, 8\}$	$\{0.25, \mathbf{0.22}, 0.36\}$
	Batch size	$\{32, 256, \mathbf{1\,024}\}$	$\{0.43, 0.45, \mathbf{0.26}\}$
	# of Epochs	$\{5, \mathbf{10}, 15\}$	$\{0.39, \mathbf{0.24}, 0.26\}$
	Input time steps	$\{240, \mathbf{1\,200}, 2\,400\}$	$\{0.28, \mathbf{0.21}, 0.29\}$
ARIMA	p,d,q	$\{8, 4, 4\}$, $\{4, 4, 4\}$, $\{4, 4, 1\}$, $\{\mathbf{4}, \mathbf{2}, \mathbf{1}\}$, $\{2, 0, 1\}$	$\{0.25, 0.31, 0.17, \mathbf{0.16}, 0.2\}$
Random Forest	Number of estimators	$\{10, \mathbf{25}, 50, 100\}$	$\{0.25, \mathbf{0.13}, 0.14, 0.35\}$

Table 3. Grid of hyperparameters for deep learning and classical models. Hyperparameter values that are indicated in bold are the ones used to obtain the results in this study.

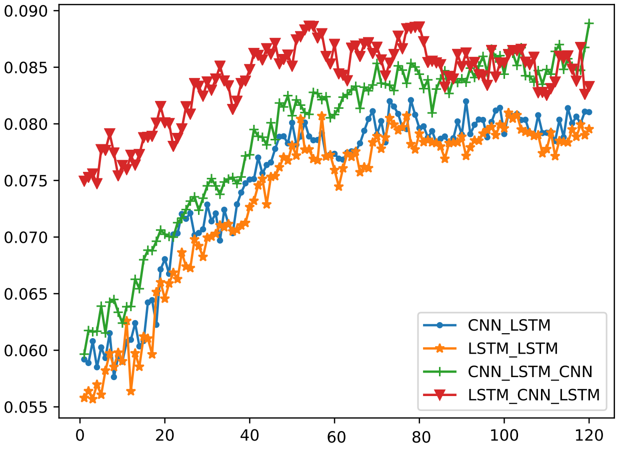


a) RMSE errors vs hours of forecasting



b) MSE errors vs hours of forecasting

some predetermined set of values and we applied a forward-search mechanism by using RMSE value as the criterion to be minimized. The predetermined sets for the hyperparameter value search are shown in Table 3 along with the corresponding average RMSE values. The first elements of sets were taken as the initial parameter values and for each hyperparameter, alternative values in the corresponding set were used with this initial configuration. For example, in the case of CNN, the hyperparameter values of the initial experiment were 1 layer, 12 filters, 10 kernels, pool size of 4, batch size of 32 and 5 epochs. With this configuration, layer size was changed to 3, 5 and 10 by keeping the values of all other hyperparameters fixed and average RMSE values are recorded. The lowest error was obtained when the layer size was 3 and therefore, the other experiments were conducted with 3 three

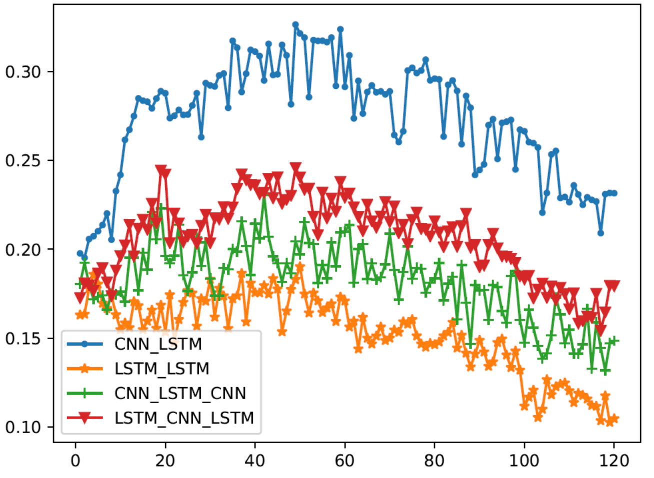


c) MAE errors vs hours of forecasting

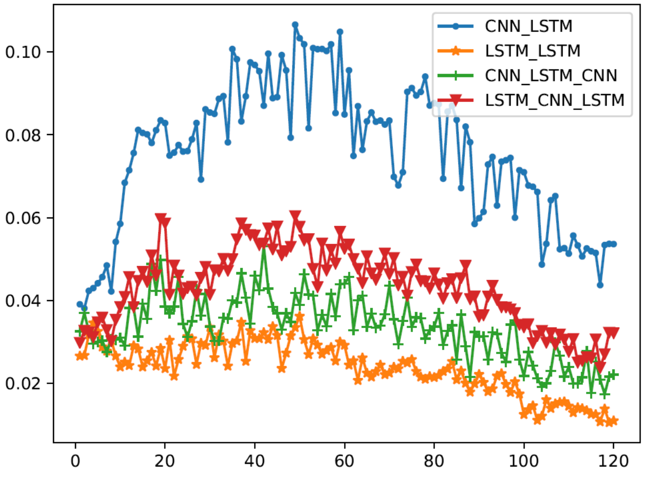
Figure 5. Fusion model forecasting errors for Chicago, Illinois datasets. Here the experiment is realized by setting Submodel-2 to one of CNN + LSTM, LSTM + LSTM, CNN + LSTM + CNN and LSTM + CNN + LSTM, Submodel-1 is set to CNN since it has the lowest error for univariate forecasting task and Submodel-3 is set to its default value MLP. The horizontal axis represents the forecasted hour. The vertical axis represents the error value.

layers for CNN. After having decided for CNN to have 3 layers, configurations with a varying number of filters for each layer were run and the configuration with the lowest error value (that is $\{8,5,3\}$) was selected. The number of kernels is another parameter that is associated with the convolutional layer. After trying $\{10,6,3\}$, $\{8,6,3\}$, $\{3,3,2\}$, and $\{1,1,2\}$ for a set of 3 parameters, each for a layer, the lowest error was obtained with $\{3,3,2\}$. Two Max-Pooling layers were added after the first and second CNN layers. Layer size of $\{4,2\}$ had the lowest error value. Batch size experiments were performed with three different values that were $\{32,256,1\,024\}$. The best result was obtained with batch size being 1 024. For the epoch size, the lowest error was obtained by 10 epochs. The input size of 1 200-time steps gave the lowest RMSE value among the set of values $\{240,1\,200,2\,400\}$.

Model-specific hyperparameters of the CNN model obtained as a result of these experiments can be seen in Table 4. During the experiments with CNN, we observed that using more than three convolutional layers made the training process longer, though it did not provide important accuracy gain. Therefore, the number of convolutional layers is fixed to 3. The number of "MaxPooling" layers is directly related to the number of convolutional layers. The numbers of "flatten" and "fully connected" layers are both 1 since the first one flattens output to 1 dimension and the latter one gives the final output vector. Keeping "Kernel Size" and "Pool Size" values low at convolutional layers gave better results than the higher values do.

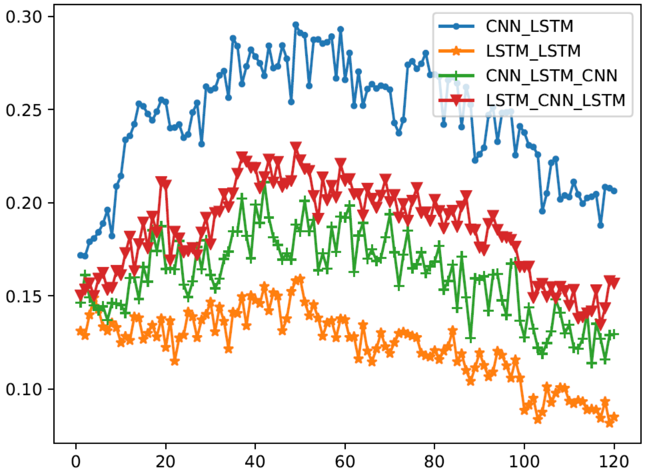


a) RMSE errors vs hours of forecasting



b) MSE errors vs hours of forecasting

A similar methodology was used for LSTM hyperparameter tuning. The initial setup consisted of one layer, one unit, a batch size of 32, and 5 epochs. After having tried various layer sizes of $\{1, 2, 4, 8\}$, the lowest error was obtained by 1 layer. Among the alternatives for the number of units, the lowest error was obtained by 3 units. For the LSTM network model, batch size and the number of epochs showed similar characteristics to those of CNN and the lowest error was obtained by the batch size of 1 024 and 10 epochs. For input time steps, the LSTM model gave the lowest error when 1 200-time steps were fed into the model. Model specific

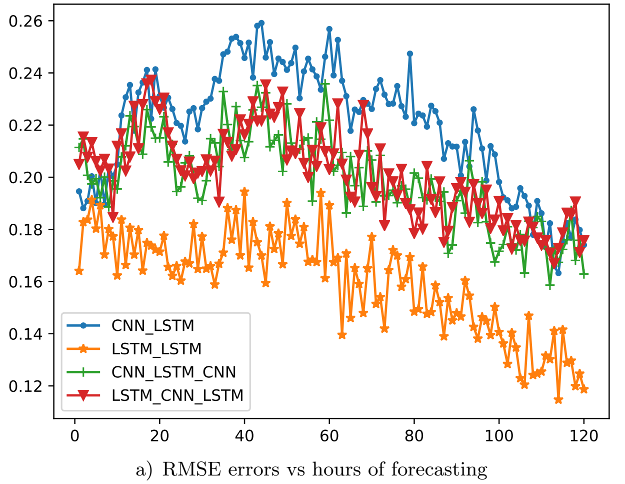


c) MAE errors vs hours of forecasting

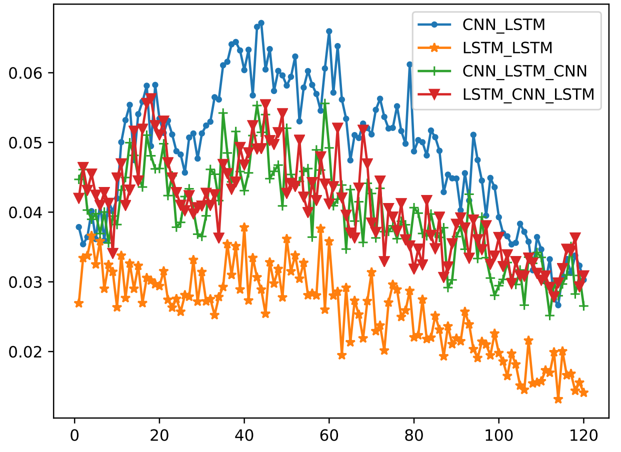
Figure 6. Submodel-2 forecasting errors for Chicago, Illinois datasets. The horizontal axis represents the forecasted hour. The vertical axis represents the error value.

hyperparameters of LSTM network model are given in Table 5. For the LSTM network, during the experiments, we observed that a single LSTM layer with 3 hidden units has a good performance. Increasing or decreasing the size of hidden units or layers did not improve the performance of the model.

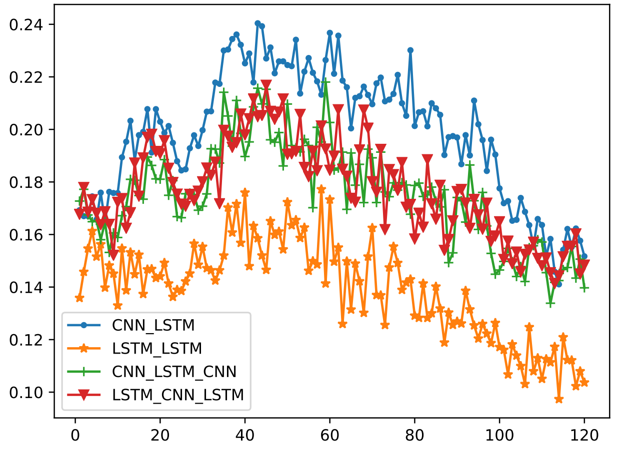
The values for ARIMA hyperparameters (p, d, q) were selected by grid search and $\{2.0, 0.0, 1.0\}$ gave an acceptable RMSE value in addition to removing some season-



a) RMSE errors vs hours of forecasting

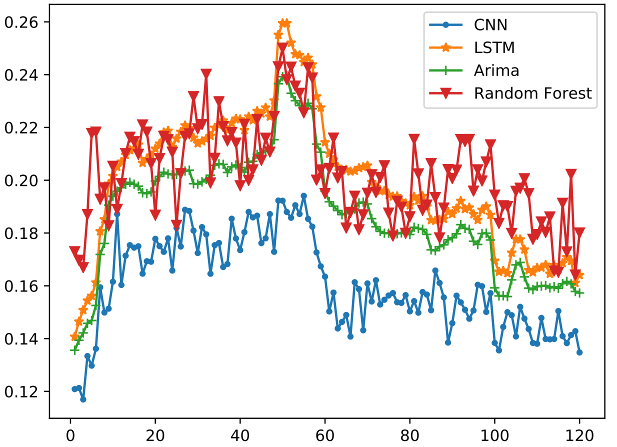


b) MSE errors vs hours of forecasting



c) MAE errors vs hours of forecasting

Figure 7. Fusion model forecasting errors for Pittsburgh, Pennsylvania datasets. Here the experiment is realized by setting Submodel-2 to one of CNN + LSTM, LSTM + LSTM, CNN + LSTM + CNN and LSTM + CNN + LSTM, Submodel-1 is set to CNN since it has the lowest error for univariate forecasting task and Submodel-3 is set to its default value MLP. The horizontal axis represents the forecasted hour. The vertical axis represents the error value.



a) Root mean squared error (RMSE)

ality and trend patterns. For the Random Forest model, the number of estimators was selected to be 25 by grid search.

Layer	**Parameters**					
	# Nodes	# Channels	# Filters	Kernel Size	Act. Func.	Pool Size
Convolution	1 200	1 or 8	8	3	Relu	–
MaxPooling	1 198	1 or 8	–	–	–	2
Convolution	599	1 or 8	5	3	Relu	–
MaxPooling	597	1 or 8	–	–	–	2
Convolution	298	1 or 8	3	2	Relu	–
Flatten	297	–	–	–	Flatten	–
Fully connected	120	–	–	–	Fully connected	–

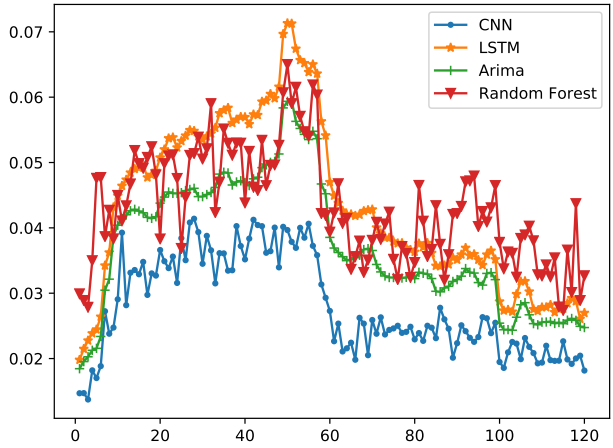
Table 4. Model specific hyper-parameters of the CNN model

Layer	**Parameters**	
	# Units	Act. Func.
LSTM	3	Relu
Fully connected	120	Linear

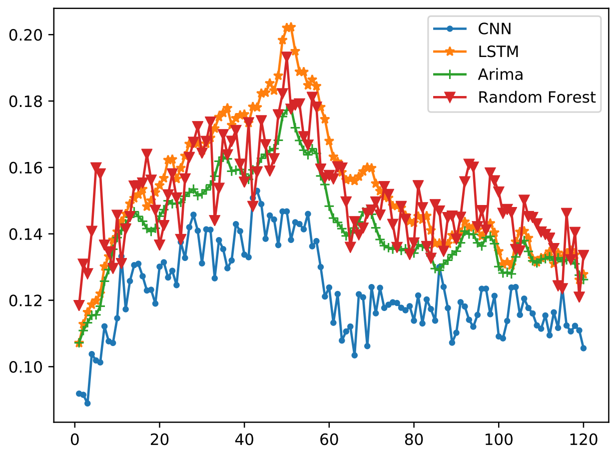
Table 5. Model specific hyper-parameters of the LSTM model

5.4 Evaluation on Electric Power Consumption Data

ARIMA, RF, CNN and LSTM network models were trained on univariate electric power consumption data for a forecast horizon of 120 hours. That is, the model would predict the electric power consumption for the subsequent 120 hours (or



b) Mean squared error (MSE)



c) Mean absolute error (MAE)

Figure 8. Submodel-1 forecasting errors running on IHEC univariate electric power consumption dataset. The horizontal axis represents the forecasted hour, the vertical axis represents the error value.

for the 5 days). The reason for choosing the prediction bound as 5 days is because it is a common number of days that the weather forecasting services use and it is an empirically determined value to achieve good test performance. Error-values in the models are obtained as follows. Size of test dataset is fixed and equal to $n_{test}^{k} = n_{inp} + n_{out} + N = 1\,370$ where $N = 50$, meaning that model performs 50 different run sessions in order to predict next 120 hours. Each of these sessions is run by a different input set. Input and output sets are formed by shifting these sets right by one hour. That is, if first input and output sets are $Group1 : X_1, X_2, \ldots, X_{1\,200}, Y_1, Y_2, \ldots, Y_{120}$ then second ones become $Group2 : X_2, X_3, \ldots, X_{1\,201}, Y_2, Y_3, \ldots, Y_{121}$. This way, the 50$^{\text{th}}$ input and output sets become $Group50 : X_{50}, X_{51}, \ldots, X_{1\,249}, Y_{50}, Y_{51}, \ldots, Y_{169}$. MSE, RMSE and MAE values are calculated for each predicted hour over related hours of all groups. For example, Average error for hour-1 is calculated by running MSE, RMSE and MAE on the set $Group1\_Y_1$, $Group2\_Y_2$, $Group3\_Y_3, \ldots, Group50\_Y_{50}$ and average error for hour-50 is calculated on the set $Group1\_Y_{50}, Group2\_Y_{51}, Group3\_Y_{52}, \ldots, Group50\_Y_{99}$. This way, the error values of all predicted hours are plotted on the error figures.

5.5 Evaluation on Fused Electric Power Consumption and Weather Data

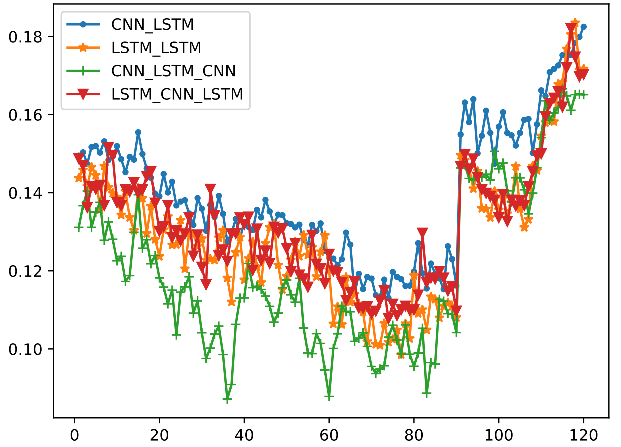
We evaluated and investigated hybrid deep neural network models in terms of the fused data. Observing that CNN has the best performance on the univariate electrical consumption dataset, Model 1 is set to be a CNN. Model 2 is selected to be one of the following models: CNN + LSTM, LSTM + LSTM, CNN + LSTM + CNN and LSTM + CNN + LSTM models.

CNN + LSTM and LSTM + LSTM are common models that are widely used in time-series forecasting. We have additionally tested the performance of CNN + LSTM + CNN and LSTM + CNN + LSTM. The basic question is whether a third model improves forecasting performance or not. The intuition behind the 3-level hybrid models comes from the question of whether adding an additional model at the start or at the end of the 2-level hybrid model increases the forecasting performance or not. The additional model put to the start might help the model to better extract the features hidden in the dataset. Similarly, a third model inserted at the end of the 2-level model might increase the performance of sequence learning.

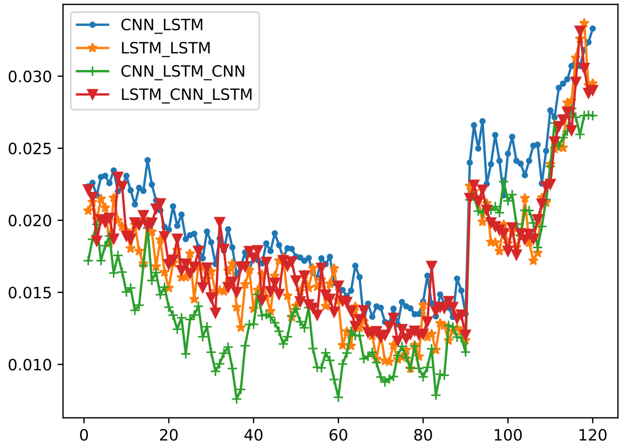
We have also questioned the effect of the level of hybrid models on forecasting performance. However, we could not find any enhancement in errors when we increase the hybrid level.

MSE, MAE and RMSE values of single models (Submodel-1), run on univariate electric power consumption data from Chicago, Illinois can be seen in Figure 4. Likewise, Figure 5 shows the MSE, MAE and RMSE values of hybrid models (Submodel-3), run on combined electric power consumption and weather datasets of Chicago, Illinois. In order to clarify the advantage of data fusion and hybrid models, MSE, MAE and RMSE values of single models, run on univariate electric power consumption data (Figure 4) and hybrid models, run on combined electric power consumption and weather datasets (Figure 5) of Chicago, Illinois can be compared. LSTM + LSTM hybrid model has the lowest RMSE value among other hybrid models, that are, CNN + LSTM, CNN + LSTM + CNN and LSTM + CNN + LSTM. Among the hybrid models with data fusion, the configuration, that is LSTM+LSTM running on multivariate and CNN on the univariate dataset, has the lowest error range that lies in the interval of 0.07 and 0.95. Similar observations are made for MSE and MAE. The results suggest that the hybrid model for fusion outperforms the single CNN model's electric power consumption forecasts. Moreover, in order to question the effect of the proposed Fusion model, we compare the error values of Submodel-1 and Submodel-2 with the proposed model. As it can be seen in Figure 5 and Figure 6 the Fusion model has lower error values since it utilizes predictions coming from both Submodel-1 and Submodel-2. As a result, it has lower error values than the other two models. The proposed fusion model, consisting of all 3 sub-models, is run on Pittsburgh, Pennsylvania dataset and similar results are obtained; that is, a hybrid model with the configuration LSTM + LSTM running on multivariate and CNN running on univariate gave the lowest error values among other models as it can be seen in Figure 7. The reason for the decrease in error values over predicted days is that the model can learn future time steps well enough since the input size is high enough. Feeding the network with 1 200 time steps and predicting the next 120 hours is not a hard problem for the model. Further time steps could even be predicted better than the former ones as the number of the predicted time steps is much lower than the number of the input time steps. IHEC is one of the popular univariate electricity consumption datasets as mentioned in Section 2. We have run our univariate models on the IHEC dataset and observed their performances based on various error metrics. Results of Submodel-1 for univariate lies between 0.14 and 0.18 (see Figure 8). CNN has the lowest error values across various runs. Fusion model is tested by using IHEC and weather dataset of Paris, obtained from [40]. Both datasets intersect between the years 2007 and 2010. As it can be seen in Figure 9, results suggest that Fusion model predicts better in most of the predicted hours. The RMSE error is below 0.14 until the 80^{th} predicted hour and then it lies between 0.14 and 0.18. On average, it performs better than Submodel-1. Moreover, this time Fusion model performs best when its Submodel-2 is set to LSTM + CNN + LSTM. That is IHEC dataset is more suitable for the hybrid model to extract meaningful information. Similar to the experiments held by various studies (see Section 2), CNN has good performance among other baseline methods such as ANN, ARIMA and SVM, since the univariate electric power consumption dataset does not include long-term relationships to be revealed by LSTM. Its short-term relationships are enough to be captured by CNN.

In order to demonstrate the efficiency of our fusion model, we compared it with kCNN-LSTM [41]. kCNN-LSTM was used for forecasting the energy consumption of a building in IIT-Bombay. The dataset was composed of electric power consumption values with 15 minutes intervals. The dataset was divided into three clusters each of which contains similar consumption patterns and kCNN-LSTM was executed separately on each cluster. Since weather data is necessary for our fusion model we have obtained the weather data for the specific time interval [40]. We were then able to compare the evaluation results of kCNN-LSTM with our CNN (univariate), LSTM + LSTM (multivariate) and Fusion (multivariate) models. The evaluation results of kCNN-LSTM are presented in Table 6. CNN is our univariate model that runs only on univariate electric power consumption data and it generated worse results than kCNN-LSTM. Our LSTM + LSTM model was not also better than kCNN-LSTM. On the other hand, our Fusion model generated better results than



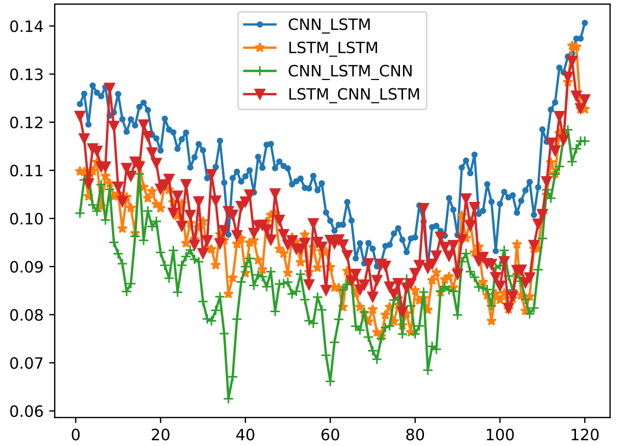
a) Root mean squared error (RMSE)



b) Mean squared error (MSE)

all other models.

In order to demonstrate the effect of the fusion model, we compared its results with the results of Submodel-1 and Submodel-2 where each sub-model is set to various prediction models. As it can be seen in Table 7, the fusion model gives better results than individual prediction models set for Submodel-1 and Submodel-2. Submodel-1 is set to one of the following models: Random Forest, ARIMA and CNN. Remark that Submodel-1 runs only on a univariate electric power consumption dataset. Submodel-2 is set to CNN + LSTM and LSTM + LSTM which are the best performing among other multivariate models. The models set for Submodel-2 run on electric power consumption and Weather datasets. Our fusion model is composed of Submodel-1 which is set to CNN and Submodel-2 which is set to LSTM + LSTM.



c) Mean absolute error (MAE)

Figure 9. Fusion model forecasting errors running on IHEC and Paris weather datasets. The horizontal axis represents the forecasted hour, the vertical axis represents the error value.

Type	Metrics	Models			
		kCNN-LSTM	CNN	LSTM + LSTM	Fusion
Cluster-1	MSE	0.0095	0.0398	0.0239	0.0079
	RMSE	0.0974	0.1726	0.1178	0.0889
	MAE	0.0711	0.1653	0.0778	0.0701
Cluster-2	MSE	0.0212	0.0428	0.0339	0.0104
	RMSE	0.1456	0.2068	0.1841	0.1019
	MAE	0.0997	0.1123	0.0861	0.0911
Cluster-3	MSE	0.0010	0.0248	0.0013	0.0007
	RMSE	0.0303	0.1574	0.0361	0.0264
	MAE	0.0165	0.0732	0.0256	0.0231

Table 6. A.1: Comparison of univariate CNN model, multivariate LSTM + LSTM and our fusion model with kCNN-LSTM [41]

As Table 7 demonstrates, the results of our Fusion model are better than those generated by Submodel-1 and Submodel-2.

Diebold-Mariano test is popularly used for testing the statistical significance of forecasting results [42], [43]. Diebold-Mariano test is run on the error results of models running on univariate electric power consumption dataset and separately on the error results of models running on multivariate (electric power consumption + weather) datasets of Chicago, Illinois. Diebold-Mariano test result is demonstrated by a heat map. The green cell means that the p-value is closer to zero which means that the result of the model on the X-axis is statistically more significant than the models on the Y-axis.

Datasets	Metrics	Submodel-1			Submodel-2		Fusion
		Random F.	Arima	CNN	CNN + LSTM	LSTM + LSTM	
Chicago	MSE	0.0553	0.0477	0.0325	0.0224	0.0252	0.0076
	RMSE	0.2351	0.2184	0.1802	0.1496	0.1587	0.0871
	MAE	0.1735	0.1566	0.1436	0.1189	0.1218	0.0742
Pittsburgh	MSE	0.0419	0.0384	0.0418	0.0578	0.0345	0.0245
	RMSE	0.2046	0.1960	0.2062	0.2404	0.1857	0.1565
	MAE	0.1681	0.1512	0.1542	0.1983	0.1548	0.1358

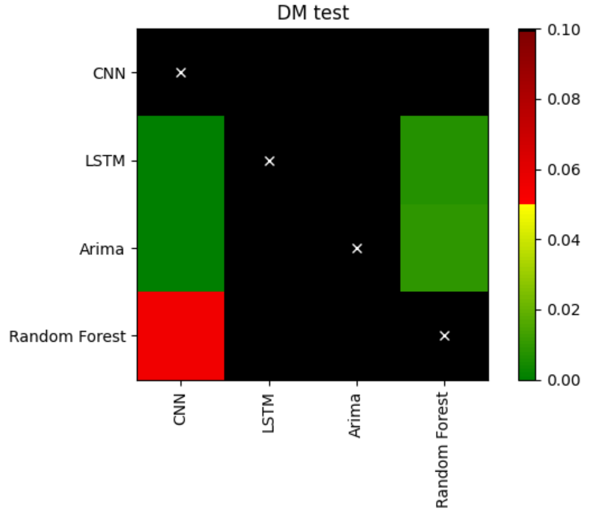
Table 7. Comparison of Submodel-1 and Submodel-2 results with the result of Fusion model across different datasets

As it can be seen in Figure 10, heat maps suggest that for the green cells, the model on the X-axis produces statistically more significant values than the models on the Y-axis. Specifically, in Figure 10 a), the CNN model performs statistically better than other models. For the multivariate (electric power consumption + weather), as it can be seen in Figure 10 b), the LSTM + LSTM model performs statistically better than other models. Figure 10 c) represents the result of Diebold-Mariano test of models. In these tests, the results of 3 different executions are compared. In the first execution, Submodel-1 is set to CNN and it is run on the univariate electricity consumption dataset. In the second execution, Submodel-2 is set to LSTM + LSTM and is run on the multivariate weather + electricity consumption dataset. Lastly, the Fusion model, where Submodel-1 is set to CNN, Submodel-2 is set to LSTM+LSTM and Submodel-3 is set to MLP, is run on the weather + electricity consumption dataset. As can be seen in the Figure, the Fusion model produces more significant results than any of the Submodels.

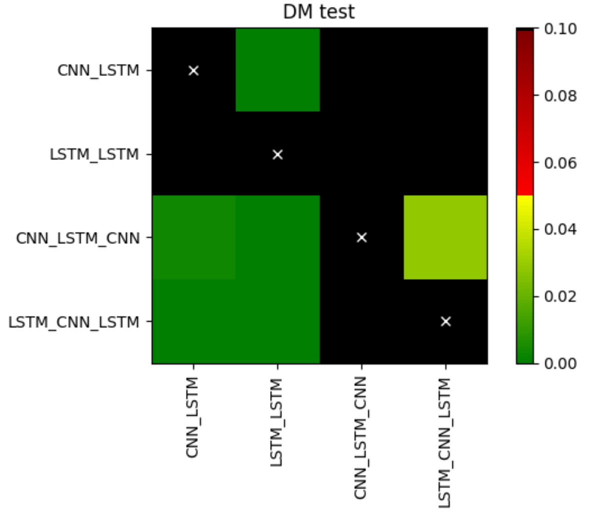
We have managed to reduce error amounts with this data fusion and hybrid model approach without requiring too much additional computational effort. There is not much difference in the total training time of a hybrid model than training the components of the hybrid model, since the outputs of the Submodel-1 and Submodel-2 are fed to an MLP which is composed of a single hidden layer of 10 nodes. In addition, as a result of the recent advances in hardware technologies and access to GPUs, the computational effort was not a problem for this study.

6 CONCLUSION

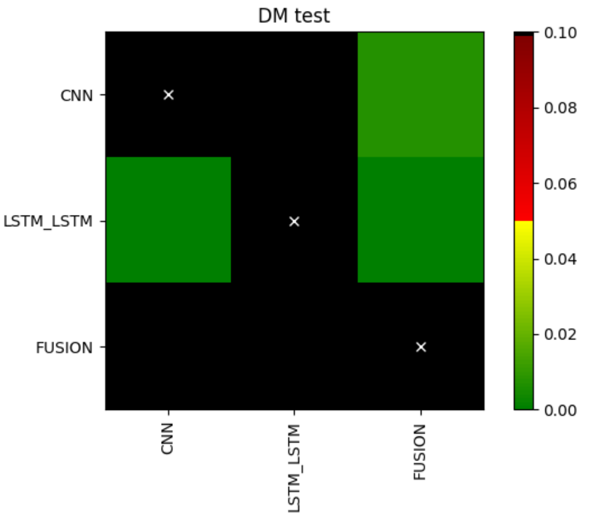
In this study, we have shown that data fusion and hybrid models can improve electric power consumption prediction tasks. Firstly, we have shown that CNN performs best for the electric power consumption prediction tasks. Then, we have shown that the data fusion model with a hybrid model consisting of CNN as Submodel-1 and LSTM + LSTM as Submodel-2, performs best among other hybrid alternatives. To the best of our knowledge, this study is the first in utilizing a mix of single and hybrid models in fusion architecture for the electric power consumption prediction task. We have shown that hybrid models together with data fusion architecture



a) Diebold-Mariano test results of models running on univariate electric power consumption dataset of Chicago, Illinois. CNN produces statistically more significant values than other models.



b) Diebold-Mariano test results of Submodel-2 which is set to one of the hybrid models seen on the horizontal axis, running on multivariate (electric power consumption + weather) dataset of Chicago, Illinois. LSTM + LSTM produces statistically more significant values than other models.



c) Diebold-Mariano test results of Submodel-1 set to CNN, Submodel-2 set to LSTM + LSTM and Fusion Model that consists of Submodel-1, Submodel-2 and Submodel-3 which is set to MLP.

Figure 10. Heat map of p-values of Diebold-Mariano test. Closer to the values to 0, Models on X-axis have more significant results than those on Y-axis.

work better than the non-hybrid models used with the univariate dataset. This study shows that when there is an available weather dataset for a region, it can be used to enhance the electric consumption prediction. As a future study, we consider applying transfer learning of these models on a target dataset with insufficient data size. We aim to show that these models learn the relation in the features of the domain dataset and can be used to predict future electric power consumption on the target dataset.

REFERENCES

[1] Rifaioglu, A. S.—Atas, H.—Martin, M. J.—Cetin-Atalay, R.—Atalay, V.—Doğan, T.: Recent Applications of Deep Learning and Machine Intelligence on in silico Drug Discovery: Methods, Tools and Databases. Briefings in Bioinformatics, Vol. 20, 2019, No. 5, pp. 1878–1912, doi: 10.1093/bib/bby061.

[2] LeCun, Y.—Bengio, Y.—Hinton, G.: Deep Learning. Nature, Vol. 521, 2015, No. 7553, pp. 436–444, doi: 10.1038/nature14539.

[3] Sureyya Rifaioglu, A.—Doğan, T.—Jesus Martin, M.—Cetin-Atalay, R.—Atalay, V.: DEEPred: Automated Protein Function Prediction with Multi-Task Feed-Forward Deep Neural Networks. Scientific Reports, Vol. 9, 2019, No. 1, pp. 1–16, doi: 10.1038/s41598-019-43708-3.

[4] Cho, K.—van Merriënboer, B.—Gülçehre, Ç.—Bahdanau, D.—Bougares, F.—Schwenk, H.—Bengio, Y.: Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In: Moschitti, A., Pang, B., Daelemans, W. (Eds.): Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). ACL, 2014, pp. 1724–1734, doi: 10.3115/v1/D14-1179.

[5] Graves, A.—Mohamed, A. R.—Hinton, G.: Speech Recognition with Deep Recurrent Neural Networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 6645–6649, doi: 10.1109/ICASSP.2013.6638947.

[6] Redmon, J.—Divvala, S.—Girshick, R.—Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[7] Huang, Z.—Zhou, S.—Heng, W.: Learning to Paint with Model-Based Deep Reinforcement Learning. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 8708–8717, doi: 10.1109/ICCV.2019.00880.

[8] Benidis, K.—Rangapuram, S. S.—Flunkert, V.—Wang, Y.—Maddix, D. et al.: Deep Learning for Time Series Forecasting: Tutorial and Literature Survey. ACM Computing Surveys, Vol. 55, 2022, No. 6, Art. No. 121, doi: 10.1145/3533382.

[9] Tadayon, M.—Pottie, G.: Comparative Analysis of the Hidden Markov Model and LSTM: A Simulative Approach. 2020, doi: 10.48550/arXiv.2001.09547.

[10] Hewamalage, H.—Bergmeir, C.—Bandara, K.: Recurrent Neural Networks for Time Series Forecasting: Current Status and Future Directions. International Journal of Forecasting, Vol. 37, 2021, No. 1, pp. 388–427, doi: 10.1016/j.ijforecast.2020.06.008.

[11] Tealab, A.: Time Series Forecasting Using Artificial Neural Networks Methodologies: A Systematic Review. Future Computing and Informatics Journal, Vol. 3, 2018, No. 2, pp. 334–340, doi: 10.1016/j.fcij.2018.10.003.

[12] Dua, D.—Graff, C.: UCI Machine Learning Repository. 2017, `http://archive.ics.uci.edu/ml`.

[13] Mocanu, E.—Nguyen, P. H.—Gibescu, M.—Kling, W. L.: Deep Learning for Estimating Building Energy Consumption. Sustainable Energy, Grids and Networks, Vol. 6, 2016, pp. 91–99, doi: 10.1016/j.segan.2016.02.005.

[14] Amarasinghe, K.—Marino, D. L.—Manic, M.: Deep Neural Networks for Energy Load Forecasting. 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), 2017, pp. 1483–1488, doi: 10.1109/ISIE.2017.8001465.

[15] Marino, D. L.—Amarasinghe, K.—Manic, M.: Building Energy Load Forecasting Using Deep Neural Networks. IECON 2016 – 42nd Annual Conference of the IEEE Industrial Electronics Society, 2016, pp. 7046–7051, doi: 10.1109/IECON.2016.7793413.

[16] Kim, T. Y.—Cho, S. B.: Predicting Residential Energy Consumption Using CNN-LSTM Neural Networks. Energy, Vol. 182, 2019, pp. 72–81, doi: 10.1016/j.energy.2019.05.230.

[17] Li, C.—Ding, Z.—Zhao, D.—Yi, J.—Zhang, G.: Building Energy Consumption Prediction: An Extreme Deep Learning Approach. Energies, Vol. 10, 2017, No. 10,

Art. No. 1525, doi: 10.3390/en10101525.

[18] Li, C.—Ding, Z.—Yi, J.—Lv, Y.—Zhang, G.: Deep Belief Network Based Hybrid Model for Building Energy Consumption Prediction. Energies, Vol. 11, 2018, No. 1, Art. No. 242, doi: 10.3390/en11010242.

[19] Deng, Z.—Wang, B.—Xu, Y.—Xu, T.—Liu, C.—Zhu, Z.: Multi-Scale Convolutional Neural Network with Time-Cognition for Multi-Step Short-Term Load Forecasting. IEEE Access, Vol. 7, 2019, pp. 88058–88071, doi: 10.1109/ACCESS.2019.2926137.

[20] Kelly, J.—Knottenbelt, W.: The UK-DALE Dataset, Domestic Appliance-Level Electricity Demand and Whole-House Demand from Five UK Homes. Scientific Data, Vol. 2, 2015, No. 1, Art. No. 1500074, doi: 10.1038/sdata.2015.7.

[21] Yan, K.—Wang, X.—Du, Y.—Jin, N.—Huang, H.—Zhou, H.: Multi-Step Short-Term Power Consumption Forecasting with a Hybrid Deep Learning Strategy. Energies, Vol. 11, 2018, No. 11, Art. No. 3089, doi: 10.3390/en11113089.

[22] Ospina, J.—Newaz, A.—Faruque, M. O.: Forecasting of PV Plant Output Using Hybrid Wavelet-Based LSTM-DNN Structure Model. IET Renewable Power Generation, Vol. 13, 2019, No. 7, pp. 1087–1095, doi: 10.1049/iet-rpg.2018.5779.

[23] Solar Power Data for Integration Studies. `https://www.nrel.gov/grid/solar-power-data.html` (Accessed: 18-Apr-2023).

[24] Makonin, S.—Popowich, F.—Bartram, L.—Gill, B.—Bajić, I. V.: AMPds: A Public Dataset for Load Disaggregation and Eco-Feedback Research. 2013 IEEE Electrical Power and Energy Conference (EPEC), 2013, pp. 1–6, doi: 10.1109/EPEC.2013.6802949.

[25] Kong, W.—Dong, Z. Y.—Hill, D. J.—Luo, F.—Xu, Y.: Short-Term Residential Load Forecasting Based on Resident Behaviour Learning. IEEE Transactions on Power Systems, Vol. 33, 2018, No. 1, pp. 1087–1088, doi: 10.1109/TPWRS.2017.2688178.

[26] Wang, Z.—Hong, T.—Piette, M. A.: Data Fusion in Predicting Internal Heat Gains for Office Buildings Through a Deep Learning Approach. Applied Energy, Vol. 240, 2019, pp. 386–398, doi: 10.1016/j.apenergy.2019.02.066.

[27] Darudi, A.—Bashari, M.—Javidi, M. H.: Electricity Price Forecasting Using a New Data Fusion Algorithm. IET Generation, Transmission and Distribution, Vol. 9, 2015, No. 12, pp. 1382–1390, doi: 10.1049/iet-gtd.2014.0653.

[28] Starter: Spanish Electricity Market: 15371217-6. `https://www.kaggle.com/code/kerneler/starter-spanish-electricity-market-15371217-6` (Accessed 18-Apr-2023).

[29] Koesdwiady, A.—Soua, R.—Karray, F.: Improving Traffic Flow Prediction with Weather Information in Connected Cars: A Deep Learning Approach. IEEE Transactions on Vehicular Technology, Vol. 65, 2016, No. 12, pp. 9508–9517, doi: 10.1109/TVT.2016.2585575.

[30] Kong, T.—Choi, D.—Lee, G.—Lee, K.: Air Pollution Prediction Using an Ensemble of Dynamic Transfer Models for Multivariate Time Series. Sustainability, Vol. 13, 2021, No. 3, Art. No. 1367, doi: 10.3390/su13031367.

[31] Sanhudo, L.—Rodrigues, J.—Vasconcelos Filho, E.: Multivariate Time Series Clustering and Forecasting for Building Energy Analysis: Application to Weather Data Quality Control. Journal of Building Engineering, Vol. 35, 2021, Art. No. 101996, doi: 10.1016/j.jobe.2020.101996.

[32] Box, G. E. P.—Jenkins, G. M.—Reinsel, G. C.: Time Series Analysis: Forecasting and Control. 4th Edition. John Wiley & Sons, 2008, doi: 10.1002/9781118619193.

[33] Boulila, W.—Ghandorh, H.—Khan, M. A.—Ahmed, F.—Ahmad, J.: A Novel CNN-LSTM-Based Approach to Predict Urban Expansion. Ecological Informatics, Vol. 64, 2021, Art. No. 101325, doi: 10.1016/j.ecoinf.2021.101325.

[34] Barzegar, R.—Aalami, M. T.—Adamowski, J.: Short-Term Water Quality Variable Prediction Using a Hybrid CNN-LSTM Deep Learning Model. Stochastic Environmental Research and Risk Assessment, Vol. 34, 2020, No. 2, pp. 415–433, doi: 10.1007/s00477-020-01776-2.

[35] Li, Y.—He, Y.—Zhang, M.: Prediction of Chinese Energy Structure Based on Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM). Energy Science and Engineering, Vol. 8, 2020, No. 8, pp. 2680–2689, doi: 10.1002/ese3.698.

[36] Li, T.—Hua, M.—Wu, X.: A Hybrid CNN-LSTM Model for Forecasting Particulate Matter (PM2.5). IEEE Access, Vol. 8, 2020, pp. 26933–26940, doi: 10.1109/ACCESS.2020.2971348.

[37] Sagheer, A.—Kotb, M.: Unsupervised Pre-Training of a Deep LSTM-Based Stacked Autoencoder for Multivariate Time Series Forecasting Problems. Scientific Reports, Vol. 9, 2019, No. 1, Art. No. 19038, doi: 10.1038/s41598-019-55320-6.

[38] Cui, Y.—Yin, B.—Li, R.—Du, Z.—Ding, M.: Short-Time Series Load Forecasting by Seq2seq-LSTM Model. 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Vol. 9, 2020, pp. 517–521, doi: 10.1109/ITAIC49862.2020.9339110.

[39] Mulla, R.: Hourly Energy Consumption. `https://www.kaggle.com/robikscube/hourly-energy-consumption` (Accessed 18-Apr-2023).

[40] Renewables.ninja. `https://www.renewables.ninja/` (Accessed 18-Apr-2023).

[41] Somu, N.—Raman M. R., G.—Ramamritham, K.: A Deep Learning Framework for Building Energy Consumption Forecast. Renewable and Sustainable Energy Reviews, Vol. 137, 2021, Art. No. 110591, doi: 10.1016/j.rser.2020.110591.

[42] Lago, J.—Marcjasz, G.—De Schutter, B.—Weron, R.: Forecasting Day-Ahead Electricity Prices: A Review of State-of-the-Art Algorithms, Best Practices and an Open-Access Benchmark. Applied Energy, Vol. 293, 2021, Art. No. 116983, doi: 10.1016/j.apenergy.2021.116983.

[43] Diebold, F. X.—Mariano, R. S.: Comparing Predictive Accuracy. Journal of Business and Economic Statistics, Vol. 20, 2002, No. 1, pp. 134–144, doi: 10.1198/073500102753410444.

Serkan Özen received his B.Sc. and M.Sc. degrees in computer engineering from the Middle East Technical University, in 2011 and 2014. Currently, he is a Ph.D. candidate at the same department and the university. He has been working as a software engineer in various companies for more than 10 years. Now he works as an embedded software engineer for a company in the defense industry. His current research interests include machine learning, deep learning and time-series analysis.

Volkan Atalay obtained his B.Sc. and M.Sc. in electrical and electronics engineering from the Middle East Technical University (METU). He received his Ph.D. in computer science from the Universite Paris Descartes, Paris, France in 1993. He spent a year at the New Jersey Institute of Technology, USA, as a visiting scholar and another year at the Virginia Bioinformatics Institute, Virginia Tech, USA, as a visiting faculty member. He is Professor in the Department of Computer Engineering, METU, where he has been since 1993. His main research interests include bioinformatics, data stream analysis and machine learning applications.