Forecasting Electric Vehicle Supply Equipment Availability

Stijn J. Rotman and Boris Čule

Department of Cognitive Science and Artificial Intelligence, Tilburg University,
Warandelaan 2, 5037 AB Tilburg, The Netherlands
s.j.rotman@tilburguniversity.edu, b.cule@tilburguniversity.edu

Abstract. The advent of electric driving poses novel scientific challenges. One such challenge is predicting the availability of electric vehicle supply equipment (EVSEs). Previous work addressing this question made use of insufficient data sources, limiting the available methods. In this paper, we make use of a much larger amount of data, opening the door to methods previously unused in this domain. The data used is especially suited for prediction using deep learning models, because of the high number of similar units of EVSEs present in the data. Specific deep learning architectures specialised in learning temporal dependencies are compared in their ability to predict the minutes of availability in a given hour for an EVSE. Long short-term memory and gated recurrent unit-based models are combined with a temporal convolution layer and compared in their performance on unseen periods and units. Of these models, the convolutional long short-term memory architecture was found to perform the best with a root mean squared error of 1.2 minutes per hour. However, we conclude that both architectures are similar in their performance, as both models were able to generalise well to unseen periods and to unseen EVSEs in those periods.

Keywords: Electric Vehicle Supply Equipment · Time Series Forecasting · Deep Learning.

1 Introduction

While still relatively small, the market for electric cars is growing rapidly [19]. Qualitative studies performed in different countries found that a lack of charger availability is a concern for potential electric car owners [4, 11, 26]. A quantitative analysis confirmed this, suggesting that an increase in charger availability has a positive effect on the uptake of electric driving [35]. Therefore, accurate forecasts regarding EVSE availability are of importance for advancing electric vehicle adoption.

In this paper, our aim is to predict the hourly availability of different units of EVSEs as accurately as possible. In order to achieve this goal, two methods for predicting multiple related time series are compared: convolutional long short-term memory neural networks (CNN-LSTM) and convolutional gated recurrent unit neural networks (CNN-GRU). This comparison is summarised in the following research question:

How do convolutional long short-term memory and convolutional gated recurrent units perform when forecasting electric vehicle supply equipment?

Performance in this context is measured using the root mean squared error (RMSE). The dataset for this paper is provided by Eco-Movement, a company specialising in EVSE data. This dataset distinguishes itself from datasets in previous studies because it combines data from various sources, vastly increasing the size and accuracy of the data. The availability of charge points across Europe is measured as session data, provided by various charge point operators for the years 2020 and 2021. Data from 2020 is used for training and data from 2021 is used for testing. An added challenge is presented by the fact that some EVSEs present in the test set are not present in the training set at all.

As mentioned above, the main contribution of our paper lies in the evaluation of the suitability of deep learning methods on forecasting EVSE availability at such a large scale. Existing work in this field uses insufficient data, focuses on different aspects of EVSEs, such as energy consumption, or deals with specific subsets of data, such as household chargers.

Our experiments show that, generally speaking, the RMSE performance of both models is quite similar across most hyperparameter settings. While the models are able to generalise well to unseen periods as well as unseen EVSEs, models created with hyperparameters that make the model too complex are not able to generate accurate predictions.

The rest of this paper is organised as follows. Section 2 situates our work by providing an overview of existing literature. The CNN-LSTM and CNN-GRU models, as well as a baseline algorithm, are introduced and elaborated upon in Section 3. Subsequently, in Section 4, we lay out our experimental set-up and present and interpret our results, followed by a discussion. Finally, we conclude the paper in Section 5.

2 Related Work

In this section, we situate our work by reviewing the related existing literature. First, we describe the broader field of research regarding electric vehicle use. Then, we discuss the field of EVSE-related forecasting, indicating gaps in the existing literature. In addition, we introduce the methods chosen for use in this paper, weighing them against other possible methods.

With the rise of electric driving, interest in studying topics concerning plug-in vehicle charging has also grown. These topics include forecasting charging schemes [16, 21], charging infrastructure [14], and electric grid integration [9]. Further topics include predicting the availability of charging spots as well as their energy consumption.

Straka et al. focus on explaining energy consumption [29]. Their work focuses on EVSE locations in the Netherlands only, using data enriched with geospatial information. Their work is able to explain why energy consumption is at a certain

level for specific EVSEs very well, though the use of many features makes it less suitable for forecasting further ahead.

Other works include a study on classifying whether household chargers will be used the next day using (a combination of) various machine learning algorithms [1]. This focus is of specific interest for power grid operators, as it helps them predict the electrical load.

Buzna et al. focus on publicly available charging stations [5]. This paper uses a hierarchical approach to create probabilistic electrical vehicle load forecasts. While this methodology is shown to predict well and be applicable to new regions, it does have significant drawbacks in the context of our work. As discussed before, the use of many features as inputs is infeasible in this context. Additionally, only the methodology is shown to be applicable to new regions, extending the forecasts to new regions still requires training separate prediction models for these regions.

Another study focusing on publicly available EVSEs is performed by Bikcora et al. [2]. Their work makes use of generalised linear models to predict availability of two particularly busy EVSEs. The authors find that using lags in combination with time indicator variables can be sufficient to obtain accurate forecasts. However, their approach requires a different model for each connector, with performances highly dependent on the specific charge point. On a small scale, separate models for specific EVSEs might not pose a problem. However, training such a number of models on data as large as ours is infeasible.

From the above, it becomes apparent that our work is unique in its approach regarding focus and scale. While other papers with similar focus exist, the amount of data used is unprecedented. The much larger amount of data available for this study opens the door for methods that are more advanced than those used previously in existing work. At the same time, the scale of the data results in a requirement to use these methods, as fitting separate models for every EVSE has become infeasible. Therefore, the availability of EVSEs needs to be predicted using not only a methodology, but also a model that can generalise across related units.

Recent developments in processing multiple related time series stem from deep neural networks (DNN). Prominent examples of DNN architectures for time series forecasting are long short-term memory (LSTM) [17] and gated recurrent units (GRU) [7]. A widely used advancement of the LSTM architecture is the inclusion of convolutional layers (CNN-LSTM) [13, 23, 22, 30].

Though less common, the GRU architecture is occasionally extended with the inclusion of convolutional layers (CNN-GRU) as well [18, 32]. As the CNN-GRU architecture is used less frequently, there is also a smaller number of performed studies comparing the CNN-LSTM and CNN-GRU architectures. Research that has been conducted comparing these methodologies suggests that CNN-GRU typically performs better than CNN-LSTM [10, 28]. The more numerous comparisons of LSTM and GRU show more ambiguity in the performance differences of these methods. Some studies report that GRU improves prediction performance over LSTM [27, 33], while others find little difference [12, 24, 34].

Emerging advancements in the broader forecasting literature include using LSTM in an encoder-decoder based approach [20, 6]. This approach has the major drawback of only forecasting sequences of a fixed length. Conventional methods that predict one time step are more flexible in the amount of steps they can predict, as the predictions can themselves be used as inputs for further periods.

In summary, the expansion of electric driving poses several questions. One of these questions is how EVSE availability can be predicted effectively. Prior work related to this question is lacking in the data used, which results in an inability to produce models that generalise well to new periods or EVSEs. We aim to close this gap in literature by using a dataset large enough to train advanced methods that do allow such generalisation. The methods we use to predict EVSE availability are CNN-LSTM and CNN-GRU. The performances of these models are compared against each other and a simple, but strong, baseline. The following section provides a technical explanation of these models.

3 Methodology

In this section, we describe the data and methodology used in this paper. We start with a specification of the data source and the preprocessing steps taken for different algorithms. Then, we provide a technical explanation of the baseline model, after which the CNN-LSTM and CNN-GRU architectures are formally specified. Subsequently, we describe the software infrastructure used in our experimental analysis.

3.1 Data and Preprocessing

The dataset for this paper is provided by Eco-Movement, a company specialising in EVSE data. As mentioned in Section 1, this dataset combines data from various sources, and is larger and more accurate than data used in prior studies.

The availability of charge points across Europe is measured as session data, provided by various charge point operators for the years 2020 and 2021. Because of the temporal dependencies in the data, observations made in 2020 are assigned to the training set and observations made in 2021 are assigned to the test set. This results in different sets of EVSEs in the two sets due to EVSEs being added or removed over time. The resulting dataset contains 39 215 EVSEs in the training set and 54 479 EVSEs in the test set. A total of 33 556 EVSEs are present in both the training set and the test set. There are therefore 5 659 EVSEs present in the training set but not in the testing set and 20 923 EVSEs present in the testing set but not in the training set.

Changes in availability status are saved as events in the database. Table 1 shows 3 example observations of such events. In order to make the data suitable for time series analysis, the sessions are used to create time series indicating the available minutes per hour. The observations for each hour are then divided by 60 to scale all values to fall between 0 and 1.

Table 1. Example rows of raw data

Timestamp	Status
2019-12-31 16:46:07	charging
2020-01-01 06:41:02	available
2020-01-01 11:46:12	charging

After dummy variables indicating month of year, day of week and hour of day are added, the time series is transformed into a sliding window. The sliding window consists of four weeks of hourly availability data with the time dummies and the availability in the next hour, which serves as the outcome label. Table 2 shows the resulting data from preprocessing the example data from Table 1 using the above steps up to the transformation into windows.

Table 2. Example rows of preprocessed data. Omitted columns are marked with dots

Availability	Jan	Feb	...	Mon	Tue	Wed	...	12AM	01AM	02AM	03AM	04AM	...
0.000	1	0	...	0	0	1	...	1	0	0	0	0	...
0.000	1	0	...	0	0	1	...	0	1	0	0	0	...
0.000	1	0	...	0	0	1	...	0	0	1	0	0	...
0.000	1	0	...	0	0	1	...	0	0	0	1	0	...
0.000	1	0	...	0	0	1	...	0	0	0	0	1	...
0.000	1	0	...	0	0	1	...	0	0	0	0	0	...
0.684	1	0	...	0	0	1	...	0	0	0	0	0	...
1.000	1	0	...	0	0	1	...	0	0	0	0	0	...
1.000	1	0	...	0	0	1	...	0	0	0	0	0	...
1.000	1	0	...	0	0	1	...	0	0	0	0	0	...
1.000	1	0	...	0	0	1	...	0	0	0	0	0	...
0.770	1	0	...	0	0	1	...	0	0	0	0	0	...

3.2 Baseline

Typically, extended periods of either 0 or 60 minutes of availability alternate with single hours where the availability falls between 0 and 60. This means that we are able to predict occupancy fairly accurately based on just two prior hours. The baseline is laid out in Algorithm 1.

This algorithm is naive, because it is only able to predict a change in occupation status *after* receiving the signal that something changed. Nevertheless, due to the nature of the data, even such a naive baseline can produce very good predictions. Beating this baseline is required to show that a model has gone beyond such a naive approach and is able to *predict* the moment the status of an EVSE changes.

Algorithm 1: Baseline

Data: A_{t-1}, the availability at $t-1$, A_{t-2}, the availability at $t-2$
Result: $\hat{A}_t$, the predicted availability at time t

$\hat{A}_t \leftarrow 0$;
if $A_{t-1} > A_{t-2}$ ***or*** $A_{t-1} = 60$ **then**
    $\hat{A}_t \leftarrow 60$;
return $\hat{A}_t$;

3.3 Recurrent Neural Networks

As stated before, methods that use deep learning are advanced methods that are capable of capturing complex relationships between features. Recurrent Neural Networks (RNNs) are DNNs capable of learning temporal dependencies. In addition to the hidden neurons present in DNNs, RNNs contain context neurons. Context neurons can save the state of a hidden neuron and use it as input in the next iteration. Even though this makes them suitable for time series forecasting problems, each time step being included in every subsequent context layer makes RNNs especially vulnerable to the vanishing or exploding gradient problems.

The LSTM architecture solves these problems with the use of logic gates. These gates allow the neural network to forget unimportant information and retain information that is important. An LSTM unit consists of three gates: a forget gate, input gate, and output gate. The forget gate determines if the context neuron should clear its memory, the input gate determines if the context neuron should add the current state to its memory, and the output gate determines if the value of the context neuron is passed on to the output.

$$G_t = \sigma(W_{Gx}x_t + W_{Gh}h_{t-1} + b_G), G \in \{f, i, o\} \tag{1}$$

These gates are expressed in Equation 1, where f denotes the forget gate, i denotes the input gate, and o denotes the output gate. As expressed in the formulas, all gates decide based on the current input x_t and previous output h_{t-1}, adjusting the weight vectors W and bias b while training. The sigmoid function $\sigma()$ indicates the binary choice each gate has.

The calculation of the context is laid out in Equations 2 and 3:

$$\bar{c}_t = tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \tag{2}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \bar{c}_t \tag{3}$$

In these equations, $tanh()$ indicates the hyperbolic tangent function and $\circ$ is the Hadamard product operator. The candidate context at time t is denoted by $\bar{c}_t$, which is used for calculating the context c_t if the input gate equals 1. If the forget gate equals 0, the context of time $t-1$ is forgotten.

The context is then used in Equation 4 to determine the output for the current iteration h_t:

$$h_t = o_t tanh(c_t) \tag{4}$$

The way these formulas work together in determining the context and the output is presented visually in Figure 1. The nodes outside of the box represent the inputs and outputs of the LSTM unit, while the boxes represent the gates, Hadamard and addition operations or hyperbolic tangent transformations performed on the outputs of the gates.

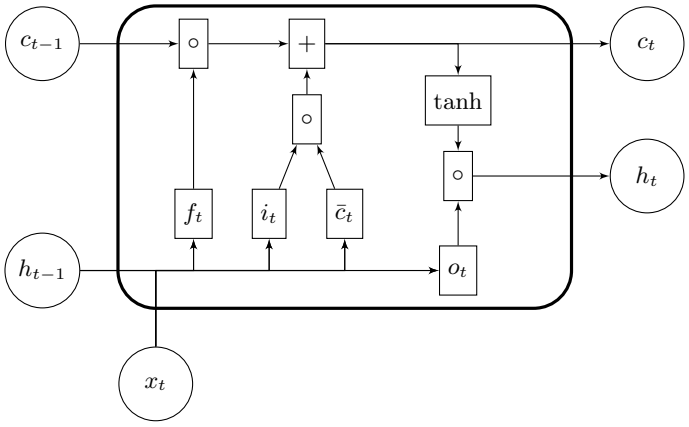

Fig. 1. Long Short-Term Memory

The GRU architecture consists of two gates: a reset gate, which is similar to a forget gate, and an update gate, which is similar to an input gate. The gates are defined similarly as in Equation 1, setting $G \in \{r, z\}$, where r denotes the reset gate and z denotes the input gate. The candidate output is defined in Equation 5:

$$\bar{h_t} = tanh(W_{hx}x_t + W_{hr}(r_t \circ h_{t-1}) + b_h) \tag{5}$$

The output is then determined as follows:

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \bar{h_t} \tag{6}$$

Instead of a separate context and output, the output of previous iterations poses as context to the model. In the calculation of the candidate output $\bar{h_t}$, the reset gate determines whether the context of the previous output should be forgotten. In the calculation of the current output h_t, the input gate chooses between the previous output and the candidate output.

The visual representation of a gated recurrent unit is shown in Figure 2. The lower amount of gates in the GRU architecture as compared to the LSTM architecture reduces model flexibility. However, it also reduces the time needed for training. Moreover, empirical evidence discussed in Section 2 suggests that the reduction in flexibility does not lead to lower prediction performance, though the comparison depends on the task at hand.

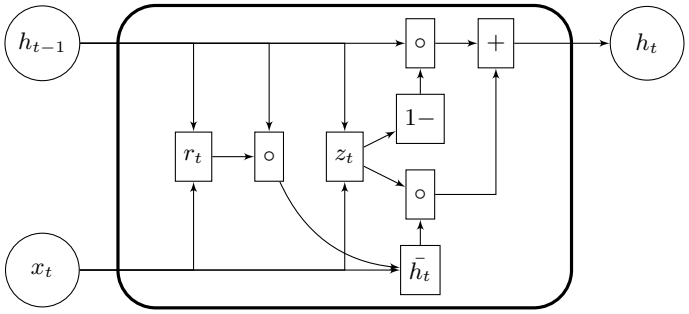

Fig. 2. Gated Recurrent Unit

3.4 Convolution

As mentioned earlier, convolutional layers can be prepended to both LSTM and GRU architectures to improve forecasting performance. While hidden layers are often fully connected, convolutional layers are sparsely connected, providing regularisation and reducing overfitting. A convolutional layer in a DNN applies an n-dimensional filter to the input. In the case of temporal convolution, the filter is one-dimensional, as opposed to the two- or three-dimensional filters typically applied in image-related tasks. Each filter in a convolutional layer performs a linear operation, smoothing the input vector.

Combining a convolutional layer and a layer containing either LSTM nodes or GRU nodes with one or more fully connected layers of hidden neurons and an output layer completes the architecture. The convolutional kernels introduce sparsity to the network and the LSTM and GRU nodes provide the ability to learn temporal relationships. The hidden neurons are able to learn complex relationships between the inputs. At the same time, the output layer transforms these relationships into predictions of the minutes a charger is available in a certain hour. A sigmoid activation function is used for the output to bind the predictions to fall between zero and one. The predicted amount of minutes of availability in an hour is then obtained by multiplying the prediction by sixty. This ensures the predictions have an upper bound of sixty minutes in an hour and a lower bound of zero minutes in an hour.

3.5 Infrastructure

The data is collected from a PostgreSQL database with SQL statements executed via Psycopg 3 [31]. Further programming is done in Python 3.9. Apart from Psycopg, the Python libraries used include NumPy [15], Pandas [25], and Keras [8] using the TensorFlow GPU backend.

4 Experiments

In this section, we first describe the experimental set-up and define the evaluation metric. Then, we report the results of the experiments, and compare them to the baseline performance. Finally, we discuss our results in the context of the existing related work.

4.1 Hyperparameter Tuning

As is the case with other DNNs, the CNN-LSTM and CNN-GRU architectures are highly flexible, have few hyperparameters and tune most parameters deterministically. However, the specific architecture implementation does need to be tuned. While there are more axes over which models can differ, this paper focuses on depth and breadth. Performing an extensive search over the vast number of possibilities is infeasible, as it would require training too many models. Therefore, a constrained grid search over both axis is performed. The search space for the depth is 1 or 2 fully connected layers and the search space for breadth is 32, 64 or 128 nodes in each layer. This results in six hyperparameter settings that are compared for each model.

The models are compared to each other and the baseline on the 2021 test set that was held out during training. The basis of comparison is the predictive performance as measured by the RMSE. The answer to the research question is found by comparing the RMSE of each respective model on the test data.

Algorithm 2: Incremental training

```
   input  : data, architecture, patience, batchsize
   output: best performing model on validation data
 1 initialise seen, model, best, tries;
 2 while tries < patience do
 3     unseen ← anti-join of data and seen;
 4     batch ← 2 * batchsize random items of unseen;
 5     train ← first half of batch;
 6     validate ← second half of batch;
 7     append train to seen;
 8     fit model with train;
 9     rmse ← root mean squared error of model on validate;
10     if rmse < best then
11         best ← rmse;
12         cache ← model;
13         tries ← 0;
14     else
15         increment tries;
16 return cache;
```

Because of the large number of charge points in the data, not the entire partition set apart for training is used. Instead, the training is performed incrementally, the specifics of which are laid out in Algorithm 2. Training with this procedure ensures that enough data is used to generalise well, while preventing the use of an unnecessarily large amount of data. This incremental training introduces randomness due to the random selection of data for training and validation. To mitigate this randomness, each parameter combination is fitted three times. While fitting more folds would give a better estimate of each architecture's performance, a trade-off needs to be made to prevent training on an unnecessarily large amount of data because of the repeated folds. For the purpose of this paper, the parameters of this incremental training scheme are set to a *patience* of 5 tries and a *batchsize* of 512 EVSEs.

4.2 Results

We now present our results, starting with the presentation of the baseline performance. To this purpose, the mean and standard deviation of the RMSE across EVSEs are reported. Then, the CNN-LSTM and CNN-GRU architectures are compared on their prediction performance. The average RMSE of the three folds trained with each hyperparameter setting is reported along with the standard deviation among folds.

The RMSE found with the baseline model is 9.659 minutes, with a standard deviation of 4.724 minutes. We use these results as a baseline reference for the more advanced methods.

Table 3. RMSE scores in minutes on all EVSEs in the test set for the CNN-LSTM and CNN-GRU models with varying depth and breadth. The lowest RMSE for each architecture is marked in bold.

		1 layer		2 layers	
Architecture	Breadth	mean	sd	mean	sd
CNN-LSTM	32 nodes	**1.225**	0.098	1.424	0.277
	64 nodes	1.493	0.056	1.497	0.135
	128 nodes	1.669	0.190	30.072	12.465
CNN-GRU	32 nodes	**1.336**	0.025	1.414	0.002
	64 nodes	1.521	0.048	1.577	0.001
	128 nodes	1.830	0.304	1.543	0.427

Results obtained using CNN-LSTM and CNN-GRU are shown in Table 3. From these results, it is clear that most parameter settings of both architectures resulted in a performance increase over the baseline. The results shown in bold are the lowest RMSE scores for the two architectures. These scores were both achieved on the hyperparameter settings that led to the least complex networks.

That is, the best predictive performance for both the CNN-LSTM and CNN-GRU architectures was found for networks with 32 nodes in each layer and a single fully connected layer.

While most other parameter settings resulted in a low RMSE between 1 and 2 minutes per hour, the CNN-LSTM architecture with the most extensive number of nodes and layers had a very high RMSE, showing no improvement over guessing. A likely explanation for the high RMSE of CNN-LSTM models with a breadth of 128 and a depth of 2 could be that such a broad model is prone to get stuck in local minima. Another reason could be that the model received too little data to train on, because of a *patience* or *batchsize* that were too low to allow the model to have learned some dependencies before stopping training. While performance might have been increased with a less strict choice of *patience* and *batchsize*, it remains unclear if that would result in a performance increase over the models that performed well with the current choice of *patience* and *batchsize*. Therefore, the increased computational cost of fitting with less restraining parameters is likely not offset by a better fitting model.

As the models were trained on a subset of the training data, the models' ability to generalise to new EVSEs can be tested as well. A similar performance on the testing data for EVSEs seen and unseen during training would indicate an ability of the model to generalise to new EVSEs as well as to new periods, whereas a gap in predictive performance would indicate that the models are able to generalise to new periods only for EVSEs seen during training.

Tables 4 and 5 present the predictive performances on the testing data for EVSEs seen during training and EVSEs not seen during training, respectively. As is apparent from these tables, the performances are very similar. The RMSE of predicting in new periods for EVSEs seen during training is generally a bit lower than that for EVSEs not seen during training. However, the small magnitude of the differences indicates that the models are able to generalise to unseen periods as well as to unseen EVSEs in unseen periods.

While the average RMSE remains more or less stable across these parts of the test set, the standard deviations of the RMSE across folds has fallen when testing on unseen EVSEs as compared to testing on seen EVSEs. This decrease in

Table 4. RMSE scores in minutes on seen EVSEs in the test set for the CNN-LSTM and CNN-GRU models with varying depth and breadth

		1 layer		2 layers	
Architecture	Breadth	mean	sd	mean	sd
CNN-LSTM	32 nodes	1.179	0.103	1.332	0.126
	64 nodes	1.498	0.115	1.467	0.155
	128 nodes	1.601	0.139	28.788	13.218
CNN-GRU	32 nodes	1.298	0.011	1.587	0.123
	64 nodes	1.664	0.080	1.525	0.247
	128 nodes	1.634	0.066	1.599	0.017

Table 5. RMSE scores in minutes on unseen EVSEs in the test set for the CNN-LSTM and CNN-GRU models with varying depth and breadth

		1 layer		2 layers	
Architecture	Breadth	mean	sd	mean	sd
CNN-LSTM	32 nodes	1.232	0.069	1.438	0.301
	64 nodes	1.492	0.049	1.501	0.176
	128 nodes	1.679	0.199	30.260	12.356
CNN-GRU	32 nodes	1.376	0.029	1.389	0.014
	64 nodes	1.500	0.065	1.585	0.011
	128 nodes	1.859	0.034	1.535	0.491

standard deviation can be intuitively explained by recognising the considerable overlap between the unseen EVSEs in the testing data for different folds. If there is much overlap between the seen EVSEs in different folds for the same model, then the set of unseen EVSEs is similar as well. However, even if there is little overlap of seen EVSEs in the training data, a large portion of the same EVSEs is still marked as unseen in both folds.

As most models show an improved RMSE over the baseline, we suspect that these models have the ability to predict a change in status, instead of only reacting to such a change. In order to substantiate this claim, we analyse the RMSE only on those hours where the availability changed. With these constraints, the performance of the baseline decreases to an RMSE of 33.455 minutes with a standard deviation of 2.000 across EVSEs. The results of testing the models on one fold for each hyperparameter combination are shown in Table 6.

From the table, it becomes apparent that performance typically fell compared to the performance on the entire testing data. However, the results still show an improvement over the baseline algorithm with the same constraints. This indicates that the models indeed learned to predict a status change, instead of merely reacting to one. The relative performance of most complex models increased on the subset of status changes. An explanation for this is that these models have not yet learned to predict mostly 0 or 60 minutes of availability.

Table 6. RMSE scores in minutes on all EVSEs in the test set for the CNN-LSTM and CNN-GRU models with varying depth and breadth. Only hours with a status change.

Architecture	Breadth	1 layer	2 layers
CNN-LSTM	32 nodes	15.170	13.567
	64 nodes	15.674	15.132
	128 nodes	12.979	20.879
CNN-GRU	32 nodes	12.638	15.409
	64 nodes	15.704	14.492
	128 nodes	14.767	16.415

4.3 Discussion

In this section, we further evaluate our experimental results and interpret them in relation to existing work. Finally, drawbacks of the current approach are given and suggestions are made for further research.

The results show little difference in the RMSEs of CNN-LSTM and CNN-GRU, which contrasts the little existing literature comparing these methods, based on which an improvement of CNN-GRU over CNN-LSTM was to be expected [10, 28]. As stated before, few studies have been performed comparing the CNN-LSTM and CNN-GRU architectures. Nonetheless, the performances of LSTM and GRU have been studied extensively in the last years, often showing little variation in prediction performance [12, 24, 34]. However, some literature shows GRU to have a slightly higher performance than LSTM [27, 33]. More research is needed to further establish in which cases one architecture is preferred over the other, both when comparing CNN-LSTM with CNN-GRU and LSTM with GRU.

Notwithstanding the promising results presented in this paper, there were certain drawbacks to the methodology used. One drawback is the scope of the periods used. The gross majority of the dataset falling during times of Covid-19 raises the question of the appropriateness of the trained models in the absence of a pandemic. As the pandemic caused many to work from home and reduce their movement, we can be confident that the charging behaviour was affected. Not being able to train on much data without this abnormal behaviour, the generalisation ability of the trained models on post-Covid periods remains unclear. However, it should be noted that the aim of this study is mostly to find which architecture is able to capture the temporal dependencies present in EVSE forecasting, not to produce a single end-all model. The insights from this paper remain important to take into account when training a model using post-pandemic data.

This work has focused solely on deep learning-based approaches. Section 2 gives an overview of other possible methodologies, some of which do not require using deep learning. A further approach that satisfies the requirement of generalisation to different EVSEs could be a more classical model, such as the autoregressive integrated moving average (ARIMA) model [3]. A drawback of such a general ARIMA model is that there could be a presence of different clusters of EVSEs. The methods employed in this study are able to recognise the charging behaviour of an EVSE as belonging to such a cluster and change its prediction strategy accordingly. However, ARIMA models would require the clustering of EVSEs, defining separate models for each cluster.

The possibility of time series clustering being required hinders the feasibility of a general ARIMA model, as time series clustering is a research field separate from time series prediction that comes with its own set of intricacies. An advantage of using a (set of) general ARIMA model(s) is that the reasons for a model to predict a certain availability are much more transparent.

In this study, little attention has been given to multistep forecasting. The deep learning-based models have been trained to predict one hour ahead. As

the only features additional to past outcome values are time-related, predicted outcomes can be used to predict further than one hour ahead. However, no assessment of the prediction performance more than one hour ahead has been made. Further research could expand upon the current work by testing the amount of periods the models can produce accurate forecasts for. Additionally, the hyperparameter search space could be extended to models with fewer nodes in each layer. As the best performing models were the models with the fewest nodes, further performance gains could be obtained by further simplifying the models, for example by only allowing 16 nodes per layer.

5 Conclusion

In this paper, we investigated the suitability of state-of-the-art deep learning methods for the task of predicting electric vehicle supply equipment availability. We formulated our research question as follows: *How do convolutional long short-term memory and convolutional gated recurrent units perform when forecasting electric vehicle supply equipment?* Our experiments showed that the CNN-LSTM typically had a lower RMSE than CNN-GRU, though their results were often similar. We can conclude that CNN-LSTM and CNN-GRU are close to each other in terms of performance when forecasting electric vehicle supply equipment, though CNN-LSTM was often able to slightly outperform CNN-GRU.

Going back to the main goal of this work, to find an accurate methodology for forecasting EVSE availability, the use of advanced (deep learning) methods has been shown to achieve this goal. In particular, we have demonstrated the ability of these methods to significantly outperform a simple, but strong, baseline. Being able to accurately predict EVSE availability will allow providers to take appropriate measures and facilitate the charging of electric vehicles. This, in turn, should lead to a further increase in the uptake of electric driving.

References

1. Ai, S., Chakravorty, A., Rong, C.: Household ev charging demand prediction using machine and ensemble learning. In: 2018 IEEE International Conference on Energy Internet (ICEI). pp. 163–168. IEEE (2018)
2. Bikcora, C., Refa, N., Verheijen, L., Weiland, S.: Prediction of availability and charging rate at charging stations for electric vehicles. In: 2016 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS). pp. 1–6. IEEE (2016)
3. Box, G.E., Jenkins, G.M., Reinsel, G.C., Ljung, G.M.: Time series analysis: forecasting and control. John Wiley & Sons (2015)
4. Broadbent, G.H., Metternicht, G.I., Wiedmann, T.O.: Increasing electric vehicle uptake by updating public policies to shift attitudes and perceptions: Case study of new zealand. Energies **14**(10) (2021). https://doi.org/10.3390/en14102920, `https://www.mdpi.com/1996-1073/14/10/2920`
5. Buzna, L., De Falco, P., Ferruzzi, G., Khormali, S., Proto, D., Refa, N., Straka, M., van der Poel, G.: An ensemble methodology for hierarchical probabilistic electric

vehicle load forecasting at regular charging stations. Applied Energy **283**, 116337 (2021)
6. Chen, L., Chen, W., Wu, B., Zhang, Y., Wen, B., Yang, C.: Learning from multiple time series: A deep disentangled approach to diversified time series forecasting. arXiv preprint arXiv:2111.04942 (2021)
7. Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches (2014). https://doi.org/10.48550/ARXIV.1409.1259, https://arxiv.org/abs/1409.1259
8. Chollet, F., et al.: Keras (2015), https://github.com/fchollet/keras
9. Das, H., Rahman, M., Li, S., Tan, C.: Electric vehicles standards, charging infrastructure, and impact on grid integration: A technological review. Renewable and Sustainable Energy Reviews **120**, 109618 (2020)
10. Dua, N., Singh, S.N., Semwal, V.B.: Multi-input cnn-gru based human activity recognition using wearable sensors. Computing **103**(7), 1461–1478 (2021)
11. Foley, B., Degirmenci, K., Yigitcanlar, T.: Factors affecting electric vehicle uptake: Insights from a descriptive analysis in australia. Urban Science **4**(4) (2020). https://doi.org/10.3390/urbansci4040057, https://www.mdpi.com/2413-8851/4/4/57
12. Fu, R., Zhang, Z., Li, L.: Using lstm and gru neural network methods for traffic flow prediction. In: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC). pp. 324–328. IEEE (2016)
13. Gasparin, A., Lukovic, S., Alippi, C.: Deep learning for time series forecasting: The electric load case. arXiv preprint arXiv:1907.09207 (2019)
14. Hardman, S., Jenn, A., Tal, G., Axsen, J., Beard, G., Daina, N., Figenbaum, E., Jakobsson, N., Jochem, P., Kinnear, N., et al.: A review of consumer preferences of and interactions with electric vehicle charging infrastructure. Transportation Research Part D: Transport and Environment **62**, 508–523 (2018)
15. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**, 357–362 (2020). https://doi.org/10.1038/s41586-020-2649-2
16. He, Y., Liu, Z., Song, Z.: Optimal charging scheduling and management for a fast-charging battery electric bus system. Transportation Research Part E: Logistics and Transportation Review **142**, 102056 (2020)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
18. Huang, Z., Yang, F., Xu, F., Song, X., Tsui, K.L.: Convolutional gated recurrent unit–recurrent neural network for state-of-charge estimation of lithium-ion batteries. IEEE Access **7**, 93139–93149 (2019). https://doi.org/10.1109/ACCESS.2019.2928037
19. Irle, R.: Global ev sales for 2021 h1 (2021), https://www.ev-volumes.com/country/total-world-plug-in-vehicle-volumes/
20. Kao, I.F., Zhou, Y., Chang, L.C., Chang, F.J.: Exploring a long short-term memory based encoder-decoder framework for multi-step-ahead flood forecasting. Journal of Hydrology **583**, 124631 (2020)
21. Karakatič, S.: Optimizing nonlinear charging times of electric vehicle routing with genetic algorithm. Expert Systems with Applications **164**, 114039 (2021)
22. Kim, M., Choi, W., Jeon, Y., Liu, L.: A hybrid neural network model for power demand forecasting. Energies **12**(5), 931 (2019)

23. Kim, T.Y., Cho, S.B.: Predicting residential energy consumption using cnn-lstm neural networks. Energy **182**, 72–81 (2019)
24. Liu, X., Lin, Z., Feng, Z.: Short-term offshore wind speed forecast by seasonal arima-a comparison against gru and lstm. Energy **227**, 120492 (2021)
25. McKinney, W., et al.: Data structures for statistical computing in python. In: Proceedings of the 9th Python in Science Conference. vol. 445, pp. 51–56. Austin, TX (2010)
26. O'Neill, E., Moore, D., Kelleher, L., Brereton, F.: Barriers to electric vehicle uptake in ireland: Perspectives of car-dealers and policy-makers. Case Studies on Transport Policy **7**(1), 118–127 (2019). https://doi.org/https://doi.org/10.1016/j.cstp.2018.12.005, https://www.sciencedirect.com/science/article/pii/S2213624X18301500
27. Raza, M.R., Hussain, W., Merigó, J.M.: Cloud sentiment accuracy comparison using rnn, lstm and gru. In: 2021 Innovations in Intelligent Systems and Applications Conference (ASYU). pp. 1–5. IEEE (2021)
28. Sajjad, M., Khan, Z.A., Ullah, A., Hussain, T., Ullah, W., Lee, M.Y., Baik, S.W.: A novel cnn-gru-based hybrid approach for short-term residential load forecasting. Ieee Access **8**, 143759–143768 (2020)
29. Straka, M., Carvalho, R., Van Der Poel, G., Buzna, L.: Analysis of energy consumption at slow charging infrastructure for electric vehicles. IEEE Access **9**, 53885–53901 (2021)
30. Tian, C., Ma, J., Zhang, C., Zhan, P.: A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network. Energies **11**(12), 3493 (2018)
31. Varrazzo, D.: Psycopg (2022), https://github.com/psycopg/psycopg
32. Wang, R., Li, C., Fu, W., Tang, G.: Deep learning method based on gated recurrent unit and variational mode decomposition for short-term wind power interval prediction. IEEE Transactions on Neural Networks and Learning Systems **31**(10), 3814–3827 (2020). https://doi.org/10.1109/TNNLS.2019.2946414
33. Yamak, P.T., Yujian, L., Gadosey, P.K.: A comparison between arima, lstm, and gru for time series forecasting. In: Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence. pp. 49–55 (2019)
34. Yang, S., Yu, X., Zhou, Y.: Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example. In: 2020 International workshop on electronic communication and artificial intelligence (IWECAI). pp. 98–101. IEEE (2020)
35. Yao, J., Xiong, S., Ma, X.: Comparative analysis of national policies for electric vehicle uptake using econometric models. Energies **13**(14) (2020). https://doi.org/10.3390/en13143604, https://www.mdpi.com/1996-1073/13/14/3604