energies

MDPI

Article

Machine Learning-Based Management of Electric Vehicles Charging: Towards Highly-Dispersed Fast Chargers

Mostafa Shibl [1,*], Loay Ismail [2] and Ahmed Massoud [1]

[1] Department of Electrical Engineering, Qatar University, Doha 2713, Qatar; ahmed.massoud@qu.edu.qa
[2] Department of Computer Science and Engineering, Qatar University, Doha 2713, Qatar; loay.ismail@qu.edu.qa
* Correspondence: ma1902206@qu.edu.qa

Received: 15 August 2020; Accepted: 13 October 2020; Published: 17 October 2020

Abstract: Coordinated charging of electric vehicles (EVs) improves the overall efficiency of the power grid as it avoids distribution system overloads, increases power quality, and decreases voltage fluctuations. Moreover, the coordinated charging supports flattening the load profile. Therefore, an effective coordination technique is crucial for the protection of the distribution grid and its components. The substantial power used through charging EVs has undeniable negative impacts on the power grid. Additionally, with the increasing use of EVs, an effective solution for the coordination of EVs charging, particularly when considering the anticipated proliferation of EV fast chargers, is imminently required. In this paper, different machine learning (ML) approaches are compared for the coordination of EVs charging. The ML models can predict the power to be used in EVs charging stations (EVCS). Due to its ability to use historical data to learn and identify patterns for making future decisions with minimal user intervention, ML has been utilized. ML models used in this paper are (1) Decision Tree (DT), (2) Random Forest (RF), (3) Support Vector Machine (SVM), (4) Naïve Bayes (NB), (5) K-Nearest Neighbors (KNN), (6) Deep Neural Networks (DNN), and (7) Long Short-Term Memory (LSTM). These approaches are chosen as they are classifiers known to have the leading results for multiclass classification problems. The results found shed insight on the importance of the techniques used and their high potential in providing a reliable solution for the coordinated charging of EVs, thus improving the performance of the power grid, and reducing power losses and voltage fluctuations. The use of ML provides a less complex method to coordinate EVs, in comparison with conventional optimization techniques such as quadratic programming, and the use of ML is faster as it requires less computational power. LSTM provided the best results with an accuracy of 95% for predicting the most appropriate power rating (PR) for EVCS, followed by RF, DT, DNN, SVM, KNN, and NB. Additionally, LSTM was also the model with the smallest error rate, at a value of $\pm$0.7%, followed by RF, DT, KNN, SVM, DNN, and NB. The results obtained from the LSTM model were similar to the results obtained from past literature using quadratic programming, with the increased speed and simplicity of ML.

Keywords: coordinated electric vehicles charging; cyber-physical systems (CPSs); Decision Tree (DT); Deep Neural Network (DNN); electric vehicles charging stations (EVCS); K-Nearest Neighbors (KNN); Long Short-Term Memory (LSTM); machine learning (ML); Naïve Bayes (NB); power rating (PR); Random Forest (RF); Recurrent Neural Networks (RNN); smart grid; Support Vector Machine (SVM)

1. Introduction

The electric vehicle (EV) industry is rapidly expanding with more countries adopting electric vehicles charging stations (EVCS). A total of approximately 5.2 million charging stations are now

available worldwide, which is an increase of 44% from 2018 [1]. In addition, the private sector is moving with high momentum to electrify all types of transportation, including buses and trucks [1]. As a result, it is vital to be able to control the charging stations by coordinating the charging of EVs in a way that protects the power grid and its components. Cyber-physical systems, which consist of both software and hardware, are needed to have a connected ecosystem that allows the control and monitoring of the EVCS. Therefore, the proposed system will be based on Internet of Things (IoT) technology which is built upon sensing, processing, communicating, and actuating. As a result, the presented system will have low complexity and low computational power to be able to easily integrate it successfully with IoT-architecture infrastructure, to ensure the system's effectiveness and scaling possibilities.

The current EVCS have standard mechanisms of either conventional or fast charging options. Despite permission from the utility is required to be able to install Level 2 EVCS, the numerous number of chargers can still have a negative impact on the grid. Thus, for the uncoordinated charging of EVs, where the user is allowed to choose between the fast and conventional charging without there being a system that checks for potentially harmful consequences of the desired option, the power grid might face unfortunate consequences such as power outages due to sudden power surges and transformer overloads, as the power of the EVCS is not closely monitored or controlled. Accordingly, an effective solution to manage the power of EVCS is highly required. Many issues may arise during the process, which include the need to predict the real loading accurately. Moreover, the distribution system should deliver power reliably, efficiently, cost-effectively, and with the best performance.

EV charging methods have seen many developments in recent years. Wireless dynamic charging has gained popularity due to its ability to charge EVs while moving, which decreases charging wait time at EVCS [2]. On the contrary, wireless dynamic charging requires a costly change in infrastructure and is not off-the-shelf. In addition, the use of public lighting has been examined for the charging of EVs [3]. However, this requires the modification of street lamps in order to be able to turn them into EVs charging stations. Thus, such methods can be expensive and are still not highly embraced in the EVs market, compared to standard EVCS. Nevertheless, the presented system in this paper can adopt such charging methods as loads and control their operation in the same way as standard EVCS.

Additionally, EV penetration level is the maximum power used for charging EV as a percentage of the average power of the normal load curve. As the penetration level increases, the load curve shifts upwards, which causes issues such as voltage fluctuations and transformer overloads. As a result, an effective technique that allows the coordination of EVs charging is pivotal to be able to avoid these negative repercussions from the uncoordinated charging of EVs. Current solutions for the coordination of EVs charging include conventional optimization techniques, such as quadratic programming and dynamic programming. However, the main limitations of these solutions are the need for complex mathematical problem formulation and the high computational intensity to provide solutions. Thus, machine learning (ML) is utilized to provide a less computationally intensive system and to eliminate the process of mathematical problem formulation.

In this paper, a range of ML techniques and classifiers are assessed to synthesize a system that can predict the optimum power of EVCS based on the current power usage of residential households, in order to optimize the use of the power grid and protect it from possible harm. ML is used to help with choosing the most appropriate operation mode of EVCS automatically with minimal human intervention. The only decision made by the user is the time of charging; however, the user will still be able to override the system controls and change the mode of charging. It uses patterns and trends in historical data to be able to label unforeseen data and make future decisions with high accuracy levels. ML techniques used in this paper are Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), Naïve Bayes (NB), Deep Neural Networks (DNN), and Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN). The proposed system uses analysis of power usage in an area and predicts the power for EVCS accordingly.

Figure 1 shows the overall sequential diagram of the presented experimental method.

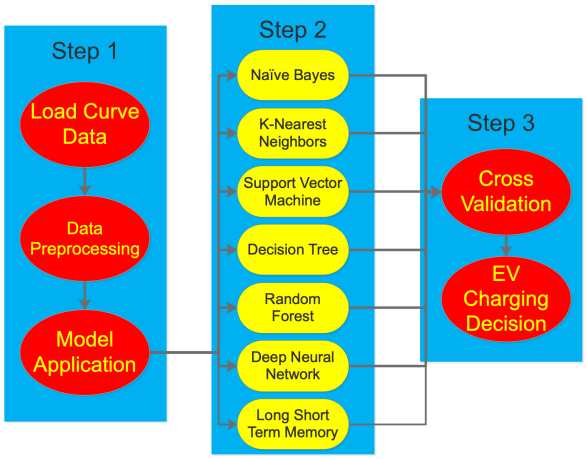

Figure 1. Sequential diagram of the presented experimental method.

This paper outlines a method for the coordinated fast charging of EVs using ML. The main contributions of this paper are as below.

1. Performance assessment of different ML approaches for coordinated EVs charging considering both conventional and fast charging.
2. Using ML to choose the most appropriate mode of operation for the EVCS, which include fast charging, conventional charging, and supporting the grid (discharging), and comparing the use of ML with quadratic programming for minimizing load variance while operating EVCS.

In the upcoming sections, Section 2 surveys the literature. Section 3 investigates the EVCS and distribution grid model. Section 4 explores the mathematical model of the problem. Section 5 discusses the different ML techniques. Section 6 presents the research methodology, Section 7 examines the results obtained by the different models, and Section 8 inspects the effect of coordinated charging of EVs on the grid. Section 9 presents the conclusion of the paper, and Section 10 presents the prospective future work.

2. Related Work

This paper addresses three main research issues namely EVCS, decision support systems, and the coordinated charging of EVs, which are surveyed in the following subsections.

2.1. Electric Vehicles Charging Stations

Much research has been conducted to advance and develop EVCS. Khan et al. [4] studied the integration of renewable energy in EVCS to improve power quality and decrease voltage deviations.

In another study by Liu et al. [5], the use of wind energy in EVCS, which is similar to the work done by Khan et al. [4], was researched. A model was proposed based on the scheduling of charging times using different scheduling patterns. The results portrayed that the operation costs are decreased and the model is able to compensate for fluctuations from the wind power generation. Moreover, the system enabled the load shifting and shaving, and was able to attain a constant load profile.

Badawy and Sozer [6] examined a way to decrease operation costs by using a photovoltaic (solar-powered) battery-powered EVCS. An operating cost objective function was formulated to be minimized. The objective function considered both electricity grid prices and the cost of battery degradation. Thus, a model for the cost of battery degradation was also produced, which took into consideration the temperature of operation, the average state of charge, and the cycle depth of discharge. The state of charge, load forecast, and grid prices are set using particle swarm optimization. Additionally, dynamic programming is used to control the power flow in short time intervals. Moreover, it is utilized for compensating errors when the predicted data is different from

the measured, real data. The results showed the usefulness of the model and its effectiveness using different conditions.

In comparison with the aforementioned work, the work conducted in this paper provides an outline for a method to change the power of the EVCS automatically using ML.

2.2. Decision Support Systems and Prediction Modeling

Many ML models have been used for the purpose of predictions based on a specific set of data. The models are usually used for fault diagnosis and detection.

Mohsenzadeh et al. [7] compared the use of a Bayesian ML algorithm for classification problems. The proposed model was an extension of the work done on relevant vector machine algorithms. The results showed that the algorithm can perform with high accuracy; however, the results' accuracy was close to the accuracies of other algorithms. The main advantage of this algorithm compared to other ML classifiers is the decreased complexity and it is less prone to overfitting.

In addition, a Bayesian Network that is privacy-aware to discover the future trustworthy IoT applications was presented by Li and Oechtering [8]. Furthermore, Viegas et al. [9] proposed a system that could predict events, such as outages, in the smart grid using Bayesian methods. In addition to a Bayesian approach, the RF algorithm was used to allow the prediction of events of interest. Similarly, Xu and Xu [10] investigated the creation of a system that utilizes a Bayesian Network for health management diagnosis for space avionics.

Another system was presented by Ali et al. [11], which consists of a Bayesian Network that could choose the most appropriate power links for power quality meter placement. In addition, Yu et al. [12] created a system that has the ability to monitor the health of hybrid systems with more than one fault. The model is able to predict whether a fault has occurred or not and the possible solutions for that fault using a Bayesian Network.

Siryani et al. [13] produced a decision support system for operating electric smart meters. The system can reduce the cost of operating such meters by analyzing the communication quality data and deciding whether or not a technician is needed to visit the site to solve a problem. In this study, four ML algorithms were used, which are Bayesian Networks, NB, DT, and RF. RF showed the most promising results with the highest accuracy scores.

Additionally, Ramona et al. [14] studied the use of SVM for multiclass feature selection while optimizing the classifier for cost and complexity. This was done using kernel target alignment and kernel class separability. The results showed that the use of kernel target alignment is more optimized than kernel class separability. However, the models only work well with linearly separable data and suffer when working with nonlinear data classification.

Furthermore, Deligiannis et al. [15] used RF due to its high accuracy and ease of use. ML was utilized to predict and forecast energy consumption and power demand through the use of IoT and smart meters. The results were notable and portrayed that energy consumption can be predicted with a mean absolute percentage error of 16% when using RF. Moreover, RF provided better results compared with an autoregressive model.

In this paper, a multiclass classification problem is devised. Thus, a range of ML classifiers that provided good results for similar multiclass classification problems in past research is used for changing the power of EVCS automatically. After that, cross-validation is utilized to assess the performance of the different models.

2.3. Coordinated Charging of Electric Vehicles

There have been many studies, advancements, and much research on the topic of coordinated charging of EVs. Clement-Nyns et al. [16] studied the effect of uncoordinated charging of EVs. The results showed that power loss and voltage deviations occurred continuously. However, a solution involving the use of quadratic programming to solve the optimization problem of minimizing power

loss was proposed and the results showed that power losses were decreased. It was also able to reduce voltage deviations.

Deilami et al. [17] researched the use of a smart real-time load controlling algorithm to be able to coordinate the charging of EVs by allowing the charging of plug-in EVs in the shortest time to minimize cost, while maintaining the voltage profile and generation limits. The approach used the maximum sensitivities selection optimization. The results showed that the developed algorithm improves the efficiency of smart grids, increases the reliability of the power source to the user, and reduces power overloads when compared with other current coordination approaches.

In a study by Wang et al. [18], the possibility of maximizing the distributed energy accommodation capacity was investigated. Depending on the time, the user is provided with the most appropriate location for charging. When using this algorithm, EVs are considered moving loads and are coordinated so that the optimal distribution of energy is achieved in the power grid.

Tang et al. [19] investigated the utilization of an online system for charging decisions. The algorithm reduces the energy cost by spreading out the energy consumption over time, without the need for any information relevant to future power usage. The algorithm was able to ensure the fulfillment of all energy demands at all times.

Research by Sun et al. [20] conveyed that power balancing issues can be solved by the use of an algorithm that enables distribution storage units to control their own charging and discharging status. A novel version of the Lyapunov optimization framework was used.

Li et al. [21] explored the use of game theory for the coordination of EVs charging. The algorithm used in this study was the Newton-type algorithm to find the Nash equilibrium of the game model. Despite the success of the model, the use of game theory is not an effective solution as it still allows the user the ability to charge during peak times but with a higher price.

Moreover, Rawat and Niazi [22] compared different techniques for charging EVs from the three different viewpoints, which were the customer's viewpoint, the operator's viewpoint, and the customer's and operator's viewpoint. The results showed that from the customer's viewpoint, the charging cost was decreased and it improved the load curve for slow charging. When using fast charging or high penetration EVs, disadvantageous effects took place in the load curve. Regarding the operator's viewpoint, the results showed that a strategy that flattens the load curve and avoids power losses and overloads was favored. Thus, a load variance function was minimized to be able to increase the overall efficiency of the system. From the viewpoint of both the operator and the customer, the coordinated charging was the best strategy as the strategy works in favor of both stakeholders as a multi-objective function was utilized.

A study by Leou [23] considered the power system constraints and costs of operation for the optimum charging control of EVs. The three systems were the coordination of charging and discharging, coordination of charging only, and uncoordinated charging. Results showed that the most optimum system for minimizing the operation costs was the coordination of charging and discharging. Furthermore, the system helped with improving the load curve as the grid was supported by the EVs during peakload times. However, taking into consideration the costs from battery degradation caused by continuous charging and discharging, the utility must buy the energy at a price of 150% of the electricity price at peakload times, to make the system effective and implementable.

Research by Jian et al. [24] studied the coordinated charging of EVs by optimization and minimizing the load variance. The downside of the model used was that assumptions were made for simplicity, which makes the model unrepresentative compared to real-life, practical situations. On the contrary, the model made was able to decrease the variance of load curves drastically with the use of quadratic programming as the main optimization technique. In addition, the energy losses were taken into consideration and the performance of the model was relatively high.

Additionally, Yao et al. [25] had a hierarchical decomposition approach for the coordination of charging EVs in EVCS. The total cost of operation was minimized by sorting out generators and EV aggregators at the same time. The coordinated charging and discharging mechanism was

based on the bi-level programming theory. The simulation results showed that the model usually turned into a coordinated charging model and discharging was only used when the load curve increased significantly.

Furthermore, Shaaban et al. [26] conveyed the possibility of an online and real-time EV charging coordination system. In [26], both minimizing operation costs and maximizing customer satisfaction were considered. The two-step formulated optimization problem considers the maximization of the customer's satisfaction first. The proposed model excelled especially during high EV penetration levels. However, the system only considered a single charging speed, which means the use of fast charging, conventional charging, and discharging was not studied.

Erdogan et al. [27] researched the peak shaving of the load curve using the discharging of EVs. A two-stage model was proposed, where the times of discharging and the level of peak shaving were first forecasted, then real-time monitoring was utilized to decide times of discharging based on current load levels. In this model, an optimization problem was also formulated for the mean square error and minimized to have effective peak shaving. The advantage of the model was its adaptability depending on the load curve, even during low penetration levels. Further, it provided results that are close to the optimum solution while maintaining low computational costs.

Moreover, the use of stochastic characterization for the creation of an EV charging demand profile through voltage usage and user charging profiles has been examined by Hong et al. [28]. The study was based on highly populated residential areas in metropolitan cities. In order to decrease the aging of the transformer, photovoltaic sources of energy were utilized. The results showed positive effects on transformers, especially when paired with energy storage systems, to decrease aging even at high EVs penetration levels. The main downsides of the proposed system are that photovoltaic cells might not provide enough power in some countries, which decreases their efficiency, as well as their expensive cost.

The main drawback of all the previously mentioned work is the use of conventional optimization techniques such as quadratic programming for minimizing load variance and power losses, which requires complex problem formulation and solving methods. The use of ML is less complex as the only requirement is having an appropriate dataset. The ML algorithms run automatically and difficult problem formulation and careful consideration of optimization constraints are not required.

The proposed system in this paper focuses on creating a framework for improving EVCS, by automatically changing their power depending on the loading on the power grid by the surrounding residential loads.

Additionally, the system is expected to improve the operation of the power grid by reducing power losses and voltage fluctuations, which occur due to the sudden increases in power usage from EVCS. Thus, these overloads can be avoided by closely monitoring and coordinating the charging of EVs. As a result, the performance of the power grid increases, and the power quality and supply provided to the user is sufficient.

In essence, the difference between the presented method and previous work is that the framework created in this paper uses an IoT-based ecosystem to be able to coordinate the charging of EVs using ML and power usage data.

3. Distribution Network and Electric Vehicles Charging Stations Operation

Figure 2 reveals an overview of the employed distribution network.

Figure 3 shows a sequence diagram explaining how the ML model will be utilized. When the EVCS is operational, the historical normal load data are processed into the ML model and the time periods available for charging are found. The EVCS then waits for a connection with an EV. Once a connection is established, the expected time of charging is found, depending on the state of charge needed at the end. After that, the time of charge is used with the historical total power usage data to find the most appropriate power of the EVCS using the ML model. If the predicted power by the model is high, fast charging is used. If it is normal, conventional charging is used. If it is low, the EV is used to support

the grid through discharging. Subsequently, if the end of the charging time is reached, then the EV is disconnected. If not, then the system continuously checks whether the power should be changed or not depending on the current time and historic total power usage, until the end of the charging time is reached.

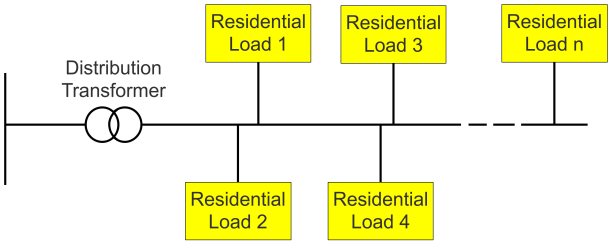

Figure 2. Schematic of the employed distribution network.

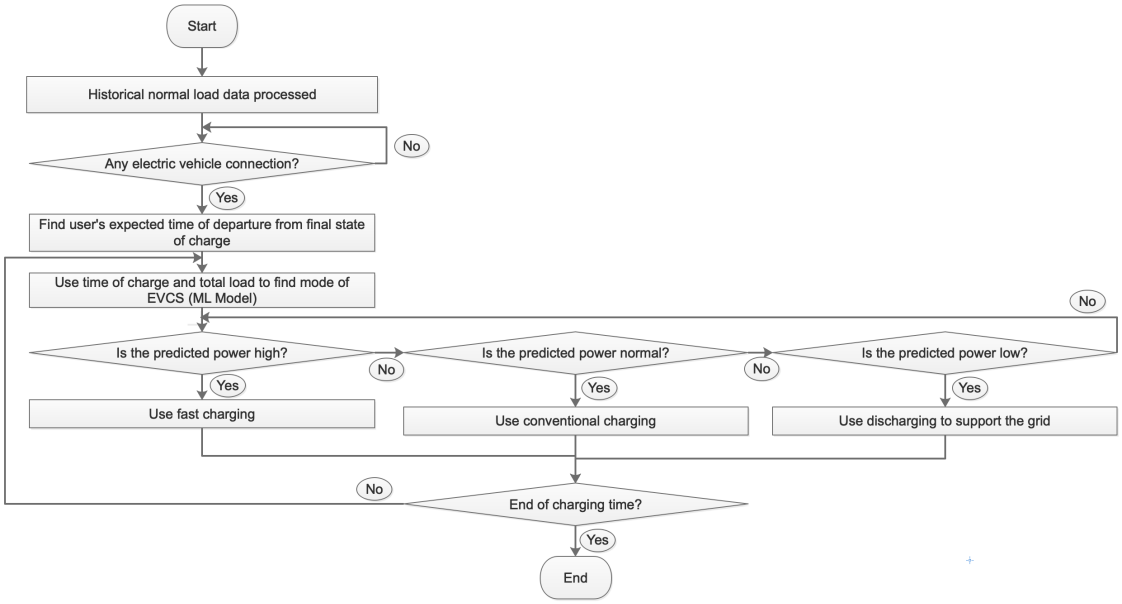

Figure 3. Sequence diagram of the operation of ML model.

4. Optimization Problem

The uncoordinated charging of EVs and its disadvantages can be solved by minimizing the load variance. Thus, the problem that is required to be solved in this paper can be put in mathematical terms. As a result, the equation formed can be solved as an optimization problem. The expression in Equation (1) shows the objective function of the problem, which is the load variance, where the equation is expected to be minimized. The inequalities in Equations (2)–(4) are the constraints for the problem. Additionally, Equation (5) represents how the charging quantity is calculated. Equation (6) shows how the mean of the power used by the microgrid is calculated. In addition, Equation (7) defines the sign function for the charging power. Finally, Equation (8) conveys the calculation for the state of charge of the EVs.

The load variance in the power grid is minimized, which helps with increasing power quality and decreasing voltage fluctuations. Thus, the load curve will be flattened and no overloads will occur in the power grid.

The day is divided into T time periods. Each time period is denoted Δt. Moreover, the assumption that each EV is charged at the start of a time period and is fully charged at the end of another time period is used. As a result, the objective function and the constraints can be formed as follows.

$$\underset{P^S_{t,n}}{min} \quad \frac{1}{T}\sum_{t=1}^{T}\left(P^A_t + \sum_{n=1}^{N} P^S_{t,n} - \mu^G_T\right)^2 \tag{1}$$

$$P^A_t + \sum_{n=1}^{N} P^S_{t,n} \le P^G_{t,max} \quad (t = 1 \sim T) \tag{2}$$

$$\begin{gathered} -P^S_{n,max} \le P^S_{t,n} \le P^S_{n,max} \\ (t = \tau^i_n \sim \tau^f_n; \quad n = 1 \sim N) \end{gathered} \tag{3}$$

$$\begin{gathered} SoC_{min,n} \le SoC_{t,n} \le SoC_{max,n} \\ (t = \tau^i_n \sim \tau^f_n; \quad n = 1 \sim N) \end{gathered} \tag{4}$$

$$\begin{gathered} \sum_{t=\tau^i_n}^{\tau^f_n}\left((\eta^S_n)^{sn}_{t,n} P^S_{t,n}\Delta t\right) = \Delta W_n = W^f_n - W^i_n \\ (n = 1 \sim N) \end{gathered} \tag{5}$$

$$\mu^G_T = \frac{1}{T}\sum_{t=1}^{T}\left(P^A_t + \sum_{n=1}^{N} P^S_{t,n}\right) \tag{6}$$

$$\begin{gathered} sn_{t,n} = \begin{cases} 1 & P^S_{t,n} \ge 0 \\ -1 & P^S_{t,n} < 0 \end{cases} \\ (t = \tau^i_n \sim \tau^f_n; \quad n = 1 \sim N) \end{gathered} \tag{7}$$

$$\begin{gathered} SoC_{t,n} = \begin{cases} \frac{W^i_n + P^S_{t,n}(\eta^S_n)^{sn}_{t,n}\Delta t}{Q_n} & t = \tau^i_n \\ SoC_{t-1,n} + \frac{P^S_{t,n}(\eta^S_n)^{sn}_{t,n}\Delta t}{Q_n} & t = \tau^i_n + 1, \ldots, \tau^f_n \end{cases} \\ (n = 1 \sim N) \end{gathered} \tag{8}$$

where

Δt denotes the time period in minutes;
T denotes the total number of time periods used in one day;
N denotes the number of EVs;
τ^i_n denotes the integer label of the time period when the n-th EV is plugged into the EVCS;
τ^f_n denotes the integer label of the time period when the n-th EV is removed from the EVCS;
P^A_t denotes the power of the loads at the t-th time in kW;
$P^S_{t,n}$ denotes the power of operation of the n-th charging station at the t-th time period in kW;
$sn_{t,n}$ denotes the function that determines the sign of $P^S_{t,n}$;
$P^G_{t,max}$ denotes the maximum power of operation of the microgrid at the t-th time period in kW;
$P^S_{n,max}$ denotes the maximum power of operation of the n-th charger at the t-th time period in kW;
η^S_n denotes the efficiency of the n-th charger;
μ^G_T denotes the mean power of the distribution network in one day in kW;
ΔW_n denotes the amount of charging required for the n-th EV in kW;
W^i_n denotes the initial amount of charge of the battery when the n-th EV is plugged into the EVCS in kWh;
W^f_n denotes the final amount of charge of the battery when the n-th EV is removed from the EVCS in kWh;
$SoC_{t,n}$ denotes the state of charge of the n-th EV at the t-th time period;

$SoC_{min,n}$ denotes the minimum value that is permitted for the state of charge of the n-th EV;
$SoC_{max,n}$ denotes the maximum value that is permitted for the state of charge of the n-th EV;
Q_n denotes the capacity of the battery of the n-th EV in kWh.

The operating power of the EVCS during each time period is determined. Moreover, due to the calculations in Equations (5)–(8), the optimization problem is nonlinear.

To solve the quadratic programming problem quickly, polynomial time can be utilized, which is a method discussed in [29]. Therefore, the optimization problem formed in Equations (1)–(8) is a polynomial time–space complexity optimization problem.

The solution to the optimization problem has many cases and depends on the values of the parameters. The parameters are determined based on the number of EVs, their states of charge, their battery capacity, and the type of charging station. The type of charging station affects the solution as the powers of EVCS are different. Moreover, the capacity of the battery affects the solution as it affects the state of charge.

The aforementioned solution to the optimization problem using quadratic programming is the current and most popular method for minimizing the load variance equation. Therefore, it is the present method for optimizing the operation of EVCS. However, such a method is complex and time-consuming. This is due to the need for careful consideration of the system to be able to create the most appropriate objective function, and the equations and inequalities for all the constraints. As a result, the use of ML as an optimization technique for minimizing load variance is introduced in this paper, as it does not require the formulation of a mathematical model for the system.

The use of ML as an optimization technique instead of quadratic programming to minimize load variance has several benefits. These include but are not limited to ML being easier to use and having more prospective applications, such as predicting the condition of the battery and the most appropriate method of charging for the battery. ML uses historical data to be trained and to identify patterns and trends to be able to make decisions from unlabeled data with high accuracy, therefore minimal human interaction is required. In addition, if some constraints or parameters change, the system can be retrained with the new data easily, which makes ML more adaptive as the formulation of a new problem with the new constraints is not required. Moreover, ML can be utilized for other decisions such as deciding peak shaving time periods and choosing appropriate times to use EVCS to support the grid. As a result, ML is a better method as it is faster, less complex, and more flexible.

5. Machine Learning Techniques

Seven different ML techniques are used in this paper. As mentioned before, the ML models used are NB, KNN, SVM, DT, RF, DNN, and LSTM. A summary of each algorithm is outlined and explained in this section. The different ML models are used in order to allow the comparison and contrasting between the different models. All the ML techniques used are supervised learning techniques, therefore all of the data utilized in this paper is labeled.

5.1. Naïve Bayes

The NB model assumes every predictor variable is independent. Thus, this means that it uses Bayes' Theorem in a naive way, which means it is a special case of a Bayesian Network.

NB uses Bayesian statistics, which is based on Bayes' Theorem. It predicts the class of a test case with values $T = a_1, a_2, a_3, \ldots, a_n$ by choosing the class c_i, which optimizes Equation (9) by maximizing it [30]. In Equation (9), $P(c_i|T)$ denotes the conditional probability of class c_i given test case T.

$$P(c_i|T) = \frac{P(T, c_i)}{P(T)} = \frac{P(c_i) \cdot P(T|c_i)}{P(T)} \tag{9}$$

In this model, the probability estimation of each class is provided by the training set. As the model tries to maximize Equation (9), the denominator is reduced as much as possible so it can be removed from the equation as it does not have a direct contribution to the prediction process.

In addition, NB classifier is able to calculate the joint probability $P(T|c_i)$ by assuming that all the attributes are conditionally independent given the class c_i [30]. Therefore, Equation (10) shows how the model calculates the joint probability. This shows that the predictions made by the model depend only on the values of $P(c_i)$ and $P(a_j|c_i)$.

$$\begin{aligned} P(T|c_i) &= P(a_1, a_2, a_3, \ldots, a_n, c_i) \\ &\approx P(c_i)P(a_1|c_i)P(a_2|c_i)P(a_3|c_i)P(a_n|c_i) \\ &= P(c_i) \cdot \prod_{j=1}^{n} P(a_j|c_i) \end{aligned} \tag{10}$$

The main advantage of using NB is that it only needs small datasets for training. Moreover, training the model is fast and is not time-consuming.

5.2. K-Nearest Neighbors

KNN is a popular ML algorithm that is simple and easy to implement. It can be used for regression and classification; therefore, it can handle both numeric and categorical data. KNN assumes that similar things are in close proximity to each other. It produces output depending on the closeness of the input from the input that is used in the training phase. The algorithm works by finding the number of K points closest to the input data and voting for the best class output depending on the classes of the K points closest to it [31].

Equation (11) shows how the algorithm calculates the distance between known points (found during training the model) and unknown input (during testing). It is also known as the Euclidean distance, with d denoting the distance. X_1 and X_2 represent the two feature vectors used which include the n number of features used in the model, therefore n is equal to the number of features, which determines the number of dimensions of the model.

$$\begin{aligned} d &= \sqrt{(X_1 - X_2)^T(X_1 - X_2)} \\ X_1 &= (x_{11}, x_{12}, \ldots, x_{1n}), X_1 \in \mathbb{R}^n \\ X_2 &= (x_{21}, x_{22}, \ldots, x_{2n}), X_2 \in \mathbb{R}^n \end{aligned} \tag{11}$$

Advantages of KNN classifier are its versatility and ease of use. However, a disadvantage is that it starts to get extremely slow as the size of the dataset and the number of features increases, in terms of training time and testing time.

5.3. Support Vector Machine

SVM is a ML classifier that works by finding a line or a hyperplane that separates points of different classes. SVM algorithm aims to find the optimal line or hyperplane between the different classes [31].

SVM uses the closest points to the line or the hyperplane (called support vectors) and finds the distance (called the margin) between them and the line or the hyperplane. The distance measured is the Euclidean distance. The algorithm aims to maximize the margin to find the optimal line or hyperplane. Therefore, it can be expressed as an optimal margin classifiers problem. Consequently, this margin acts as a boundary between the classes, and depending on the input, the predicted class is given by checking which class area the point falls into.

For a binary classification problem, the objective function of the algorithm and its constraints are illustrated in Equation (12). In Equation (12), w represents the vector of weights and b denotes the

intercept. $y^{(i)}$ and $x^{(i)}$ are a training example, where $x^{(i)}$ is a vector of input features and $y^{(i)}$ is the corresponding output, which is the class. n denotes the number of training examples.

$$\begin{aligned} &min_{w,b} \quad \frac{1}{2}||w||^2 \\ &s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1, i = 1, \ldots, n \end{aligned} \tag{12}$$

If the data are not linearly separable, another, higher dimension is added, where the data become linearly separable. Then, using mathematical transformation, the decision boundary is plotted back into the original dimension [31].

5.4. Decision Tree

DT classifier produces a final tree with branches, which represent the observations, and leaves, which represent the final prediction. During the running of the algorithm, the dataset is divided into a number of subsets, which are then used to develop the DT [32].

DT is a famous classifier that can be used for both classification and regression. As a result, DT can be used with both numeric and categorical data. DT are used best when there are missing values in the dataset, which makes it a good choice for a variety of applications.

In Equation (13), X indicates the input vector of a known number of features, Y indicates the output, and S indicates the training set with the couples of input features and their output.

$$\begin{aligned} &X = (x_1, x_2, \ldots, x_n)^T, X \in \mathbb{R}^n \\ &Y \in \mathbb{R} \\ &S = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\} \end{aligned} \tag{13}$$

In the final tree, X represents the branches and Y represents the leaves, which are the decision nodes. Equation (14) represents the purpose of the model. The model is expected to predict the value of Y using the input X, by applying the function p, which is representative of the tree.

$$Y = p(X) \tag{14}$$

The function p is created by recursively branching nodes in the training phase to produce decision nodes that give the most accurate and pure nodes at the end. When applying the function, the inputs, X, are used to make a decision on which branches to pursue to finally get a prediction and output, Y.

5.5. Random Forest

RF is another ML classifier that can be used for classification and regression, so like DT, it can work with numeric and categorical data. RF algorithm is very similar to DT. The only difference is that RF creates numerous DTs when running the algorithm during training. Therefore, RF is a classifier consisting of a group of tree-structured classifiers. The output is based on the most popular class for the input [32].

When dealing with regression, the output prediction is the mean prediction of the individual trees. When dealing with classification, the output class is the mode of the classes of the individual trees. Thus, RF classifier is based on averaging or voting for the best class between all the different trees.

The main advantages of using RF is that it deals well with the issue of overfitting, it usually has very high predictive accuracy, and its usability for a multitude of different applications.

5.6. Deep Neural Networks

DNN is a ML technique that uses Deep Learning. Deep Learning is a ML method that allows the training of a model with different layers. The more layers used in a model, the deeper it

becomes. Multilayer perceptron is a type of DNN, which is a feedforward DNN and uses nonlinear activation functions.

The model is broken down into three types of layers: the input layer, hidden layers, and output layer. Each layer has a set of neurons that are connected with the neurons of the next layer. The input layer consists of the input data. The output layer consists of the output data or the class. The hidden layers are the layers that perform mathematical computations on the input [31,33].

Each connection between neurons has a weight that determines the importance of that connection. The weight specifies the impact of the input on the output of the next neuron and so on, until reaching the overall final output [31].

Assuming a model has a total of N layers, including the input, hidden, and output layers, Equations (15) and (16) reveal the mathematical connection between the layers of a DNN. $z^{(l+1)}$ is the output vector of the neurons at level l. h is the activation function and $a^{(l+1)}$ is its output vector, applied to $z^{(l+1)}$. Moreover, $a^{(l+1)}$ acts as the input vector to level $l+1$. As a result, $a^{(1)}$ denotes the starting input vector and $a^{(N)}$ denotes the final output vector. W^l and b^l represent the matrices of weights and bias at level l, respectively.

$$z^{(l+1)} = W^l a^{(l)} + b^l \tag{15}$$

$$a^{(l+1)} = h(z^{(l+1)}) \tag{16}$$

In addition, each neuron has an activation function. The activation function imitates the signal as it passes through to the next set of connected neurons. If the result of the activation function is greater than a threshold value, then the output is passed, and if the value is not greater than the threshold value, then the connection is ignored. This is an iterative process, so it is repeated until the final layer is reached and a class is chosen, which becomes the final output of the model [33].

The advantages of DNN is its flexibility to be used for many applications, as the same model can be used for many applications, and it usually has high prediction accuracy. The main disadvantage is that it needs large datasets to be able to perform well and it is very prone to over fitting as there is no direct theory on how to choose the number of layers and neurons in the model.

5.7. Recurrent Neural Networks

RNN are a type of neural networks, which are very densely connected. The main difference between RNN and regular feed forward neural networks are the initiation of time and the fact that the output is fed back into the neural network for a certain number of times.

Similar to other neural networks, RNN also have layers, which are the input layer, hidden layers and the output layer. The input is fed into the network and passes through the whole network and the output is reached. After that, the output is re-fed into the neural network. This allows the modeling of time and sequence dependent data [31].

The drawback of conventional RNN is the vanishing gradient problem. When used practically with long memories and more steps, the back propagation gradients start to accumulate or become zero. As a result, the model breaks down and accuracy becomes extremely low.

A more specific type of RNN is the long short-term memory (LSTM) networks. This type of neural network helps with the reduction of the vanishing gradient problem by reducing the number of times gradients that are less than zero are multiplied. As a result, RNN become a lot more useful for long term memory applications, such as predicting the total power usage of households in an area during the period of the whole year. This is done by making an internal memory state that is combined with the processed input. Thus, the effect of multiplying smaller gradients is significantly reduced. A forget gate is used to decide which states are to be remembered and which are to be forgotten. This controls the effects of preceding inputs and time dependence. Additionally, an input gate and an output gate are also utilized in LSTM cells [31]. Figure 4 illustrates the LSTM cell.

Regarding the input gate, the input is first given a value between −1 and 1 using a tanh activation function, shown in Equation (17). U_g represents the weight for the input and V_g represents the weight of the previous cell output. b denotes the input bias. x_t is the starting input of the network and h_{t-1} denotes the output of the neural network that is re-fed into the network. The subscript g represents the starting values of the input weights and the bias values.

$$g = tanh(b_g + x_t U_g + h_{t-1} V_g) \tag{17}$$

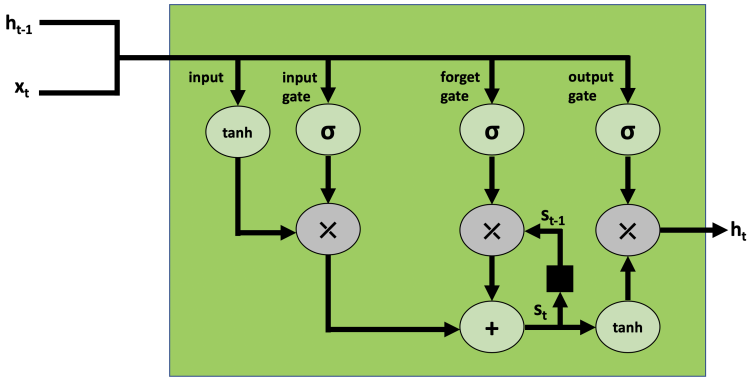

Figure 4. Long short-term memory (LSTM) cell diagram.

After that, the input is then multiplied element wise by the output of the input gate. The input gate is a hidden layer of sigmoid activated nodes, with x_t and h_{t-1} input weighted values. The output is a value between 0 and 1. This is multiplied element wise by the input to decide which inputs are switched on and off. Equation (18) shows the equation used by the input gate. The subscript i signifies that the values belong to the input gate [31].

$$i = \sigma(b_i + x_t U_i + h_{t-1} V_i) \tag{18}$$

Consequently, the output of the input gate can be expressed as the element wise multiplication of i and g, shown in Equation (19).

$$i \circ g \tag{19}$$

The forget gate is a hidden layer of sigmoid activated nodes, with x_t and h_{t-1} input weighted values, as illustrated in Equation (20). The subscript f signifies that the values belong to the forget gate [31].

$$f = \sigma(b_f + x_t U_f + h_{t-1} V_f) \tag{20}$$

The main function of the forget gate is that it acts as weights for the internal state. The output of the element wise multiplication of the previous internal state and the output gate is illustrated as $s_{t-1} \circ f$. It is advantageous as the final state is simply added to the input. This is not the case in conventional RNN where the final state is multiplied or put in a sigmoid activation function, which causes the vanishing gradient problem. The output s_t of the forget gate is expressed in Equation (21) [31].

$$s_t = s_{t-1} \circ f + i \circ g \tag{21}$$

Finally, the output gate is made from another tanh function and a sigmoid function. Like in other gates, the output of the sigmoid function is multiplied by the output of the tanh function. This is done to be able to decide which value of the state is finally output from the model. The output gate and the

final output can be illustrated in Equations (22) and (23), respectively. The subscript o signifies that the values belong to the output gate [31].

$$o = \sigma(b_o + x_t U_o + h_{t-1} V_o) \tag{22}$$

$$h_t = tanh(s_t) \circ o \tag{23}$$

6. Machine Learning for Coordinated Electric Vehicles Charging

The power data of different households is used to be able to calculate the total power usage in an area at different time intervals. This allows the model to predict the power of EVCS that will least harm the power grid. The power is decided depending on the total power usage in the area to avoid sudden power usage increases that cause power overloads in the power grid.

6.1. System Limitations

The system proposed in this paper uses ML, which has several limitations. The first limitation is that ML is stochastic. This means that ML algorithms run randomly. As a result, the accuracy of using such models fluctuates slightly every time they are run and cannot be predicted precisely, which could cause different and changing results. Consequently, the fluctuating accuracies must be closely monitored as it can be problematic if the change in accuracy becomes large.

In addition, another limitation of the system is the accuracy of the data used in training. The dataset used when training the model must be precise and large enough. However, the size of the dataset used in training must not be too large where the model will be overfitted and not general enough. Thus, the accuracy of the model will be too low as the model is not adapted to correctly identifying the class of unknown data inputs.

Another limitation is sensibility and practicality. The model created at the end must be practical. This means that the model should be created in a way to allow its use in real-life. Therefore, the model should be easily implementable and interpretable. As a result, having a model that is easy to understand and has the ability to be easily utilized in EVCS is imperative.

In terms of the power grid, the main constraint is that the overall power usage, while operating EVCS, should not be significantly higher than the normal peak power usage at all times. Thus, the charging of EVs must be done at times where the power usage is capable of handling the extra load brought from EVCS. In addition, techniques for peakload shifting can also be used to allow the use of EVCS at different times. As a result, the overall load curve would flatten and the power usage would be constant. This would keep the power quality high and avoid voltage fluctuations. Moreover, it would decrease the charging time of EVs because the power provided to the EVCS would be constant and power grid overloads would not occur.

6.2. Model Parameters

In the model provided in this paper, the dependent variable or the class label is the power rating (PR) of the EVCS, derived from the total power usage data in an area for residential households. The independent variables are the date and the time of day. The model does not require real-time data as input and only uses the time of charging and historical load data.

The parameters used in this model, which are the data groups in the dataset, are the following.

- Day: This is a numeric data type with a value between 1 and 31 inclusive depending on the day of the month. The values accepted are integers. It is an independent variable.
- Month: This is a numeric data type given a value between 1 and 12 inclusive depending on the month. The values accepted are integers. It is an independent variable.
- Hour: This is a numeric data type with a value between 0 and 23 inclusive as it is in a 24-h clock format. The values accepted are integers. It is an independent variable.

- Minute: This is a numeric data type with a value between 0 and 50 inclusive. The values used in the dataset are multiples of 10. It is an independent variable.
- PR: This is a categorical data type. It holds values high, normal, or low. This is representative of the classes in the model. It is the dependent variable.
- Total Power Usage: This is a numeric data type. It holds the value of the historical power usage by all 200 households in an area. It is the dependent variable in the LSTM model, which is the forecasted load value by the LSTM model.

6.3. Data Description

The power usage dataset [34] consists of 52,560 data records collected from a residential household area in the United States. As there is a lack of historical EVs charging data, the model uses the normal load data from a residential area. Each record has 201 attributes, which are the date and time and the power usage of 200 residential households. The data was collected over a period of a whole year with a time interval of 10 min. The total usage of all 200 households was calculated. A new dataset [35], also consisting of 52,560 data records, was derived from the original dataset. The new dataset has 5 attributes: day, month, hour, minute, and power. The power attribute was created depending on the calculated total power usage of all 200 residential households.

The attributes of the new dataset have the following specifications.

1. Data Cycle: Collected every 10 min
2. Data Type: Numeric and categorical
3. Data Format: Stored in a comma separated values (csv) file

During the data analysis, the following aspects were considered in the methodology.

1. How day and month affect the power usage of residential households?
2. How time (hour and minute) affect the power usage of residential households?
3. How total power usage affects PR of EVCS?

6.4. Model Construction

Python was used to construct the different models. The scikit-learn library was used to utilize the following classifiers; NB, SVM, KNN, DT, and RF. Additionally, the scikit-learn library was used to cross-validate results and produce confusion matrices for each model. KNN was given the default *K*-value of 5. To add to that, the gamma parameter for SVM was set to auto, which means it will be equal to the reciprocal of the number of features, and the default radial basis function kernel was used. Moreover, the n_estimators parameter, which specifies the number of trees, for RF was given a value of 100.

In addition, Python was also used to allow the construction of the DNN. The Keras library with TensorFlow backend was used in this case. The classes were encoded to be able to use them in the DNN as numeric data. High, low, and normal were given values 0, 1, and 2, respectively. The input layer, which used a relu activation function, had four features as inputs and 10 neurons. There were two hidden layers, which had relu activation functions. The first had 10 neurons and the second had 8 neurons. Finally, the output layer had three neurons, which are the three output classes, and utilized a softmax activation function. The batch size used was 16 and the epochs value was 300. The parameters of each algorithm have been set based on the testing with different values to fine-tune and maximize the performance of each algorithm.

Furthermore, Python was utilized to allow the construction of the LSTM. Keras library with TensorFlow backend was used in this case as well. In the LSTM model, the model was used as a regressive model. Thus, the model was used to estimate the total power usage first and then decide the PR of EVCS, depending on the value of the total power usage. The input layer had four features as inputs and 100 neurons. There were 2 hidden layers, which utilized both relu and sigmoid activation functions. The first had 60 neurons and the second had 20 neurons. The output layer had 1 neuron,

which is the output value, and used a sigmoid activation function. An RMSprop optimizer was used with a learning rate value of 0.0001. The batch size used was 16. In addition, the epochs value was 25. Finally, the time steps utilized was 15, which represents a time of 150 min in our dataset. The values of hyperparameters chosen provided the highest accuracies with low computation times.

After building the models and applying them, the models would be able to predict three classes (high, normal, and low) depending on the day, month, hour, and minute. As a result, regarding EVCS, if the PR is predicted to be high, then fast charging can be used. If the PR is normal, then conventional charging can be used. If the PR is low, then the EVCS can be used to support the grid or use slow charging.

6.5. Machine Learning Predictive Models

The training of the models was done using JupyterLab and Anaconda Distribution tool. Anaconda Distribution is an open source tool that makes the building and training of ML models using Python easy, as it has a simple GUI interface. It has all the libraries needed and makes it easier for large scale datasets and projects to be handled.

The tool was used to build the ML predictive models, train the models, and test the models. After the algorithms were created and used, cross-validation was utilized to check the value and worth of the models and their accuracy at predicting the most appropriate PR for EVCS. In addition, confusion matrices were produced for each model to be able to check the class recall percentage and the class precision percentage and to check for false positives and false negatives.

7. Results

Accuracy shows how well the model is able to correctly analyze and label unlabeled data. It is a proportion between the number of data that was labeled correctly and the total number of data provided. To quantify the accuracy of the models, a ten-fold cross-validation test was used. Figure 5 shows an overview comparison between the models using the cross-validation results, excluding LSTM. The results for NB were the lowest with a mean accuracy of 63% and a standard deviation of 0.0055. For KNN, a mean accuracy of 82% and a standard deviation of 0.0066 were obtained. In addition, SVM had a mean accuracy of 83% and a standard deviation of 0.0063. DNN had a mean accuracy result of 81% and a standard deviation of 0.0074. Furthermore, DT had a mean accuracy of 89% and a standard deviation of 0.0052. Finally, RF had the highest mean accuracy result of 90% and the smallest standard deviation at 0.0038.

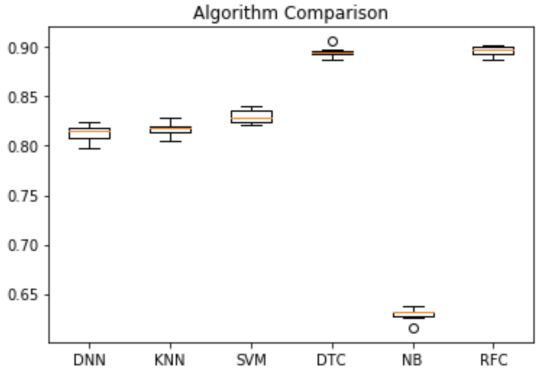

Figure 5. Algorithm comparison using cross-validation results.

Tables 1–9 show the confusion matrices for all the ML models. The confusion matrices are used as a way to visualize the accuracy of each ML classifier and to examine their reliability. The matrices include the class precision (p_c) and class recall (r_c) percentage for each class label, which are calculated as seen in Equations (23) and (24). Each value in the tables is color coded to signify the performance of

the ML classifier for the different classes. Red is used for low performance, orange is used for medium performance and green is used for high performance.

$$p_c = \frac{real_c}{t_p c} \cdot 100 \tag{24}$$

$$r_c = \frac{real_c}{t_t c} \cdot 100 \tag{25}$$

7.1. Classification Accuracy

NB had the worst results. As shown in Table 1, it was not able to predict low for any of the data entries. The high class recall was 79% and the normal class recall was 62%. The overall mean accuracy for all the classes was 63% only.

Table 1. Confusion matrix for NB.

	Predicted High	**Predicted Low**	**Predicted Normal**	**Class Recall**
true High	4742	0	1277	79%
true Low	401	0	967	0%
true Normal	1195	0	1930	62%
class precision	75%	0%	46%	

Regarding KNN, the class recall for high, low, and normal were 92%, 70%, and 71%, respectively. The overall mean accuracy was 83%. These results are illustrated in Table 2.

Table 2. Confusion matrix for KNN.

	Predicted High	**Predicted Low**	**Predicted Normal**	**Class Recall**
true High	5562	53	404	92%
true Low	117	964	287	70%
true Normal	629	267	2229	71%
class precision	88%	75%	76%	

The results for SVM can be seen in Table 3. The recall for the high class label was 92%. For the low class label, the class recall was 68%. For the normal class label, the class recall was 75%. The mean accuracy was 84%.

Table 3. Confusion matrix for SVM.

	Predicted High	**Predicted Low**	**Predicted Normal**	**Class Recall**
true High	5523	78	418	92%
true Low	110	926	332	68%
true Normal	543	235	2347	75%
class precision	89%	75%	76%	

As we move into the results for DT shown in Table 4, the values start to increase and produce better results. The class recall for high, low, and normal were 94%, 87%, and 83%, respectively. The mean precision of all classes was 90%.

Table 4. Confusion matrix for DT.

	Predicted High	Predicted Low	Predicted Normal	Class Recall
true High	5685	12	322	94%
true Low	5	1186	177	87%
true Normal	320	201	2604	83%
class precision	95%	85%	84%	

As seen in Table 5, when using RF, the class recall for high was 95%, for low was 86%, and for normal was 84%. The value for the mean accuracy was 90%. It can be deduced that RF performs better than the aforementioned ML classifiers, being closely on par with DT. However, despite the same mean accuracy, RF has a higher overall class recall for all the classes.

Table 5. Confusion matrix for RF.

	Predicted High	Predicted Low	Predicted Normal	Class Recall
true High	5698	18	303	95%
true Low	14	1180	174	86%
true Normal	339	167	2619	84%
class precision	94%	86%	85%	

A multilabel confusion matrix is used for DNN. Tables 6–8 illustrate the confusion matrices for each class.

In terms of the high class label, the rate of true negatives was 88% and the rate of true positives was 89%. Additionally, the accuracy for the high class label was 89%.

Table 6. Confusion matrix for DNN for High class.

	Predicted No	Predicted Yes	Class Recall
true No	6328	1110	88%
true Yes	878	9029	89%
class precision	86%	91%	

For the low class label, the percentage of correctly predicting not having a low PR was 96% and the percentage of correctly predicting having a low PR was 68%. Moreover, the accuracy for the low class label was 92%.

Table 7. Confusion matrix for DNN for Low class.

	Predicted No	Predicted Yes	Class Recall
true No	14,181	844	96%
true Yes	561	1759	68%
class precision	94%	76%	

Regarding the normal PR class, the percentage of true negatives and true positives were 87% and 76%, respectively. The value of the accuracy for predicting the normal class label was 84%.

Table 8. Confusion matrix for DNN for Normal class.

	Predicted No	Predicted Yes	Class Recall
true No	11,100	1127	87%
true Yes	1642	3476	76%
class precision	91%	68%	

The LSTM regression model had a mean squared error of 0.00087. This shows that the accuracy of the LSTM model is extremely high and shows potential for providing the best results for the classification problem of choosing the most appropriate PR for EVCS. Figure 6 shows the LSTM regression results over the time period of a day. It is seen that the real and predicted values are significantly close and the model is able to provide accurate results following the trend shown by the real values.

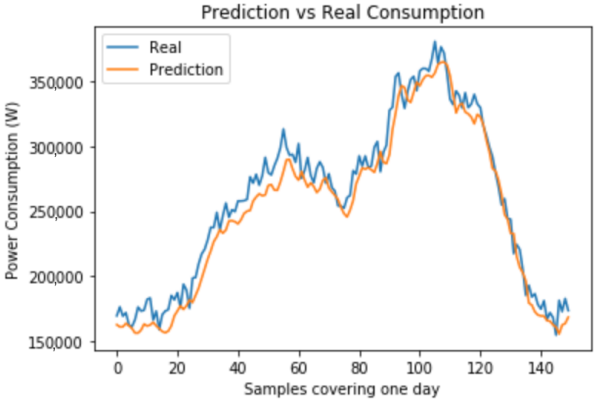

Figure 6. LSTM regression results over a day.

Table 9 shows the confusion matrix for the LSTM model. The class recalls were 96%, 94%, and 95% for high, low, and normal, respectively. The value for the mean accuracy was 95%. Thus, it can be deduced that LSTM performs better than all the other ML classifiers used.

Table 9. Confusion matrix for LSTM.

	Predicted High	Predicted Low	Predicted Normal	Class Recall
true High	1963	0	90	97%
true Low	0	896	54	96%
true Normal	63	40	2142	94%
class precision	96%	94%	95%	

7.2. Comparison Between ML Models

Table 10 represents a summarized comparison of all the aforementioned results between all the ML classifiers used in terms of their accuracy. As seen previously, LSTM provides the best accuracy for predicting the PR of EVCS correctly at a value of 95.3%. After that, RF provides good accuracy results at 90.3%. This is closely followed by DT, which has an accuracy of 90.1%. DNN comes next with an accuracy of 88.2%, which is followed by SVM with 83.7% accuracy. After that, KNN comes next with an accuracy of 83.3%. Last, with the lowest accuracy, comes NB, at an accuracy of 63.5%.

Table 10. Accuracy comparison between different ML models.

Model	Accuracy	Error Rate
Naïve Bayes (NB)	63.5%	±63.5%
K-Nearest Neighbors (KNN)	83.3%	±8.3%
Support Vector Machine (SVM)	83.7%	±8.7%
Decision Tree (DT)	90.1%	±6.1%
Random Forest (RF)	90.3%	±5.3%
Deep Neural Network (DNN)	88.2%	±20.2%
Long Short Term Memory (LSTM)	95.3%	±0.71%

These results show that NB is an extremely bad model that performs weakly for predicting the best PR of EVCS that avoids dangers to the power grid. The results show that the best model to use for this application is LSTM.

In addition, LSTM is also the model with the smallest error rate, which is ±0.71%. This is followed by RF (±5.3%), DT (±6.1%), KNN (±8.3%), SVM (±8.7%), DNN (±20.2%), and NB (±63.5%).

Overall, the best model that portrays the most promising results is LSTM. It has both the highest accuracy and lowest error rate. Thus, based on the approach followed in this paper, LSTM is the most noteworthy method for analyzing the total power usage of an area and predicting the most suitable PR for EVCS.

The high accuracy obtained by the LSTM model is due to its ability to detect time series and time-dependent patterns, which are inherent in the used dataset. Additionally, LSTM would provide high accuracy results for most distribution systems as the general load curve pattern is similar for most distribution systems. In addition, due to the multiclass classification, some algorithms, such as NB, SVM, and KNN, lose their high predictive accuracy as they are based on probability and function graphing. Therefore, the increased number of classes decreases their effectiveness as classifiers. Surprisingly, DNN did not provide high predictive accuracy, despite the fine-tuning of its parameters and structure. A possible explanation can be the size of the dataset, which might not be large enough for the training of the DNN.

8. Effect of Coordinated Electric Vehicles Charging on the Grid

The LSTM model was utilized to examine the effect of the coordinated charging of EVs on the distribution network. Figure 7a shows the total power of households shown in Figure 2 (typically 200 households) in a day. The employed data for the household load and EVCS loads (conventional and fast charging) can be found in [35].

Figure 7b shows a comparison for different cases of load curves in a day. The first case is the regular daily load curve. The second is the load curve while using EVCS at conventional charging with no coordination. Next, the flattened load curve for using coordinated charging in EVCS using the LSTM model; the transformer maximum supported power level is also shown. As seen in Figure 7, the use of EVCS without coordination, with conventional charging, produces a similar load curve but with higher values. Consequently, the use of fast charging, without coordination, is expected to cause more overload as the power usage will be even further increased. These problems include power quality decrease and voltage fluctuations.

Figure 7b shows the load curve for the coordinated operation of EVCS. Moreover, it also shows the different modes used in the EVCS throughout the day. At the start of the day, at midnight, fast charging is used. Towards 8 am, the mode is changed to conventional charging. During the peakload time, the EVCS is used to support the grid to shave the peakload to guarantee that the power usage does not increase over the maximum supported power level. During this time, the process of discharging is used and the EV is used to provide vehicle-to-grid power. This helps the protection of the transformer and increases its lifetime. After that, conventional charging is used as the power of the households starts to decrease, then the mode of operation for the EVCS is changed back to fast charging.

The flattened load curve from the coordinated charging of EVs using LSTM is similar to the load curves provided by other optimization techniques such as quadratic programming [36]. However, the use of ML provides more simplicity and faster speed, due to the decreased computational power needed. Thus, the LSTM model can be used as an online system through a cloud without technical complications, which makes it an effective solution that can be implemented in an IoT-based infrastructure.

Moreover, the LSTM model relies on the time-dependent power usage throughout the day for residential loads. Therefore, it can be deduced that the number of EVs or EVCS loads do not affect the accuracy of the model, which means the model has the ability to be scaled up for the use of more EVCS and higher EVs loads. This is due to the LSTM model's independence of the EVs loads.

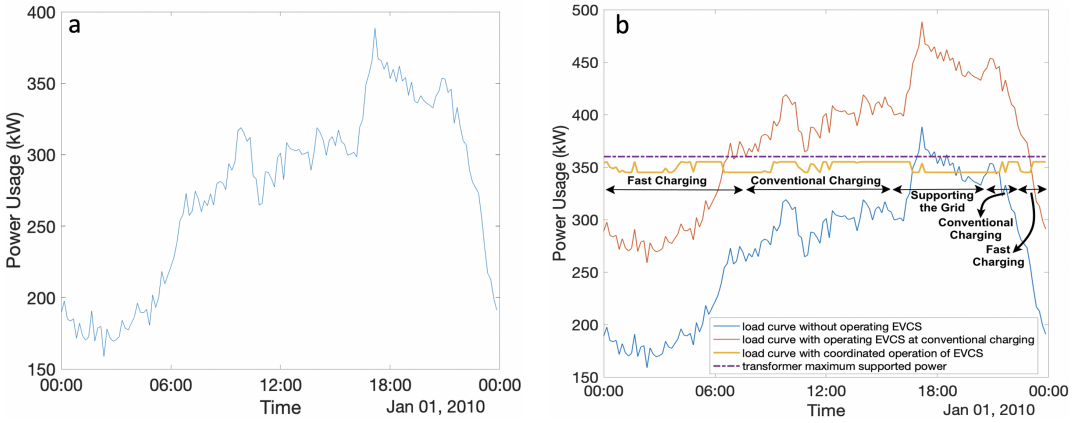

Figure 7. (**a**) Normal load curve over a day. (**b**) Comparison of different load curves over a day.

Figure 8 shows the performance of the LSTM model after the introduction of Gaussian white noise (GWN), to simulate the model under random and abrupt scenarios and the partially random nature of the residential loads.

As seen in Figure 8, the uncertainty does not affect the performance of the model. The model is still able to predict the most appropriate PR for the EVCS accurately and the load curve for the coordinated charging of EVs is still below the transformer maximum supported power level. Moreover, the curve in Figure 8 is nearly identical to the coordinated charging curve in Figure 7b, where there was no uncertainty introduced. This conveys that despite the stochastic nature of ML, ML nevertheless provides an effective and powerful solution for coordinating the charging of EVs and is minimally affected by uncertainty and noise.

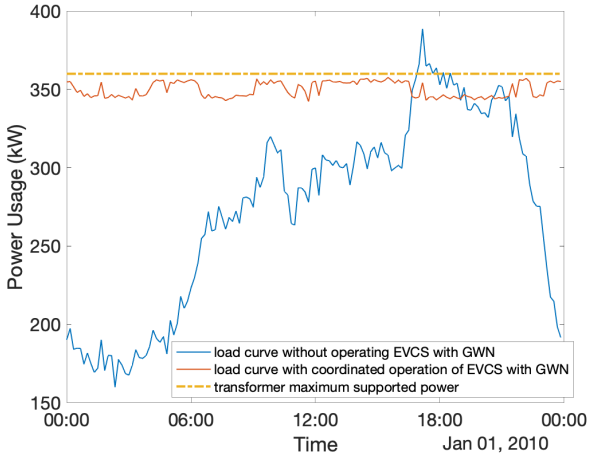

Figure 8. Comparison of different load curves over a day with GWN.

9. Conclusions

In conclusion, this paper provides a method for choosing the best mode for charging in EVCS that causes the least harm to the power grid. It can solve many issues faced from implementing EVCS, such as power overloads and power outages. Therefore, the power grid is protected from these sudden power surges that are caused by the operation of EVCS. Consequently, the distribution grid and its components such as transformers are protected, which increases their life cycle. In addition, this leads to slower deterioration of the distribution grid, which decreases the costs of maintenance.

The model produced for predicting the PR of EVCS is new and has not been previously studied. Seven different ML classifiers (NB, KNN, SVM, DT, RF, DNN, and LSTM) were used in order to provide the best possible results and choose the best model. The leading ML model that provided the best results, which were significant, is LSTM. The presented model provides an effective EVs charging coordination method, which requires little computational power. Thus, the system can be implemented online as a cloud system with no complications.

The model is based on the power usage data of 200 residential households. It is able to choose the best method of operation for EVCS, which are fast charging, conventional charging, and supporting the grid or slow charging. As mentioned before, LSTM was the best model with an accuracy of 95.3% and an error rate of $\pm$0.71%, which provides the best results for predicting the most suitable PR for EVCS.

10. Future Work

Future work can encompass working with price rates of electricity, where the price of electricity in EVCS fluctuates and will be determined depending on the pressure on the power grid and the total power usage in the area.

Another aspect that can be further researched is seasonality in the use of electricity. This means that the models can be further improved as the households' power usage changes depending on the seasons.

Moreover, using IoT architecture and automation for the operation of EVCS can be further studied. As a result, the charging and discharging processes of EVs can be done automatically depending on the power usage. Thus, this will require no user interaction and the EVCS can be powered only when the power grid can accommodate the power it needs to provide for the EVCS; it would also allow the shifting and shaving of the load profile.

Finally, the model can be created to be able to get real-time data by using real-time power usage data. This means that the model will be able to adapt itself to the changes that might occur in the power usage daily.

Author Contributions: Conceptualization, M.S., L.I. and A.M.; methodology, M.S., L.I. and A.M.; writing–original draft preparation, M.S.; writing–review and editing, L.I. and A.M.; project administration, L.I. and A.M.; funding acquisition, A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Qatar National Research Fund grant number NPRP (10-0130-170286).

Acknowledgments: This publication was made possible by NPRP grant NPRP (10-0130-170286) from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. IEA. *Global Electric Vehicles Outlook 2019*; International Energy Agency: Paris, France, 2019.
2. Maglaras, L.A.; Jiang, J.; Maglaras, A.; Topalis, F.V.; Moschoyiannis, S. Dynamic wireless charging of electric vehicles on the move with Mobile Energy Disseminators. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)* **2015**, *6*, 239–251.
3. Mario, A.R.; Fadi, A.A.; Gagnaire, M.; Lascaux, Y. Telewatt: An innovative electric vehicle charging infrastructure over public lighting systems. In Proceedings of the 2nd International Conference on Connected Vehicles and Expo (ICCVE), Las Vegas, NV, USA, 2–6 December 2013.

4. Khan, A.; Memon, S.; Sattar, T. Analyzing Integrated Renewable Energy and Smart-Grid Systems to Improve Voltage Quality and Harmonic Distortion Losses at Electric-Vehicle Charging Stations. *IEEE Access* **2018**, *6*, 26404–26415. [CrossRef]
5. Liu, C.; Wang, J.; Botterud, A. Assessment of Impacts of PHEV Charging Patterns on Wind-Thermal Scheduling by Stochastic Unit Commitment. *IEEE Trans. Smart Grid* **2012**, *3*, 675–683. [CrossRef]
6. Badawy, M.; Sozer, Y. Power Flow Management of a Grid Tied PV-Battery System for Electric Vehicles Charging. *IEEE Trans. Ind. Appl.* **2017**, *53*, 1347–1357. [CrossRef]
7. Mohsenzadeh, Y.; Sheikhzadeh, H.; Reza, A.; Bathaee, N.; Kalayeh, M. The Relevance Sample-Feature Machine: A Sparse Bayesian Learning Approach to Joint Feature-Sample Selection. *IEEE Trans. Cybern.* **2013**, *43*, 2241–2254. [CrossRef]
8. Li, Z.; Oechtering, T. Privacy-Aware Distributed Bayesian Detection. *IEEE J. Sel. Top. Signal Process.* **2015**, *9*, 1345–1357.
9. Viegas, J.; Vieira, S.; Melicio, R.; Matos, H.; Sousa, J. Prediction of events in the smart grid: Interruptions in distribution transformers. In Proceedings of the 2016 IEEE International Power Electronics and Motion Control Conference (PEMC), Varna, Bulgaria, 25–30 September 2016; pp. 436–441.
10. Xu, L.; Xu, J. Integrated System Health Management-Based Progressive Diagnosis for Space Avionics. *IEEE Trans. Aerosp. Electron. Syst.* **2014**, *50*, 1390–1402. [CrossRef]
11. Ali, S.; Wu, K.; Weston, K.; Marinakis, D. A Machine Learning Approach to Meter Placement for Power Quality Estimation in Smart Grid. *IEEE Trans. Smart Grid* **2016**, *7*, 1552–1561. [CrossRef]
12. Yu, M.; Xia, H.; He, Y.; Wang, H.; Jiang, C.; Chen, S.; Li, M.; Xu, J. Scheduled Health Monitoring of Hybrid Systems with Multiple Distinct Faults. *IEEE Trans. Ind. Electron.* **2017**, *64*, 1517–1528. [CrossRef]
13. Siryani, J.; Tanju, B.; Eveleigh, T. A Machine Learning Decision-Support System Improves the Internet of Things' Smart Meter Operations. *IEEE Internet Things J.* **2017**, *4*, 1056–1066. [CrossRef]
14. Ramona, M.; Richard, G.; David, B. Multiclass Feature Selection With Kernel Gram-Matrix-Based Criteria. *IEEE Trans. Neural Netw. Learn. Syst.* **2012**, *23*, 1611–1623. [CrossRef] [PubMed]
15. Deligiannis, P.; Koutroubinas, S.; Koronias, G. Predicting Energy Consumption Through Machine Learning Using a Smart-Metering Architecture. *IEEE Potentials* **2019**, *38*, 29–34. [CrossRef]
16. Clement-Nyns, K.; Haesen, E.; Driesen, J. The Impact of Charging Plug-In Hybrid Electric Vehicles on a Residential Distribution Grid. *IEEE Trans. Power Syst.* **2010**, *25*, 371–380. [CrossRef]
17. Deilami, S.; Masoum, A.; Moses, P.; Masoum, M. Real-Time Coordination of Plug-In Electric Vehicle Charging in Smart Grids to Minimize Power Losses and Improve Voltage Profile. *IEEE Trans. Smart Grid* **2011**, *2*, 456–467. [CrossRef]
18. Wang, Z.; Fan, S.; Liu, B.; Liu, X.; Wei, Z. Coordinated Charging Strategy of Plug-In Electric Vehicles for Maximising the Distributed Energy Based on Time and Location. *J. Eng.* **2017**, *13*, 1740–1744. [CrossRef]
19. Tang, W.; Bi, S.; Zhang, Y. Online Coordinated Charging Decision Algorithm for Electric Vehicles without Future Information. *IEEE Trans. Smart Grid* **2014**, *5*, 2810–2824. [CrossRef]
20. Sun, S.; Dong, M.; Liang, B. Real-Time Power Balancing in Electric Grids With Distributed Storage. *IEEE J. Sel. Top. Signal Process.* **2014**, *8*, 1167–1181. [CrossRef]
21. Li, J.; Li, C.; Xu, Y.; Dong, Z.; Wong, K.; Huang, T. Noncooperative Game-Based Distributed Charging Control for Plug-In Electric Vehicles in Distribution Networks. *IEEE Trans. Ind. Inform.* **2018**, 14, 301–310. [CrossRef]
22. Rawat, T.; Niazi, K. Comparison of EV smart charging strategies from multiple stakeholders' perception. *J. Eng.* **2017**, *2017*, 1356–1361. [CrossRef]
23. Leou, R. Optimal charging/discharging control for electric vehicles considering power system constraints and operation costs. *IEEE Trans. Power Syst.* **2016**, *31*, 1854–1860. [CrossRef]
24. Jian, L.; Xue, H.; Xu, G.; Zhu, X.; Zhao, D.; Shao, Z. Regulated charging of plug-in hybrid electric vehicles for minimizing load variance in household smart microgrid. *IEEE Trans. Ind. Electron.* **2013**, *60*, 3218–3226. [CrossRef]
25. Yao, W.; Zhao, J.; Wen, F.; Xue, Y.; Ledwich, G. A hierarchical decomposition approach for coordinated dispatch of plug-in electric vehicles. *IEEE Trans. Power Syst.* **2013**, *28*, 2768–2778. [CrossRef]
26. Shaaban, M.; Ismail, M.; El-Saadany, E.; Zhuang, W. Real-time PEV charging/discharging coordination in smart distribution systems. *IEEE Trans. Smart Grid* **2014**, *5*, 1797–1807. [CrossRef]

27. Erdogan, N.; Erden, F.; Kisacikoglu, M. A fast and efficient coordinated vehicle-to-grid discharging control scheme for peak shaving in power distribution system. *J. Mod. Power Syst. Clean Energy* **2018**, *6*, 555–566. [CrossRef]
28. Hong, S.; Lee, S.G.; Kim, M. Assessment and Mitigation of Electric Vehicle Charging Demand Impact to Transformer Aging for an Apartment Complex. *Energies* **2020**, *13*, 2571. [CrossRef]
29. Sun, W.; Yuan, Y. *Optimization Theory and Methods*; Springer Science + Business Media, LLC: New York, NY, USA, 2006.
30. Russel, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson India Education Services Pvt. Ltd.: Noida, India, 2018.
31. Géron, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*; O'Reilly Media: California, CA, USA, 2017.
32. Ho, T. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 832–844.
33. Nielsen, M. *Neural Networks and Deep Learning*; Determination Press: California, CA, USA, 2019.
34. Muratori, M. Impact of Uncoordinated Plug-In Electric Vehicle Charging on Residential Power Demand. *Nat. Energy* **2018**, *3*, 193–201. [CrossRef]
35. Shibl, M. Bus Load Data 2020. Available online: Https://www.kaggle.com/mostafashibl/electric-vehicles-charging-stations-power-rating (accessed on 29 December 2019).
36. Zhang, P.; Qian, K.; Zhou, C.; Stewart, B.; Hepburn, D. A Methodology for Optimization of Power Systems Demand Due to Electric Vehicle Charging Load. *IEEE Trans. Power Syst.* **2012**, *27*, 1628–1636. [CrossRef]