EVCharging-GPT: Predicting Electric Vehicle User Charging Behavior Using Large Language Models

Houzhi Li[1], Jinyu Wang[1†], Zixin Jiang[2], Guorui Su[2], Chaowen Yan[2], Hao Su[2], Zhichun Wang[2†]

[1]State Grid Smart Internet of Vehicles Co., LTD., Beijing 100031, China
[2]School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China

Keywords: Large language models; Electric vehicle charging; Charging behavior prediction; Sequence prediction; Multi-task prediction

Citation: Li H.Z., Wang J.Y., Jiang Z.X., et al.: EVCharging-GPT: Predicting Electric Vehicle User Charging Behavior Using Large Language Models. Data Intelligence, Vol. 7, Art. No.: 2025r12, pp. 336–357, 2025. DOI: https://doi.org/10.3724/2096-7004.di.2025.0002

ABSTRACT

With the rapid growth of the electric vehicle (EV) market, accurately predicting user charging behavior has become particularly important. This not only guides power distribution and charging station planning but is also crucial for improving user satisfaction and operational efficiency. This study aims to predict the charging behavior of EV users using Large Language Models (LLMs). Unlike traditional methods such as Long Short-Term Memory (LSTM) and XGBoost, or single-task prediction models, our proposed model, EVCharging-GPT, is the first to integrate the text generation capabilities of LLMs with a multi-task learning framework for EV user behavior prediction. We construct an EV user charging data processing flow and create a dataset of real scenarios for fine-tuning and testing the model. By carefully designing prompt templates, we transform the charging behavior prediction task into a text-to-text format, allowing the model to leverage its rich pre-trained knowledge base to make effective predictions. Additionally, we integrate temporal and static categorical features through natural language prompts and employ LoRA (Low-Rank Adaptation) technology to achieve efficient domain adaptation. To verify the effectiveness of the EVCharging-GPT model, we conduct extensive comparative experiments with various LLMs and traditional models. The results demonstrate the potential of the LLM-based approach for predicting user behavior in EVs and provide a solid foundation for future research and applications.

† Corresponding authors: Jinyu Wang (E-mail: wangjinyu@evs.sgcc.com.cn, ORCID: 0009-0005-4631-2987); Zhichun Wang (E-mail: zcwang@bnu.edu.cn, ORCID: 0000-0001-8797-7065).

© 2025 Chinese Academy of Sciences. Published under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

1. INTRODUCTION

With an increasing global focus on sustainable energy and the reduction of carbon emissions, electric vehicles (EV), as a representative of green transportation, are rapidly growing their market share. However, the popularization of electric vehicles not only depends on the advancement of vehicle technology but also faces multiple challenges such as charging infrastructure, charging behavior patterns, and economic incentives. Especially in the aspects of charging station layout, charging time selection, charging demand prediction, and discount and marketing strategy formulation, accurately predicting the charging behavior of electric vehicle users has become the key to improve the efficiency of the charging network, optimize the user experience, and promote the further popularization of electric vehicles. In order to better understand and manage this complex system, it is particularly important to construct a model that can accurately predict the charging behavior of EV users.

Predicting electric vehicle user charging behavior is crucial for charging station operators, grid managers, and even automobile manufacturers. It can not only help optimize the layout of charging stations, reduce queueing waiting time, improve the efficiency of charging piles, but also provide data support for the formulation of dynamic electricity price and discount strategy, and then guide users to rationally plan charging time, balance the load of the grid, and achieve a win-win situation of economic and environmental benefits. In addition, accurate prediction is also the basis for implementing effective marketing strategies, which helps to improve user satisfaction and loyalty.

However, current work in the EV field has largely focused on power distribution and grid stability, with limited emphasis on user behavior prediction. Among the existing works, our work is the first to model the user charging behavior prediction problem as a text-to-text sequence prediction task based on the data characteristics of EV charging behavior. This approach involves processing a series of chronologically ordered data points to predict future charging events. However, several challenges arise in this process:

(1) Multi-task prediction: predicting the charging time, charging amount, charging location, and applicable discount strategy at the same time requires the ability of the model to handle multivariate time series.

(2) External knowledge fusion: There are many factors that affect charging behavior, including but not limited to weather conditions, driving habits, and electricity pricing policies. How to effectively integrate these domain knowledge into the machine learning model to improve the prediction accuracy is a difficult problem to be solved.

(3) Interpretability requirements: the prediction model not only needs to be accurate, but also needs its decision-making process to be transparent and interpretable, so that decision makers can understand the reasons behind the prediction results and make more reasonable strategy adjustments.

To tackle these challenges, we propose EVCharging-GPT, an innovative EV user charging behavior prediction model based on the Transformer architecture. EVCharging-GPT leverages the text-to-text paradigm of large language models (LLMs) to handle multivariate time series effectively, enabling simultaneous

prediction of charging time, amount, location, and discount strategies while capturing complex dependencies. It integrates prompt templates to incorporate external knowledge such as pricing policies, enhancing prediction accuracy, and employs LoRA fine-tuning to improve model efficiency and interpretability. Our contribution is detailed as follows:

(1) We collect and construct an electric vehicle user charging record dataset, which covers user charging related activities, for fine-tuning and performance testing of large-scale language models, significantly improving the applicability and accuracy of the model in a specific domain.

(2) We propose EVCharging-GPT, a model to predict the charging behavior of EV users, which is specifically fine-tuned to the charging behavior of EV users and can effectively capture the key factors that affect user decision-making.

(3) Experimental studies under the framework of single-task and multi-task learning show that the EVCharging-GPT model has good flexibility and robustness in different scenarios. More importantly, it proves that it has better prediction ability than existing methods.

2. RELATED WORK

2.1 Large Language Models

In recent years, LLMs have made significant strides in natural language processing. Models like ChatGLM4 [1], GPT-4 [2], and Llama3 [3] have demonstrated remarkable capabilities in learning and understanding large-scale text data through deep learning algorithms. They have shown strong performance in various tasks such as text generation, sentiment analysis, and question answering. Tay et al. [4] and Ouyang et al. [5] highlighted that LLMs can handle diverse tasks using specifically designed prompts without training or fine-tuning, advancing the field of NLP through prompt engineering. Yao et al. [6] further improved LLM performance on link prediction tasks through fine-tuning, proving that fine-tuning is a valuable method for downstream tasks, achieving excellent results without expensive training from scratch [7-10]. The fine-tuning methods for LLMs are also evolving. Techniques like Low-Rank Adaptation (LoRA) [11] introduce trainable low-rank decomposition matrices into each layer of the Transformer architecture. This reduces the number of trainable parameters needed for downstream tasks while maintaining model quality. LoRA lowers training costs, improves training throughput, and does not add inference latency, making it effective across various tasks. Additionally, LLMs have made breakthroughs in knowledge graph tasks, including link prediction. Using a text-to-text training paradigm, these models can perform generative multi-hop link prediction. Research efforts like GraphEdit [12], MuseGraph [13], and InstructGraph [14] have shown that LLMs can effectively process structured data through natural language. Their extensive training on large datasets enables them to generalize to unseen tasks or prompts, showcasing their Chain of Thought (CoT) reasoning capabilities.

2.2 Large Language Models for Time series

The advancement of LLMs has brought new opportunities to the field of time series analysis, so many researchers have begun to use these LLMs to enhance the ability of time series modeling. However, LLMs do face a number of challenges in time series analysis. Firstly, the massive volume and diversity of time series data, coupled with the concept drift caused by its non-stationarity, hinder the continuous adaptation and retraining of the model. Nonetheless, recent studies have shown that the use of LLMS for time series analysis is still promising and many research opportunities exist. These include developing new tokenization methods that can better capture temporal dynamics, designing better prompts to improve model performance, enhancing the interpretability of LLM-based models, and exploring multimodal properties in time series data.

For example, Time-LLM [15] and TEST [16] reformat the input time series data, program it into text prototypes that are more convenient for language models to understand, and utilize text cue prefixes to assist LLMs to perform time series prediction. In addition, GPT4TS [17] and UniTime [18] fine-tune LLMs by freezing specific parts to provide effective initial feature representations for time series analysis tasks to improve performance. TEMPO [19] is specifically designed for non-stationary time series data, and achieves superior performance on multiple benchmark datasets by combining time series decomposition and prompt selection strategies. Llm4ts [20] uses a two-stage fine-tuning method. Firstly, the model is aligned with the characteristics of time series through Supervised fine-tuning (SFT), and then the model is further fine-tuned in the direction of downstream prediction tasks, so that the model can better adapt to different types of data and different downstream tasks without damaging the inherent characteristics of LLM. In addition, a novel multimodal time series anomaly detection method based on graph attention network (MST-GAT [21]) is proposed, which integrates the reconstruction module of the variational autoencoder and the prediction module of the multilayer perceptron to optimize the model performance.

While Time-LLM and similar studies convert time-series data into text, they focus on single-variable prediction and do not address the integration of domain-specific knowledge. In contrast, our EVCharging-GPT is designed to handle the heterogeneous characteristics of charging behavior, such as the coupling of electricity pricing strategies and user preferences. By designing prompt structures that encode both static attributes (e.g., Discount Type) and dynamic sequences (e.g., Charging Intervals), EVCharging-GPT enhances the effectiveness of multi-task learning.

2.3 Large Language Models for Power Systems

The application of LLMs in the field of power systems has indeed revolutionized, and their capabilities have even surpassed human levels in some aspects, providing rich opportunities for multiple applications in power systems. These applications include, but are not limited to, key areas such as power system modeling, grid stability analysis as well as consumer behavior analysis. Huang et al. [22] pointed out in their study that large foundational models, such as GPT-4, show great potential in power system modeling and operation, which can significantly improve the efficiency and reliability of power system

operation processes. Through deep learning and advanced algorithms, these models effectively capture the complexity and dynamic changes of the power system to optimize the allocation and use of power resources. However, it has also been shown that even advanced models such as GPT-4 cannot generate adequate results in zero-shot text generation tasks due to the scarcity of public power system data [23]. This shows that despite the powerful potential of LLMs, their performance is largely limited by the quality and quantity of available data.

In order to solve this problem, the latest power system foundation model PowerPM [24] improves the representation ability of power system time series data through self-supervised pre-training and contrastive learning, and achieves excellent performance in several downstream tasks. The PowerPM model utilizes a self-supervised pre-training framework, including masked power time series modeling and two-view contrastive learning, to capture temporal dependencies in power time series data and be aware of the differences between different time series Windows. The experiments involve five real-world power time series scenario datasets, including private and public data. By pre-training on large-scale power time series data, PowerPM achieves state-of-the-art performance on a variety of downstream tasks. PowerPM maintains its superiority when transferred to public datasets, demonstrating its remarkable generalization ability across different tasks and domains.

At the same time, LLMs have also begun to be applied to the field of electric vehicles. As an important part of new energy vehicles, the research on electric vehicle charging behavior is of great significance for grid load management, charging facility planning and user experience optimization. Using LLMs, researchers can analyze users' charging habits, preferences, and relationships with environmental factors to more accurately predict future charging demand. In related research, some scholars have explored the use of machine learning methods to predict the charging behavior of evs, but most studies focus on traditional statistical methods and shallow neural networks. In contrast, LLMs are expected to provide more accurate prediction results due to their powerful representation ability and ability to capture complex patterns. In addition, the LLM is also able to understand and generate charging events described by natural language, which makes it possible to extract charging behavior information from unstructured data sources such as social media and news reports. For example, the recent ChatEV [25] transforms the EV charging demand prediction task into a text-to-text format through personalized prompt templates and retrieval enhanced generation technique RAG, leveraging semantic knowledge from the pre-trained corpus of LLMs. At the same time, zero-shot augmentation instructions and model-independent meta-learning method Reptile are used to fine-tune the Sentence-T5 model, integrate domain knowledge, and effectively adapt the model to the task of charging demand prediction, improve the prediction accuracy and model adaptability.

Although the existing research such as ChatEV transforms the charging demand prediction into a text task, it relies on RAG and fixed templates, and cannot dynamically integrate multi-source time series data. In contrast, our EVCharging-GPT supports real-time behavioral sequence modeling by building context-aware cue templates through sliding Windows. In addition, traditional methods (such as LSTM [26]) are limited by single-task prediction and static feature processing, while the LoRA fine-tuning strategy in this

study can optimize classification and regression tasks at the same time, taking into account both efficiency and accuracy.

3. TASK DEFINITION

User charging behavior refers to the charging activities that users perform at specific charging stations for electric vehicles.An user charging behavior includes six attributes such as charging interval, time slot, price level, charging station, charging energy and discount type. Therefore, we can define it as a six-tuple consisting of these six attributes:

$$a = (i,\ t,\ p,\ s,\ c,\ d) \tag{1}$$

where i represents the interval between this charging session and the last one for the user, t represents The time slot of this charging session, p represents the price level for the user's charging payment, s represents the charging station used by the user, c represents the amount of electricity charged by the user, d represents the discount type used for the user's charging payment. For example: User a charged at charging station 10 at 9 am. It had been 5 days since the last charging. He charged a total of 23 KWh of electricity and used a cash discount at checkout, which reduced the original bill amount by 2 yuan. In the end, he only spent 20 yuan to complete this charging session. This user charging behavior is represented as (5, *morning, peak, 10, 23 KWh, cashdiscount*)

User charging behavior prediction refers to the analysis of user historical charging behavior from multiple perspectives to forecast their charging behavior patterns over a future period of time. But it is clear that this prediction process cannot directly utilize all of the user's historical charging data. We assume that a user's current charging behaviors are related to their most recent charging behaviors. Suppose we have a time series dataset:

$$P = \{a_1,\ a_2,\ \cdots,\ a_T\} \tag{2}$$

where a_i represents a six-tuple containing the observation values for the six attributes of charging interval, time slot, price level, charging station, charging energy and discount type at time i.

The goal of user charging behavior prediction is to use the recent charging behavior of users to predict the user charging behavior at the next time step. By training a prediction module, we are able to forecast the user charging behavior at the next time step using historical charging data.For generality, let's denote the prediction module as f. The formal definition of the prediction process is as follows:

$$p_{t+k+1} = f(W_{t:t+k})\ ,\ W_{t:t+k} = \{a_t,\ a_{t+1},\ \cdots,\ a_{t+k}\} \tag{3}$$

where $W_{t:t+k}$ represents the historical charging behavior of user from time t to time $t + k$. There are various methods to set up the prediction module, such as traditional approaches like ARIMA [27], LSTM [26], XGBoost [28], or we can utilize existing LLMs. Based on the powerful language modeling capabilities of

these LLMs, we can design prompts that incorporate the features of each data dimension to perform time series prediction in conjunction with the LLMs.

4. DATASET PREPARATION

In this section,we will introduce the detail of data processing.Given a database of user charging behaviors, the processing of this data will result in a dataset of user charging activities.To achieve this objective, we have designed a methodology that encompasses three steps, including: (1) Data pre-processing, (2) Data selection, and (3) Data building. The following will sequentially introduce these three steps.

Step 1: Data pre-processing. In this step, we first perform null value removal, eliminating data points where the charge is zero. Subsequently, we encode the Chinese characters in the data using IDs, and we apply coarse-grained discretization to the discount type in the data. Considering the mentioned scenario where the charge is zero, the cause could be a malfunctioning charging pile that led to the interruption of the charging behavior. Therefore, this portion of the data should be flagged as abnormal and deleted.

Step 2: Data selection. In this step, we first combine the charging behavior data where the charging intervals are short and the charging amounts are small. Following this, we filter the charging behavior data of users, selecting only the data from users who have at least 10 charging behavior records within a specific time interval. Finally, the obtained charging behavior data corresponding to each user is sorted by time.

Step 3: Data building. In this step, we first construct a dataset for each group of data corresponding to a user by employing a sliding window approach, with a window size of 9 and a sliding step of 1. Next, we set the instructions for inputting the prediction model. Finally, we aggregate the obtained datasets into a final dataset and convert it to JSON format.

The flowchart of data processing is illustrated in Figure 1.

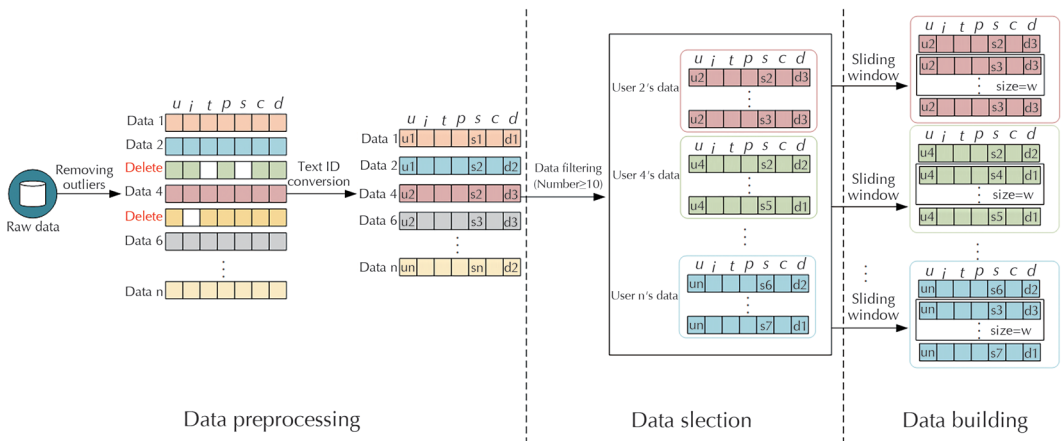


Figure 1. The flowchart of data processing.

5. EVCHARGING-GPT

In this section, we provide a unified mathematical definition for EV user charging behavior prediction, which includes the input, output and objective function of the model. This definition aims to build a framework that can understand and predict the charging behavior of users, the ChargingEV-GPT framework diagram is shown in Figure 2.

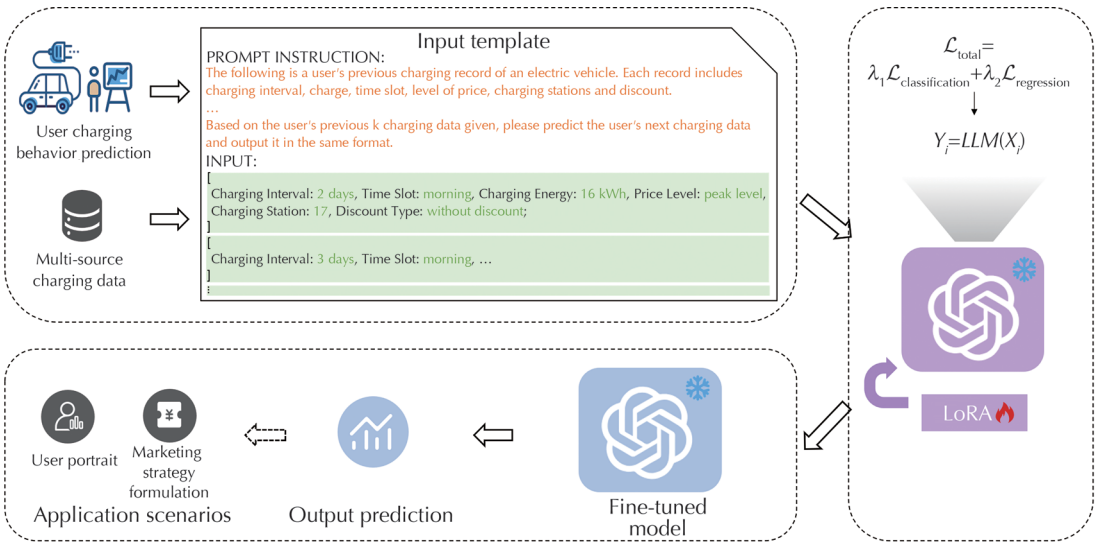


Figure 2. ChargingEV-GPT framework diagram.

The core of EVCharging-GPT is to use the powerful text generation and understanding capabilities of LLMs to transform the charging behavior prediction task of electric vehicle users into a text-to-text format. Through the carefully designed prompt template, the model can effectively fuse the time series data and the static classification data to generate accurate prediction results.

5.1 Structured Input Design and Contextual Understanding

First, in the definition of the task in Section 3, we define the six-tuple time series P. For LLMs, feeding a piece of data directly to the output may lead to random and inaccurate outputs that the model does not understand the task, even if we have represented P in natural language. To this end, we design structured statements that guide the model to understand the task and generate predictions. Thus, the input X_i to the model is a combination of the cue template and the six-tuple P.

Specifically, we set the sliding window length k to 9, which means that P contains relevant information about the past nine historical charging times of user i, and these data provide the context of user behavior for the model. We specifically classify the six-tuples in P into two information types: one is time series features, such as charging interval and charging energy, which reflect the dynamic changes of user

charging behavior. The other is the static category factor, such as time slot, price level, charging station and discount type, which play a key role in the user's charging decision.

We have designed a special Input Template for this purpose, the specific format can be seen in the "Input Template" section of Figure 2.

5.2 Model Processing and Prediction Mechanism

We utilize LLMs, such as Llama or Qwen, as the backbone network for our model. These LLMs have been pre-trained on vast amounts of text data, enabling them to understand and generate natural language effectively. With these models, we can generate future charging behavior predictions for user *i* based on its understanding of task-specific instructions.

Once the input X_i is fed, the LLM processes this information to understand the context and patterns in the historical charging behavior. The model leverages its pre-trained knowledge to identify trends, correlations, and contextual nuances within the data. This understanding forms the basis for generating accurate and contextually relevant predictions. The LLM's ability to understand natural language allows it to interpret the structured prompt and extract meaningful insights from the historical charging data.

After comprehending the context, the LLM generates a text output that represents the predicted attributes of the next charging event. This output Y_i includes all six attributes of the charging behavior: charging interval, time slot, price level, charging station, charging energy, and discount type. The model's ability to generate coherent text ensures that the prediction is not only accurate but also contextually relevant, providing a comprehensive view of the user's next charging behavior. The prediction Y_i can be computed as follows:

$$Y_i = LLM(X_i) \tag{4}$$

The key to EVCharging-GPT's ability to capture complex patterns in time series data is the self-attention mechanism in the Transformer architecture. This mechanism dynamically weighs the importance of different parts of the input sequence, allowing the model to capture long-range dependencies and intricate temporal relationships.

5.3 LoRA Fine-tuning Optimization

Our goal is to accurately predict the future charging behavior of a specific user based on a range of textual information. To achieve this goal, we optimize the model so that it can understand and process natural language input and generate predictive outputs related to user charging behavior. In order to further improve the prediction ability of the model, we adopt the LoRA (Low-Rank Adaptation) technique to fine-tune the pre-trained LLM. LoRA adjusts the model parameters by introducing a low-rank structure into the weight matrix of the model, so as to achieve efficient fine-tuning of the model. The steps are as follows:

Step 1: Initialize LoRA adapters: Add LoRA adapters to each Transformer layer of the model. These adapters consist of two low-rank matrices W_1 and W_2, and a trainable scaling factor α.

Step 2: Fine-tuning process: During fine-tuning, we freeze the original weights of the LLMS and only update the parameters of the LoRA adapter. In this way, the model can learn task-specific adjustments without destroying the knowledge learned during the pre-training phase.

Step 3: Update rules: The update rules of LoRA adapter are as follows:

$$W' = W + \alpha\left(W_1\ W_2\right) \tag{5}$$

Where W is the original weight matrix and W' is the updated weight matrix.

Step 4: Combined loss function: We employ a combined loss function that combines the loss from a classification task (cross-entropy loss) and the loss from a regression task (mean square error loss) to simultaneously optimize the model's performance on classification and continuous value prediction. The combined loss function is defined as follows.

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{classification}} + \lambda_2 \mathcal{L}_{\text{regression}} \tag{6}$$

Here, λ_1 and λ_2 are weight parameters to balance the importance of the two tasks. The loss for the classification task is calculated using the cross-entropy loss:

$$\mathcal{L}_{\text{classification}} = -\sum_{i=1}^{I}\sum_{c=1}^{C} y_{ic} \log\left(p_{ic}\right) \tag{7}$$

Where y_{ic} is an indicator variable such that $y_{ic} = 1$ if category c is the true offer type for user i and $y_{ic} = 0$ otherwise; p_{ic} is the probability that the model predicts that user i belongs to class c. The loss for the regression task is calculated using the mean squared error loss:

$$\mathcal{L}_{\text{regression}} = \frac{1}{IK}\sum_{i=1}^{I}\sum_{k=1}^{K}\left(y_i^k - \hat{y}_i^k\right)^2 \tag{8}$$

where I is the number of users, K is the number of predicted features, y_i^k is the actual value of the predicted feature for user i, and $\hat{y}_i^k$ is the predicted value.

Step 5: Merging weights after fine-tuning: After fine-tuning, the weights of the LoRA adapter are merged with the original model weights to obtain the final fine-tuning model.

6. EXPERIMENTS AND ANALYSIS

6.1 Dataset

In this subsection, we construct a real-world dataset for comprehensive experiments based on the methodology described in the Section Dataset Preparation. The original data comes from the charging stations service database being continuously updated, which is from a certain company. We selected

1,013 representative users from the raw data and extract 19 charging records for each user. Each historical record is reorganized into the following six-tuple: charging interval, charge, time slot, price level, charging station and discount. The average charging interval is 5 days and the average charge is 59 hours.

6.2 Baselines

Our method is benchmarked against two representative types of methods: the statistical methods and LLM-based methods, which are separately used as single-task prediction and multi-ask prediction. To be specific, the statistical methods are as follows:

- **ARIMA** [27]. ARIMA is a statistical model used for time sequence forecasting that combines autoregressive, differencing, and moving average components to capture various patterns and trends in data. The paper [27] reviews and discusses the application of ARIMA-based time series analysis methods in forecasting, including model building, parameter estimation, and model building, as well as their relationship with other forecasting methodologies.
- **XGBoost** [28]. XGBoost is an optimization of the gradient boosting framework used to solve regression, classification and ranking problems in machine learning. Luo et al. [28] propose an algorithm based on the stacking integration algorithm of neural network with XGBoost, and the integrated algorithm is employed for forecasting the mutli-feature load dataset in the Quanzhou area from 2016 to 2018.
- **LSTM** [26]. LSTM is a typical recurrent neural network with long short-term memory, which is employed for predicting the electric vehicle battery swapping demand, particularly considering temporal demand patterns obtained from the dataset.

The open-source language models of LLMs-based methods are ChatGLM4, Qwen2.5, and Llama3. The details are as follows:

- **Llama3** [3]. Llama3 is a herd of LLMs that natively support multilinguality, coding, reasoning, and tool usage, with the largest model being a 405B-parameter dense Transformer. Llama3.1-70B-Instruct and Llama3.1-8B-Instruct are used for foundation models.
- **ChatGLM4** [1]. GLM4 language serise represent most capbale models that are trained with all the insights and lessons gained form the preceding three generation of ChatGLM. Its human perference-aligned version GLM-4-9B-Chat is used as a baseline method, which shows superior performance beyond Llama3.1-8B.
- **Qwen2.5** [29]. Qwen2.5 is the latest addition to the Qwen family, focusing on smarter and more knowledgeable language models. Besides the instruction-tuned language models, the base language model of flagship open-source model Qwen2.5-72B reaches top-tier performance even against larger models like Llama-3-405B. Qwen2.5-72B-Instruct and Qwen2.5-7B-Instruct are used for baselines.

Two closed-source flagship language models: Qwen-Max, and GLM4-Plus, are employed for the benchmark. In addition, Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct, are added to single-task

prediction, to enhance the credibility of the experiments. We ensure a fair comparison by standardizing evaluation conditions across all models.

6.3 Evaluation Metrics

We use the root mean square error (RMSE), the mean absolute error (MAE), and the accuracy for evaluation.

RMSE metric evaluates the performance of forecasting the continuous variables: charging interval and charge energy. RMSE is a commonly used measure to assess the difference between values predicted by a model and the values actually observed. It provides a comprehensive view of how well the model's predictions match the actual data. Moreover, lower RMSE values indicate a better fit to the data, making it easier to select the most accurate model among serveral alternatives. The RMSE is calculated as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{9}$$

where n is the total number of data points in the dataset; y_i is the actual observed value for the i-th data point; $\hat{y}_i$ is the predicted value for the i-th data point, where the hat symbol^indicates that it is an estimated value rather than the actual value.

MAE metric is also used for evaluating methods' performance on continuous variables, and represents the average of the absolute differences between forecasted values and observed values. Compared to RMSE, MAE is less sensitive to outliers since it uses absolute values instead of squared values, and then large errors do not get disproportionately magnified, making MAE a more robust metric in the presence of extreme values in the data. Here is the formula for calculating MAE:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{10}$$

where n is the total number of data points in the dataset; y_i is the actual observed value for the i-th data point; $\hat{y}_i$ is the predicted value for the i-th data point, where the hat symbol^indicates that it is an estimated value rather than an actual value.

Accuracy metric assesses the performance of forecasting the categorical variables: time slot, price level, charging station and discount type. Accuracy metric represents the proportion of correct predictions, which include both true positives and true negatives, made by a model out of the total number of cases in a dataset. It is a key performance metric used to evaluate the effectiveness of classification models. The formula for calculating Accuracy metric is:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{11}$$

where TP represents the number of positive instances that are correctly classified as positive; TN represents the number of negative instances that are correctly classified as negative; FP represents the number of negative instances that are incorrectly classified as positive; FN represents the number of positive instances thar are incorrectly classified as negative.

6.4 Implementation Details

Environment Settings. We implement our approach using PyTorch and Transformers. All experiments are conducted on a Linux workstation with an Intel (R) Xeon (R) Silver 4210 CPU @ 2.20 GHz, 128 GB RAM, and two GeForce RTX 3090 GPUs with 48 GB VRAM.

Training Settings. We create a dataset comprising 80% training samples based on instruction-following input-output pairs of six-tuple. The backbone language models are Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct. We use LoRA for fine-tuning, with a batch size of 8 and a learning rate set to 1e-5, and the training duration extended to 2 epochs.

Inference Settings. Testing data is 20% of the whole samples. Qwen-Max and GLM4-Plus are accessed through their APIs at a temperature setting of 0. We employ RMSE, MAE and Accuracy for evaluating all models.

6.5 Main Results

The forecasting performance for both multi-task prediction and single-ask prediction includes a six-tuple. The main results are presented in Table 1, and Table 2. The evaluation results of continuous variables are shown in Table 1 in different language models, while the performance for categorical variables is shown in Table 2. Several key findings can be observed.

Closed-source model Qwen-Max leads in accuracy in terms of time slot, price level and charging station, compared to other language models, and only slightly lags behind Llam3.1-8B-Instruct in the discount prediction task. In terms of predicting charging interval and charge, Qwen2.5-72B-Instruct has essentially achieved the best performance.

It is observed that the performance of two closed-source language models significantly outperforms those of open-source language models. Qwen-Max achieves the best performance across various reasoning types on average. For language models within the same series, those with a larger number of parameters have demonstrated better performance, which also indicates that language models adhere to the scaling laws in data science. For instance, within the Qwen2.5 series, Qwen2.5-72B-Instruct outperforms Qwen2.5-7B-Instruct in all tasks; similarly, within the Llama3.1 series, Llama3.1-70B-Instruct leads over Llama3.1-8B-Instruct.

In addition, the role of fine-tuning language models for multitask prediction may not be as meaningful as expected. Specifically, we first evaluate two large open source models, Llama3.1-8B-Instruct and

Table 1. The evaluation results for continuous variables of multi-task prediction. The best results are shown in bold, while the second-best results are underlined.

Model	Charging Interval		Charging Energy	
	RMSE	MAE	RMSE	MAE
GLM4-Plus	5.255	2.976	**10.919**	7.891
Qwen-Max	5.245	2.861	11.050	**7.877**
Qwen2.5-72B-Instruct	**5.226**	**2.850**	11.214	7.965
Llama3.1-70B-Instruct	6.547	4.045	11.639	8.460
GLM4-9B-Chat	5.314	3.031	12.029	8.819
Llama3.1-8B-Instruct	6.603	4.465	12.395	8.881
Qwen2.5-7B-Instruct	5.583	3.142	11.697	8.310
Llama3.1-8B-Instruct-Tuned	5.321	2.986	12.081	8.707
Qwen2.5-7B-Instruct-Tuned	5.412	3.099	11.177	7.937

Table 2. The evaluation results for categorical variables of multi-task prediction. The best results are shown in bold, while the second-best results are underlined.

Model	Time Slot	Price Level	Charging Station	Discount Type
GLM4-Plus	0.602	0.572	**0.704**	0.671
Qwen-Max	**0.612**	**0.674**	**0.704**	0.691
Qwen2.5-72B-Instruct	0.606	0.582	0.703	0.683
Llama3.1-70B-Instruct	0.582	0.576	0.586	0.562
GLM4-9B-Chat	0.579	0.548	0.678	0.653
Llama3.1-8B-Instruct	0.431	0.469	0.620	**0.692**
Qwen2.5-7B-Instruct	0.541	0.513	0.687	0.650
Llama3.1-8B-Instruct-Tuned	0.555	0.541	0.684	0.687
Qwen2.5-7B-Instruct-Tuned	0.579	0.561	0.680	0.674

Qwen2.5-7B-Instruct; subsequently, these models are fine-tuned in the training set and then evaluated in the test set. Compared with the foundation models, the fine-tuned language models do not demonstrate performance worthy of the resources expended on fine-tuning. However, it cannot be denied that this situation may arise because the model's parameter count is not large enough to showcase the advantages of language models after fine-tuning.

6.6 Single-Task Prediction

The indicators in the six-tuple may inherently influence each other. For instance, the Charging Interval and Charging Energy might be affected by factors such as Time Slot and Price Level. Therefore, the prediction of the six-tuple can be viewed as a multi-task learning problem, where the model may learn the underlying interrelationships between different indicators. Training each task independently allows us to gain insights into the predictive power and characteristics of each indicator in isolation, as well as evaluate the performance of different models across various tasks. Hence, we conducted a comparison of the prediction results for both the six-tuple and individual tasks to verify whether the multi-task model improves overall prediction accuracy, or if independent task training is more suited to certain specific tasks. Similar to the six-tuple prediction task, for the single-task training, we chose the following evaluation metrics: for continuous variables, we used Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) to assess prediction loss. For discrete variables, we only evaluated the model's prediction accuracy.

In the single-task training experiments, we employed traditional models from the baseline (LSTM [26], XGBoost [28], ARIMA [27]) and two LLM (LLaMa3.1-8B-Instruct [3], Qwen2.5-7B-Instruct [29]) to train and predict individual tasks.

In the prediction of continuous data, ARIMA is capable of efficiently modeling the underlying structure of time series, whereas XGBoost excels at handling complex decision boundaries and multidimensional features in classification tasks, thereby enhancing classification accuracy [30]. By assigning each model to the task for which it is best suited, the overall prediction performance can be optimized.For the reasons outlined above, we employ ARIMA exclusively for predicting continuous variables and reserve XGBoost for the classification tasks, including Time Slot,Price Level, Charging Station, and Discount Type. We utilize both large models and our proposed model for predicting all these individual tasks.

As shown in Table 3, the performance on the test set for the continuous variables across different models. The LSTM model, as an earlier RNN method designed to capture hidden patterns in time series, achieved optimal results on almost all indicators, with only the Charging Interval's MAE metric reaching the second-best level. Notably, among the two large models, the fine-tuned LLaMa3.1-8B-Instruct model achieved the best MAE for Charging Interval and the second-best RMSE, while the fine-tuned Qwen2.5-7B-Instruct model reached the second-best RMSE and MAE for Charging Energy.

As shown in Table 4, the performance on the test set for the discrete variables across the models. LSTM still performed well, achieving the best prediction accuracy for Price Level, and the second-best accuracy for Time Slot and Discount Type, with only a small gap compared to the best performance. XGBoost achieved the best prediction accuracy for Discount Type. Among the two large models, the fine-tuned LLaMa3.1-8B-Instruct model achieved the best accuracy for Time Slot, while the fine-tuned Qwen2.5-7B-Instruct model reached an accuracy of 72.4% for Charging Station prediction, which was the best performance.

Table 3. The evaluation results for continuous variables of single-task prediction. The best results are shown in bold, while the second-best results are underlined.

Model	Charging Interval		Charging Energy	
	RMSE	MAE	RMSE	MAE
ARIMA	5.755	4.082	10.079	7.429
LSTM	**4.537**	3.092	**8.879**	**6.706**
Llama3.1-8B-Instruct	6.678	4.762	13.695	9.341
Qwen2.5-7B-Instruct	5.606	3.509	11.346	8.473
Llama3.1-8B-Instruct-Tuned	5.325	**2.877**	11.110	8.228
Qwen2.5-7B-Instruct-Tuned	5.377	3.130	9.561	7.032

Table 4. The evaluation results for categorical variables of single-task prediction. The best results are shown in bold, while the second-best results are underlined.

Model	Time Slot	Price Level	Charging Station	Discount Type
LSTM	0.523	**0.554**	0.631	0.681
XGBoost	0.515	0.534	0.407	**0.689**
Llama3.1-8B-Instruct	0.396	0.465	0.644	0.624
Qwen2.5-7B-Instruct	0.486	0.504	0.685	0.656
Llama3.1-8B-Instruct-Tuned	**0.524**	0.507	0.673	0.641
Qwen2.5-7B-Instruct-Tuned	0.516	0.544	**0.724**	0.673

7. CONCLUSION

In this study, we pioneered the use of LLMs to predict the charging behavior of EV users. Our model, ChargingEV-GPT, achieves an innovative way of integrating data by turning the prediction task into a text-to-text problem. This approach is capable of fusing multiple data types, including time series data (such as a user's charging volume and charging interval) and categorical data (such as discount type, charging period, electricity price period, and charging station). ChargingEV-GPT also introduces the Lora-based fine-tuning technology to realize the adaptive adjustment of knowledge, which further improves the model's ability to predict the charging behavior of electric vehicles. Through extensive experiments on real-world urban datasets, as well as in-depth testing on a single task, we verify the effectiveness of our method in predicting user charging behavior, and compare the performance of language models of different scales with traditional models on this task.

To boost prediction accuracy and expand applications, future research should focus on optimizing model structures and training for better performance, enhancing data with new sources and augmentation techniques, integrating multi-modal data for comprehensive predictions, increasing model interpretability for user understanding, developing real-time predictive systems, and ensuring model generalization across different cities and user groups. These efforts will advance predictive modeling.

Through the research in these directions, we expect to promote the development of EV charging behavior prediction technology and provide support for the optimization of smart grid and EV charging infrastructure.

AUTHOR CONTRIBUTIONS

Houzhi Li: proposed the research problems; designed the research framework; wrote and revised the manuscript. Jinyu Wang: proposed the research problems; designed the research framework. Zixin Jiang: designed the research framework; performed the research; wrote and revised the manuscript. Guorui Su: performed the research; collected and analyzed the data; wrote and revised the manuscript. Chaowen Yan: performed the research; collected and analyzed the data; wrote and revised the manuscript. Hao Su: performed the research; collected and analyzed the data; wrote and revised the manuscript. Zhichun Wang: designed the research framework; performed the research; wrote and revised the manuscript.

ACKNOWLEDGMENTS AND FUNDING

This work was supported by the National Natural Science Foundation of China (No. 62276026).

REFERENCES

[1] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Rojas, G. Feng, H. Zhao, H. Lai *et al.*, "Chatglm: A family of large language models from glm-130b to glm-4 all tools," *arXiv preprint arXiv: 2406.12793*, 2024.

[2] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv: 2303.08774*, 2023.

[3] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv: 2307.09288*, 2023.

[4] Y. Tay, J. Wei, H. W. Chung, V. Q. Tran, D. R. So, S. Shakeri, X. Garcia, H. S. Zheng, J. Rao, A. Chowdhery *et al.*, "Transcending scaling laws with 0.1% extra compute," *arXiv preprint arXiv: 2210.11399*, 2022.

[5] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27730–27744, 2022.

[6] L. Yao, J. Peng, C. Mao, and Y. Luo, "Exploring large language models for knowledge graph completion," *arXiv preprint arXiv: 2308.13916*, 2023.

[7] Z. Liu, W. Ping, R. Roy, P. Xu, M. Shoeybi, and B. Catanzaro, "Chatqa: Building gpt-4 level conversational qa models," *arXiv preprint arXiv: 2401.10225*, 2024.

[8] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu *et al.*, "Instruction tuning for large language models: A survey," *arXiv preprint arXiv: 2308.10792*, 2023.

[9] D. Aggarwal, A. Sathe, I. Watts, and S. Sitaram, "Maple: Multilingual evaluation of parameter efficient finetuning of large language models," *arXiv preprint arXiv: 2401.07598*, 2024.

[10] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen *et al.*, "Parameter-efficient fine-tuning of large-scale pre-trained language models," *Nature Machine Intelligence*, vol. 5, no. 3, pp. 220–235, 2023.

[11] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv: 2106.09685*, 2021.

[12] Z. Guo, L. Xia, Y. Yu, Y. Wang, Z. Yang, W. Wei, L. Pang, T.-S. Chua, and C. Huang, "Graphedit: Large language models for graph structure learning," *arXiv preprint arXiv: 2402.15183*, 2024.

[13] Y. Tan, H. Lv, X. Huang, J. Zhang, S. Wang, and C. Yang, "Musegraph: Graph-oriented instruction tuning of large language models for generic graph mining," *arXiv preprint arXiv: 2403.04780*, 2024.

[14] J. Wang, J. Wu, Y. Hou, Y. Liu, M. Gao, and J. McAuley, "Instructgraph: Boosting large language models via graph-centric instruction tuning and preference alignment," *arXiv preprint arXiv:2402.08785*, 2024.

[15] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan *et al.*, "Time-llm: Time series forecasting by reprogramming large language models," *arXiv preprint arXiv:2310.01728*, 2023.

[16] C. Sun, H. Li, Y. Li, and S. Hong, "Test: Text prototype aligned embedding to activate llm's ability for time series," *arXiv preprint arXiv: 2308.08241*, 2023.

[17] T. Zhou, P. Niu, L. Sun, R. Jin *et al.*, "One fits all: Power general time series analysis by pretrained lm," *Advances in neural information processing systems*, vol. 36, pp. 43322–43355, 2023.

[18] X. Liu, J. Hu, Y. Li, S. Diao, Y. Liang, B. Hooi, and R. Zimmermann, "Unitime: A language-empowered unified model for cross-domain time series forecasting," in *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 4095–4106.

[19] D. Cao, F. Jia, S. O. Arik, T. Pfister, Y. Zheng, W. Ye, and Y. Liu, "Tempo: Prompt-based generative pre-trained transformer for time series forecasting," *arXiv preprint arXiv: 2310.04948*, 2023.

[20] C. Chang, W.-C. Peng, and T.-F. Chen, "Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms," *arXiv preprint arXiv:2308.08469*, 2023.

[21] C. Ding, S. Sun, and J. Zhao, "Mst-gat: A multimodal spatial-temporal graph attention network for time series anomaly detection," *Information Fusion*, vol. 89, pp. 527–536, 2023.

[22] C. Huang, S. Li, R. Liu, H. Wang, and Y. Chen, "Large foundation models for power systems," in *2024 IEEE Power & Energy Society General Meeting (PESGM)*. IEEE, 2024, pp. 1–5.

[23] R. S. Bonadia, F. C. Trindade, W. Freitas, and B. Venkatesh, "On the potential of chatgpt to generate distribution systems for load flow studies using opendss," *IEEE Transactions on Power Systems*, 2023.

[24] S. Tu, Y. Zhang, J. Zhang, and Y. Yang, "Powerpm: Foundation model for power systems," *arXiv preprint arXiv:2408.04057*, 2024.

[25] H. Qu, H. Li, L. You, R. Zhu, J. Yan, P. Santi, C. Ratti, and C. Yuen, "Chatev: Predicting electric vehicle charging demand as natural language processing," *Transportation Research Part D: Transport and Environment*, vol. 136, p. 104470, 2024.

[26] S. Wang, A. Chen, P. Wang, and C. Zhuge, "Short-term electric vehicle battery swapping demand prediction: Deep learning methods," *Transportation Research Part D: Transport and Environment*, vol. 119, p. 103746, 2023.

[27] P. Newbold, "Arima model building and the time series analysis approach to forecasting," *Journal of forecasting*, vol. 2, no. 1, pp. 23–35, 1983.

[28] S. Luo, B. Wang, Q. Gao, Y. Wang, and X. Pang, "Stacking integration algorithm based on cnn-bilstm-attention with xgboost for short-term electricity load forecasting," *Energy Reports*, vol. 12, pp. 2676–2689, 2024.

[29] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han *et al.*, "Qwen technical report," 2023. [Online]. Available: https://arxiv.org/abs/2309.16609

[30] I. Paliari, A. Karanikola, and S. Kotsiantis, "A comparison of the optimized lstm, xgboost and arima in time series forecasting," in *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE, 2021, pp. 1–7.

AUTHOR BIOGRAPHY



Houzhi Li, Deputy Director of the Platform Operation Center and Chief Operating Officer at State Grid Intelligent Vehicle Network Technology Company. He obtained his master's degree from Tianjin University and has extensive experience in EV charger construction, operation, maintenance, corporate customer development, interconnection of charging piles, plug-and-charge with post-payment, and other innovative initiatives.



Jinyu Wang graduated from Beijing University of Posts and Communications with a master's degree in Communications and Information Systems in 2014. He currently works at State Grid Intelligent Vehicle Network Technology Company, focusing on the development and operation of charging platform systems.



Zixin Jiang received her BS degree in Computer Science and Technology from the College of Artificial Intelligence, Beijing Normal University in 2024. Currently, she is a graduate student in the School of Artificial Intelligence at Beijing Normal University. Her research interests include large language models, and knowledge graph completion.



Guorui Su received his BS degree from the School of Computer and Communication Engineering at Northeastern University in Qinhuangdao. Currently, he is a graduate student in the School of Artificial Intelligence at Beijing Normal University. His research interests include knowledge fusion and entity alignment.



Chaowen Yan received his BS degree in Computer Science and Technology from Nanjing Unversity of Finance & Economics, China, in 2024. He is currently a graduate student in the School of Artificial Intelligence at Beijing Normal University. His research interests include large language models and knowledge graph fusion.



Hao Su received his BS degree in Data Science and Big Data Technology from Henan Agricultural University, China, in 2019. He is currently a graduate student in the School of Artificial Intelligence at Beijing Normal University. His research interests include large language models, knowledge graphs, and their applications.



Zhichun Wang is currently an associate professor at the School of Artificial Intelligence, Beijing Normal University. He received his Ph.D. degree from Tianjin University in 2010, and worked as post-doctoral researcher in Tsinghua University from 2011 to 2012. His research interests include large language models and knowledge graphs.