The Open V2X Management Platform: an intelligent charging station management system

Christos Dalamagkas[a], VD Melissianos[a], George Papadakis[a], Angelos Georgakis[a], Vasileios-Martin Nikiforidis[a], Kostas Hrissagis-Chrysagis[a]

[a]*Research Projects Coordination Department, Public Power Corporation, Athens, Greece*

Abstract

We present an open-source web-based system, called Open V2X Management Platform (O-V2X-MP), which facilitates the management of charging points for electric vehicles with the goal of realizing Vehicle-to-Everything (V2X) scenarios. First, we describe its backend, which comprises several components connected through a microservices architecture leveraging Docker containers. Then, we elaborate on its frontend, which provides numerous functionalities for common users (i.e., EV drivers) and administrators. Finally, we demonstrate its data analytics capabilities, showing that O-V2X-MP can seamlessly integrate AI pipelines from the Python ecosystem. In particular, we examine two tasks of particular interest for charging point operators: (i) the clustering of EV drivers into profiles of predictable behavior, and (ii) the prediction of the overall daily load for each individual charging station. In our experiments, we use proprietary and public real-world data, verifying the high effectiveness achieved in both tasks.

Keywords: OCPP protocol, Electric Vehicles, charging points management, open-source Web platform, Data Analytics

1. Introduction

In the energy sector, vehicle-to-everything (**V2X**) is the vision of leveraging electric vehicles (**EVs**) as a means of improving grid stability by taming the stochasticity of renewable energy resources [1]. V2X is actually a broad vision that includes specialized scenarios like vehicle-to-grid (**V2G**) [2], where EVs act as a mobile energy storage, charging and discharging to the power grid. Indeed, treating a fleet of EVs as a huge battery can help balance grid load, reduce emissions, and even provide emergency power during outages [3]. V2X also plays a crucial role in demand-response scenarios, which manage electricity demand, particularly during peak periods when the grid is under stress; the goal is to incentivize consumers (EV drivers in our case) to reduce their electricity consumption, by shifting the charging of their vehicles to off-peak times [4].

As a major *charging point operator* (**CPO**), Public Power Corporation strives to realize the V2X vision by making the most of the largest network of EV chargers in Greece, with more than 2,000 charging stations nationwide[1]. Maintaining this large network of EV chargers is a complex task that involves a wide diversity of technical challenges. To address them, the R&D group of PPC has been developing an open-source, web-based Charging Station Management System (**CSMS**) that supports V2X scenarios, called *Open V2X Management Platform* (**O-V2X-MP**). This platform enables the remote control, monitoring, and maintenance of charging stations. It can also retrieve remote diagnostics, detailing the charger's health status, real-time availability, and audit logs. To the best of our knowledge, O-V2X-MP is the first open-source CSMS platform with a wide range of value-added services like advanced cyber-security mechanisms and data analytics on top of the transactional data gathered from the EV chargers.

We provide a thorough description of O-V2X-MP, delving both into its backend and frontend. At its core lies the *Open Charge Point Protocol* (**OCPP**)[2], which specifies the communication between the platform and the charging points. O-V2X-MP supports the latest version of OCPP, 2.0.1 [5], which offers more advanced features, like additional smart charging functionalities, improved transaction handling, and device management. It also provides backward compatibility for chargers implementing earlier OCPP versions, like the popular 1.6 [6].

The O-V2X-MP backend comprises several individual components that are effectively integrated through a microservices architecture. We visualize this architecture through the *C4 model* [7], which unfolds four levels:

1. At the highest level, it elaborates on the principal entities engaged with the platform, such as the EV users and the platform manager (see Section 4.1).
2. At the second level, it delves into the auxiliary services that support the O-V2X-MP platform and the communications between them (see Section 4.2).

Email addresses: c.dalamagkas@ppcgroup.com (Christos Dalamagkas), v.melissianos@ppcgroup.com (VD Melissianos), g_a.papadakis@ppcgroup.com (George Papadakis), an.georgakis@ppcgroup.com (Angelos Georgakis), v.nikiforidis@ppcgroup.com (Vasileios-Martin Nikiforidis), k.chrysagis@ppcgroup.com (Kostas Hrissagis-Chrysagis)

[1]Please refer to https://www.deiblue.gr for more details.

[2]Please refer to https://openchargealliance.org/protocols/open-charge-point-protocol for more details.

3. At the third level, it zooms into the interior of the O-V2X-MP platform, providing details about the modules it encompasses (see Section 4.3).

4. The lowest level focuses on implementation details like UML and entity relationship diagrams, but is omitted for brevity.

The frontend of O-V2X-MP satisfies a series of non-functional and functional requirements. The latter are distinguished into three types:

1. The generic ones, which apply to all types of accounts and include the authentication and the overview of charging stations along with their daily tariff and capacity schedules and the corresponding weather conditions.

2. The user ones, which are crafted for EV drivers and include the registration form, the update of their preferences and most importantly, the routing recommendations and the charging history management.

3. The administrator ones, which are crafted for CPOs and pertain to the update of charging station information (i.e., their location and their technical characteristics), to the live monitoring of charging stations and their sessions as well as to the manual specification and update of tariffs.

These functionalities are demonstrated through a video that is available at: `https://www.youtube.com/watch?v=FzjwQQ0XEKI&ab_channel=EV4EUDashboard`. The source code of the O-V2X-MP backend is publicly available at `https://github.com/EV4EU/ov2xmp` and of the O-V2X-MP frontend at `https://github.com/EV4EU/ov2xmp-gui`. Both repositories will be maintained for at least the next five years, in the context of PPC's R&D program.

We also focus on the value-added services that equip O-V2X-MP with unique characteristics. These services are mostly offered by the data analytics module, which allows for combining any ML model from the Python ecosystem into pipelines that provide insights into the data gathered by the platform. This is demonstrated through extensive experiments involving real, operational data. Two use cases are experimentally examined:

1. The clustering of EV drivers into meaningful user profiles that allow for predicting their behavior.

2. The prediction of the demand for the energy to be delivered by each charging station for the day ahead.

Both tasks are crucial for forecasting the overall load in a network of charging stations, ensuring the stability of the energy grid.

To sum up, O-V2X-MP is the first open-source platform that combines OCPP with a wide range of value-added services like data analytics on top of the transactional data gathered from EV chargers. To this end, O-V2X-MP comprises numerous individual components that are effectively integrated through a microservices architecture. We delve into its backend and frontend in Sections 4 and 5, respectively, while Section 6 experimentally evaluates two data analytics services. We conclude our work with a brief discussion about the future extension to the O-V2X-MP platform in Section 7.

2. Preliminaries

OCPP constitutes a standardized communication protocol designed to facilitate seamless data exchange between EV chargers and a CSMS. Developed and maintained by the Open Charge Alliance (OCA)[3], OCPP has been established as an open-source protocol, ensuring vendor neutrality and fostering interoperability within the EV charging infrastructure.

In summary, OCPP defines a message structure and a set of functionalities that govern the communication flow between EV chargers and the CSMS. This enables:

1. *Monitoring and Control*: The CSMS can remotely monitor the status of EV chargers, including power delivery, connector availability, and fault conditions. Additionally, control commands can be issued to initiate, stop, or schedule charging sessions.

2. *Metering and Billing*: OCPP facilitates the exchange of energy consumption data between the EV chargers and the CSMS, enabling accurate metering and subsequent billing of EV drivers.

3. *Plug & Play Compatibility*: An OCPP-compliant EV charger can seamlessly integrate with any OCPP-compliant CSMS, regardless of the specific vendor.

The O-V2X-MP platform supports the two main versions of the OCPP protocol: v1.6 [6] for smart charging scenarios, and v2.0.1 [5] for V2X scenarios.

2.1. OCPP 1.6

This version was released in 2015 and remains widely deployed, constituting the de-facto standard implemented by commercial EV chargers, due to its robust feature set and its low complexity. The O-V2X-MP platform supports all OCPP 1.6 operations. In fact, the following OCPP 1.6 messages [6], coming from the EV charger, can be processed by the O-V2X-MP:

- *BootNotification*: It runs during the initiation of the charging station to signal its availability to the CSMS and to provide information about its capabilities.

- *Heartbeat*: It is periodically transmitted by the charging station to confirm its operational status and connection to the CSMS.

- *StatusNotification*: It notifies the CSMS about the status of the charging station.

- *Authorize*: It is a command issued by the charging station to the CSMS, whenever a customer plugs their EV to initiate the customer authorization process.

[3] `https://openchargealliance.org`

- *StartTransaction*: It initiates the charging session, after user authorization, conveying details about the charger, the customer, and the charging parameters.
- *StopTransaction*: It notifies the CSMS about the termination of the charging session, including relevant details.
- *MeterValues*: It retrieves the current energy consumption of the charging station.
- Notification messages like *DiagnosticsStatusNotification* and *FirmwareStatusNotification*.

Moreover, the O-V2X-MP platform can use the following OCPP 1.6 messages [6]:

- *Reset*: It is used for restarting an EV charger.
- *RemoteStartTransaction*/*RemoteStopTransaction*: Issued by the EV user to start and stop an EV charging transaction through an app (e.g., a mobile app like eMSP – see Section 4.1).
- *ReserveNow*: Issued by the CSMS, on behalf of the EV user, to reserve an EV charger connector.
- *CancelReservation*: Used to cancel a reservation.
- *ChangeAvailability*: Used by the CPO to manually change the availability of an EV charger (e.g., to render it unavailable for preventing EV users from using it).
- *ChangeConfiguration*: It is used for changing a configuration parameter on the targeted EV charger.
- *ClearCache*: Clears the authorization cache of the targeted EV charger.
- *UnlockConnector*: Manually unlocks an EV charger connector.
- *GetConfiguration*: Retrieves the value of one or multiple configuration parameters from an EV charger.
- *GetCompositeSchedule*: Retrieves the composite schedule of the given duration.
- *ClearChargingProfile*: Deletes a charging profile from the targeted EV charger.
- *SetChargingProfile*: Sets a charging profile to the targeted EV charger and connector.
- *GetDiagnostics*: Instructs the EV charger to upload log files to a specific HTTP or FTP server.
- *UpdateFirmware*: Instructs the EV charger to download a firmware upgrade from a specific HTTP or FTP server.
- *TriggerMessage*: Instructs the EV charger to send a specific OCPP message to the CSMS.
- *GetLocalListVersion*: Retrieves the version of the local authorization list installed on the EV charger.
- *SendLocalList*: Sends a new local authorization list to the specified EV charger.

2.2. OCPP 2.0.1

First released in 2020, this remains the latest version of OCPP, conveying the numerous enhancements and additional functionalities. More specifically [5]:

- It offers advanced authorization techniques, such as remote transactions, PIN-Code, ISO 15118 Plug & Charge, Simple start button, Credit/debit card, Server-generated ID.
- It eliminates the need for a dedicated Virtual Private Network (VPN) tunnel between the EV charger and the CSMS as a means of ensuring security and encrypted communication. Instead, data packets are encrypted at the protocol level, inherently providing a secure connection.
- It incorporates security profiles for authentication, security logging, secure firmware updates, and security event log notifications.
- It minimizes the messages exchanged between the EV charger and the CSMS, thereby reducing data usage.
- It enables EVs to request the desired amount of energy in kilowatt-hours (kWh). This way, the CSMS is aware of the energy needs per EV, adjusting charging outputs accordingly.

Most importantly, OCPP 2.0.1 supports smart charging scenarios through the following commands:

- *ReserveNow*: It enables an EV user to reserve a specific charger for a designated period of time.
- *CancelReservation*: It revokes a reservation made by the previous command.
- *GetCompositeSchedule*: It retrieves a plan of the charging station's availability for a specific time period. This plan includes its current state, the planned maintenance, and any existing reservations of the charger.
- *SetChargingProfile*: It defines a charging profile for a specific time period, which is crucial in V2G applications with variable charging rates.

Note that these commands appear as part of OCPP 1.6, too, but their implementation in OCPP 2.0.1 has been enhanced, with every message distinguished into Request and Response Protocol Data Unit.

2.3. Related Works

O-V2X-MP supports both OCPP 1.6 and 2.0.1. In this respect, the work most relevant to ours is *SteVe*[4], an open-source CSMS that implements all versions of OCPP up to 1.6, enabling simple smart charging scenarios. Its backend is implemented in Java, and its frontend in JavaScript. However, SteVe has the following limitations:

[4] `https://github.com/steve-community/steve`

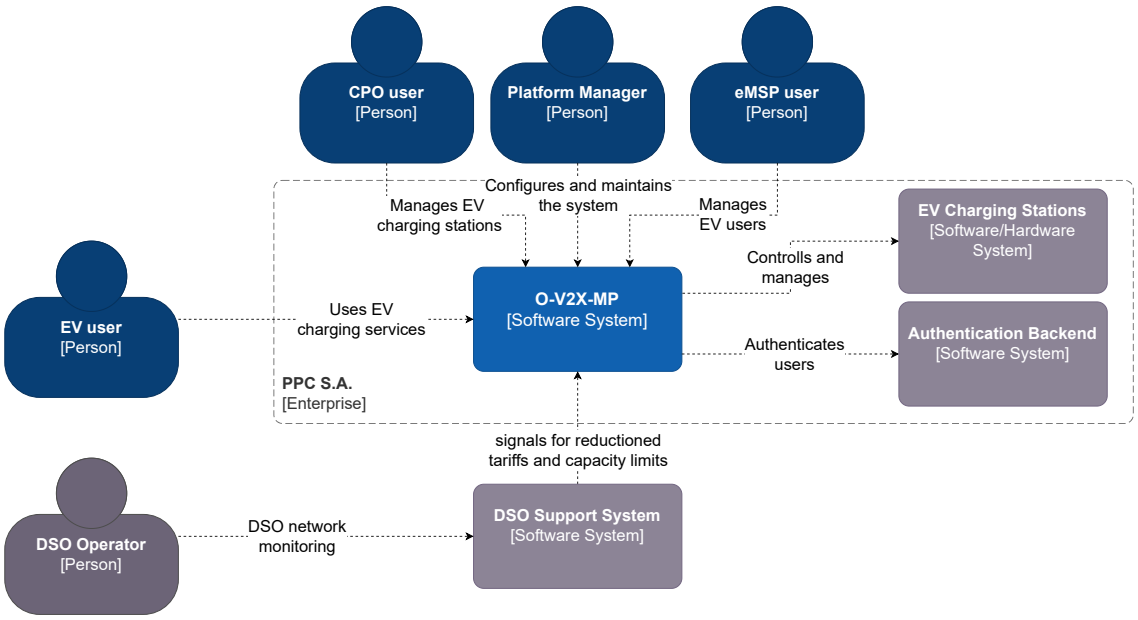

Figure 1: The System Context diagram of the O-V2X-MP platform.

- Its architecture is intricate and hard to extend with new modules and functionalities.
- It cannot support OCPP 2.0.1 without substantial modifications to its backend.
- It offers no value-added services on top of the transaction data.
- It lacks any effective cybersecurity mechanisms.

Our goal is to go beyond SteVe through an extensible open-source platform that leverages the Python AI ecosystem to combine the OCPP 2.0.1 functionalities with advanced services like data analytics and resilience mechanisms. Note that O-V2X-MP is equipped with advanced cybersecurity mechanisms, which ensure the confidentiality, availability and integrity of the sensitive information handled by the platform, but we omit them, as they are analyzed in a separate work [8].

3. System Overview

The O-V2X-MP platform was designed and developed such that it meets the following non-functional requirements:

1. *Scalability* with respect to the number of users and the number of integrated charging stations.
2. *High efficiency* (i.e., low run-times) so that all operations are carried out without delays.
3. *Modularity* to enhance the reuse of core components, reducing the code base and facilitating its maintenance.
4. *Extensibility* to reduce the development effort and time for adding new functionalities that are necessary for the smart charging and V2X scenarios of the EV4EU demos.
5. *Usability* in the sense that it enables users and administrators to perform any operation with a few clicks. Special care has been taken to support novice EV drivers that have no expert knowledge.

The first two requirements are met through a zero-intelligence interface, which moves all complex functionalities (e.g., joins between database tables) to the backend. The next two are met by the microservices architecture of the backend, as discussed in Section 4, while the last one is met by the intuitive user interface described in Section 5.

4. Backend

We now present the architecture of the O-V2X-MP backend, based on the C4 model [7],

a set of abstractions and diagrams that facilitate the description of the inner architecture of complex software systems. Following it,

the O-V2X-MP is analysed with the help of the following three diagrams:

1. The System Context diagram, which depicts the users and the external systems that interact with our CSMS.
2. The Container diagram, which provides a high-level overview of the CSMS architecture, highlighting how the responsibilities are distributed across the individual components and how they communicate with one another.
3. The Component diagram, which delves into a specific container and identifies its major structural building blocks and their interactions.

We elaborate on each diagram in a separate subsection below.

4.1. System Context View

Figure 1 depicts the system context diagram of O-V2X-MP, which involves the following **primary entities**:

1. The *EV user*, i.e., a driver who is using the O-V2X-MP platform in order to find the nearest and cheapest charging stations to charge their cars as well as to manage their payments and charging history.

2. The *CPO user*, who owns and manages the charging stations that have been registered to the platform. The goal is to ensure that the chargers are operational, and available around the clock. This encompasses tasks such as regular remote and on-site maintenance, diagnostics, price settings, and point-of-interest data management.

3. The *Platform Manager*, who administers the O-V2X-MP system. This entails maintenance duties (e.g., observation of the overall health of the O-V2X-MP services, troubleshooting and restart of services) as well as deployment and configuration of O-V2X-MP.

4. The *eMSP*[5] *user*, who is responsible for the mobile app that is typically used by EV users to connect to a charging station. The role of eMSP is to provide EV charging services to drivers by offering access to multiple charging points within a geographic area. To this end, the eMSP user manages the authentication credentials of drivers that are used for authorising EV charging transactions (e.g., their RFID card IDs).

O-V2X-MP interacts with the following **external systems**:

1. The *Authentication Backend* is an external directory or database of the organisation operating O-V2X-MP, which stores all the users accessing the O-V2X-MP. This backend can be an LDAP server or an Identity Provider compliant with the OpenID Connect protocol.

2. The *DSO Support System* is the platform operated by the regional or national Distribution System Operator (DSO). Its purpose is to monitor and control the electric power distribution network, ensuring that the distribution system operates safely and within its technical limitations. This interaction involves the *DSO operator*, an external user who represents the DSO and is responsible for managing the DSS.

4.2. Container View

Figure 2 depicts the container diagram of O-V2X-MP. According to the C4 model, a container represents an application or a data store. It is a software that runs independently and needs to be running for the whole system to work as intended. Note that the C4 model definition clearly highlights that its containers should not be confused with Docker containers (even though the latter share many similarities with the former). Therefore, Figure 2 depicts the containers based on the C4 model approach.

[5] eMSP stands for eMobility Managed Service Provider.

The containers of O-V2X-MP are distinguished into two categories: (i) the essential ones, which comprise the core of the O-V2X-MP system that is responsible for managing the EV charging stations, and (ii) the auxiliary ones, which facilitate the operation of O-V2X-MP and provide management options to the Platform Manager and the CPO user.

The **essential containers** are the following:

- *OCPP Server*: This service handles all the connections with the EV charging stations. It is the main interface of O-V2X-MP with the EV charging stations, allowing O-V2X-MP to get the real-time status from the EV charging stations as well as to issue OCPP commands on them. Two are the major technologies implementing this service:
  1. the Python Sanic framework[6], which is a versatile and robust HTTP2/WebSocket webserver, and
  2. the mobilityhouse/ocpp[7] Python package, which implements the OCPP 1.6 and 2.0.1 protocols.

- *O-V2X-MP Core*: It provides all the web-based services to the O-V2X-MP users. It stores the state of the system, in terms of EV charging stations and EV users, and establishes the associations/relationships between the various entities of the system, in terms of Django models. It is implemented with the Python Django framework[8].

- *Task Manager*: It is responsible for executing asynchronous tasks, e.g., for issuing requests to the EV charging stations and for getting the results. The tasks are submitted through the O-V2X-MP core. The technology of this service is based on the Python Celery framework[9].

- *Relational Database*: It provides permanent storage of the O-V2X-MP Django models, mainly related to the associations/relationships behind the O-V2X-MP core entities. The enabling technology is PostgreSQL [10].

- *Internal Message Broker*: Based on Redis[11], this service enables the Django backend to create and feed WebSocket endpoints, which can be consumed by the platform's frontend or third-party applications.

The following containers provide **auxiliary services**:

- *Database* Manager: Based on pgAdmin[12], this is a web-based GUI that allows the Platform Manager to inspect the contents of and administer the database, troubleshooting, or correcting possible issues and inconsistencies.

- *Docker Manager*: Based on Portainer[13], this is a web-based GUI that allows the Platform Manager to inspect and manage

[6] https://sanic.dev/en
[7] https://github.com/mobilityhouse/ocpp
[8] https://www.djangoproject.com
[9] https://github.com/celery/celery
[10] https://www.postgresql.org
[11] https://redis.io
[12] https://www.pgadmin.org
[13] https://www.portainer.io

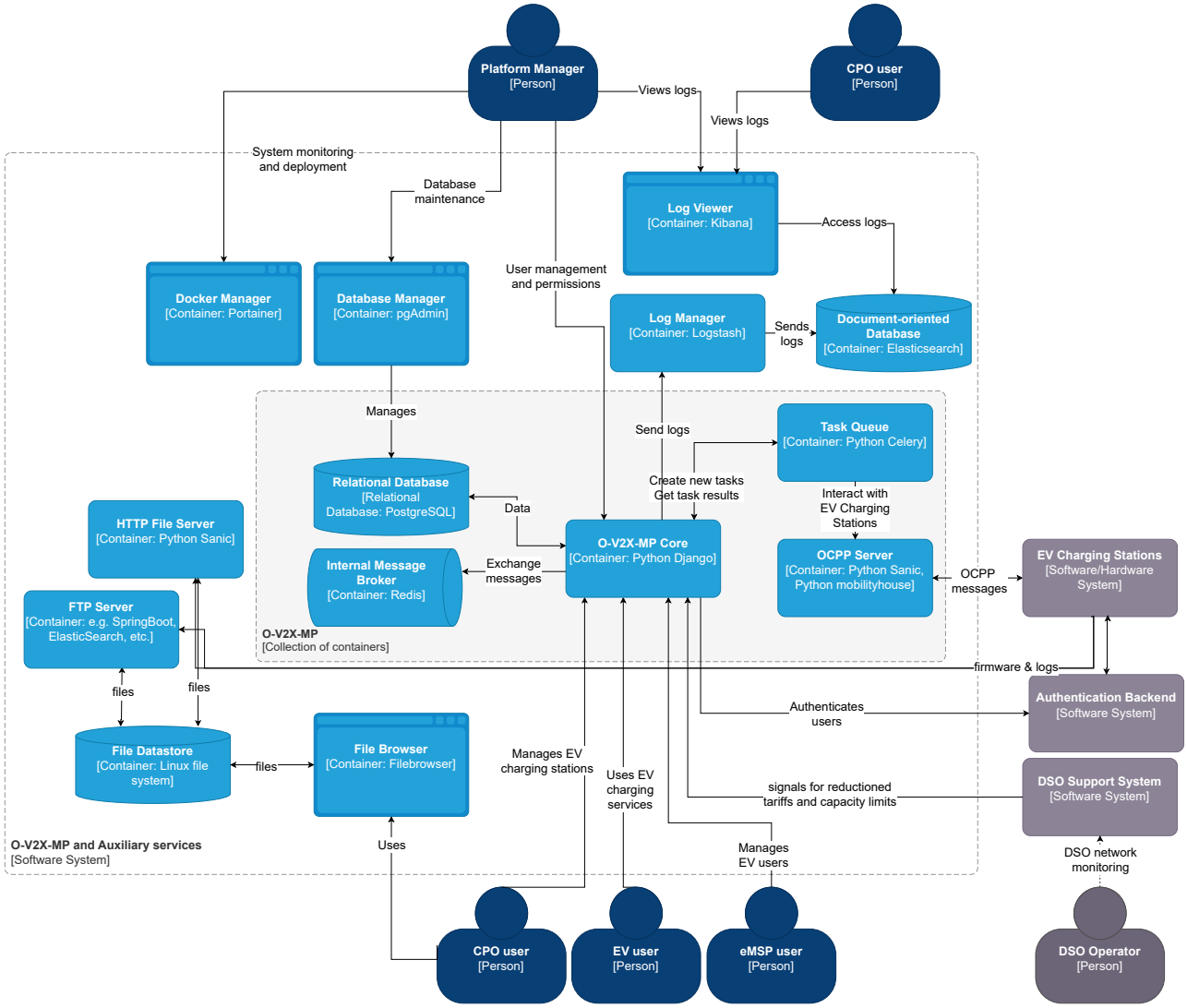

Figure 2: The Container diagram of the O-V2X-MP core and its auxiliary services.

the Docker containers comprising O-V2X-MP, supposing that the containers are deployed using Docker. Through the GUI, the logs of each docker container can be inspected in real-time, while individual docker containers can be stopped and started on demand.

- *File Datastore*: This is a file storage to be utilised by the EV charging stations. Through the HTTP file server or the FTP server, an EV charging station can upload log files or download a specific firmware. Regardless of the protocol (HTTP or FTP), this is the single storage that the EV charging stations can access.

- *HTTP File server*: Allows EV charging stations to upload files via HTTP POST requests or download files via HTTP GET requests. By default, basic user authentication is applied, with the EV charging station providing the correct credentials in order to perform any request. This container has been implemented using the Python Sanic framework.

- *FTP Server*: Allows EV charging stations to upload or download files via FTP. This container has been implemented using a custom Python script.

- *File Browser*: Based on the open-source FileBrowser project[14], it allows the CPO user to gain user-friendly access to the contents of the file datastore through a dedicated web-based GUI. Through this container, the CPO user can easily upload a new firmware file that should be downloaded by the EV charging stations using HTTP or FTP. Moreover, the CPO user can easily download a log file uploaded by an EV charging station via FTP or HTTP.

- *Log Manager*: Based on Logstash[15], this service processes the system logs sent by O-V2X-MP (e.g., Django logs, OCPP messages) and dispatches them to multiple destinations.

[14] `https://github.com/filebrowser/filebrowser`
[15] `https://www.elastic.co/logstash`

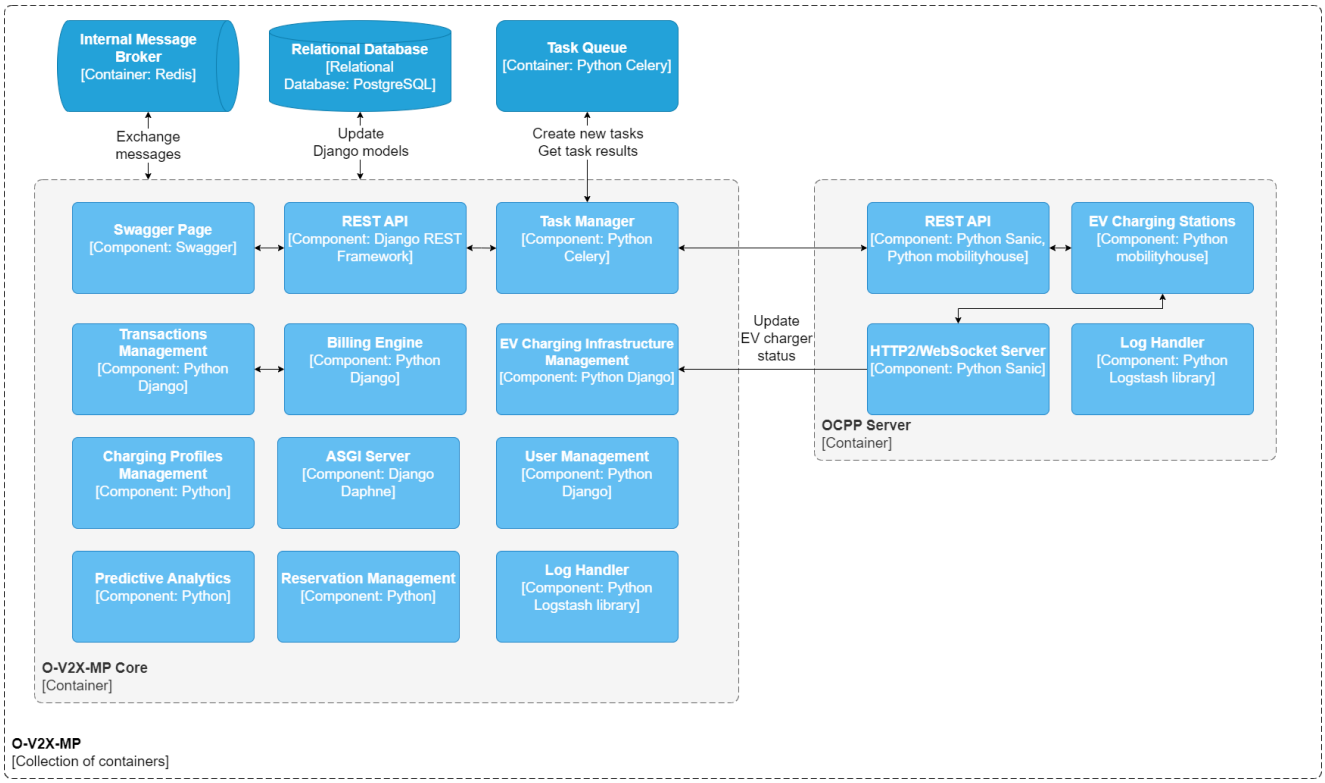

Figure 3: The Component diagram of the O-V2X-MP platform.

- *Document-oriented Database*: Based on Elasticsearch[16], this database provides permanent storage for the platform logs.
- *Log Viewer*: Based on Kibana[17], this is a web-based GUI application that visualises the contents of the document-oriented database storing the O-V2X-MP logs. Through custom visualisations, the Platform Manager and the CPO user can observe statistics about the O-V2X-MP operation, inspect the logs and detect possible issues on the EV charging stations or on the O-V2X-MP operation.

4.3. Component View

Figure 3 depicts the component diagram of O-V2X-MP, analysing its two major containers: the O-V2X-MP Core (on the left) and the OCPP Server (on the right) – the auxiliary services discussed in Section 4.2 are not analysed with component diagrams, given that they are used without any further changes.

The **OCPP server** consists of the following components:

- *HTTP2/WebSocket server*: This is the entry point of the OCPP server. When a new HTTP request comes in, specifying the OCPP 1.6 or 2.0.1 subprotocols, the server creates a new Python mobilityhouse ChargePoint16 or ChargePoint201 object, respectively, by using the Python WebSocket object that corresponds to the new connection. The new object is saved into a Python list, which can be accessed at anytime by the rest of the OCPP server components.
- *EV Charging Stations*: This component refers to the Python list that contains the objects representing the EV charging stations of the mobilityhouse library. The classes of those objects (ChargePoint16 and ChargePoint201) inherit the ocpp.v16.ChargePoint and the ocpp.v201.ChargePoint classes respectively. The Python lists are attached to the ctx object of the Sanic app, and thus they can be globally accessed at anytime by other code blocks of the process. The ChargePoint16 and ChargePoint201 subclasses implement the custom logic of O-V2X-MP for handling incoming OCPP messages and passing arguments to OCPP requests initiated by the CSMS. For example, the objects of those classes authenticate an RFID tag when an OCPP 1.6 Authorize message is received and, depending on the outcome, they send an "Accepted" or "Rejected" response. These classes are the glue between the OCPP server and the Django-based O-V2X-MP core. Since the classes import the Django settings file, they can use the Django Object-Relational Mapper (ORM) to update the Django model instances, propagating the changes back to the relational database of Django. For any update concerning an EV charging station, the component uses the unique ID of the EV charging station to update the corresponding model instance of Django via the Django ORM Python interfaces.
- *REST API*: This component is a collection of REST API endpoints served by the HTTP2/WebSocket server. These endpoints allow Django to utilise CSMS-initiated OCPP commands. By using the unique ID of the targeted EV charging

[16] `https://www.elastic.co/elasticsearch`
[17] `https://www.elastic.co/kibana`

station, the Django backend sends an HTTP POST request to the endpoint of the desired OCPP command (the POST request includes all the necessary parameters of the OCPP command). The REST API of the OCPP server retrieves the corresponding Chargepoint object in the ctx list and sends the OCPP command through the correct WebSocket session.

- *Log Handler*: This component sends all Python logs, mainly including OCPP messages, to the Log Manager (Logstash).

The components of **O-V2X-MP Core** are the following:

- *ASGI Server*: Based on the Daphne web server[18], this is the entry point for the users accessing the web services of the O-V2X-MP. Daphne serves the web pages and views of Django. It also supports WebSocket, allowing Django to create WebSocket endpoints and other Python components to send dynamically new data. For example, the OCPP server can feed a WebSocket endpoint with new meterValues concerning an ongoing transaction. The WebSocket endpoints can be used by frontend components to get and visualise the live feed of data transmitted by the backend components.

- *User Management*: This component encompasses all the functionalities relevant to users and permissions. Multiple categories of users are granted access the Django system, however, their access permissions are limited depending on their role. For example, a Platform Manager can access every aspect of O-V2X-MP (superuser access), an EV user can access only their transactions, etc. Moreover, this component can associate the users with external identities provided by an LDAP server or Identity Provider. For Single-Sign-On, the OpenID-Connect protocol is supported.

- *EV Charging Infrastructure Management*: This component corresponds to the Django models that represent the EV charging infrastructure. The users utilise the Django models to get the current status of the infrastructure, e.g., the available EV charging stations, their connectors and their locations. The models are automatically updated by the OCPP server.

- *Charging Profiles Management*: This component allows the creation and management of charging profiles. The charging profiles follow the OCPP 1.6 and 2.0.1 formats, and they can be applied at any time by the CPO to any connected EV charging station.

- *Transactions Management*: This component maintains all the EV charging transactions, be it ongoing, failed or completed. Moreover, it allows only users with the appropriate privileges to see the transactions. For example, the eMSP user can see all transactions, while an EV user accesses only their own transactions.

- *Billing Engine*: This component maintains the tariffs and calculates the Charge Detail Record (**CDR**) of a transaction. When a transaction is finished, the Billing Engine processes it and generates the corresponding CDR, which is used to bill the eMSP for using the EV charging station. The tariffs allow the nuanced assignment of cost rates, such as utilizing the AC cost rate for AC charging connections and the DC charging rate for DC charging connections. This dynamic tariff management structure facilitates a flexible and adaptable cost framework. O-V2X-MP can create cost rates for the following categories:

  - One-time charging session fee.
  - Consumption-based fee for the charging session (price per kilowatt-hour, kWh).
  - Time-based fee for the charging session.
  - Combinations of the aforementioned fees.

  The CDR encapsulates the details of a concluded charging session and stands as the sole billing-relevant object. A CDR is the cumulative Session object combined with pertinent tariff information. Following a successful charging session, the CPO transmits a CDR to the eMSP associated with the registered driver. The eMSP subsequently collects the payment from the driver and remits the difference to the CPO.

- *Reservation Management*: This component manages the reservations created by the EV users, who book a specific EV charging station for a specific timeframe in the future. Each time a user creates a reservation, a corresponding OCPP command is issued to the appropriate EV charging station, and a Django model instance is created.

- *REST API*: This component provides access to all Django models and available OCPP commands, through a RESTful API that is publicly exposed. Its purpose is to offer all the possible interactions for operating O-V2X-MP, thus decoupling the backend from the frontend. This approach allows the independent development of web-based frontend applications or mobile apps. Moreover, it is possible for third-party applications to interact with the O-V2X-MP for specific purposes, e.g., creation and installation of charging schedules by the DSS. In terms of security, this REST API is protected by JSON Web Token (JWT) authentication and authorisation. The application or user that needs access to the API, issues a JWT token by providing their username and password. Then, for all requests, they use the JWT token instead.

- *Swagger Page*: This is a web page that provides a user-friendly interface for accessing the REST API based on Swagger[19]. Every time the user accesses this page, Django dynamically generates the OpenAPI specification of the REST API, by processing the Python source code with the help of the Django REST Framework. Then, the OpenAPI specification is visualised in this page.

[18] `https://github.com/django/daphne`

[19] `https://swagger.io`

- *Task Manager*: This component is responsible for managing asynchronous tasks. When an asynchronous task concerning the execution of an OCPP command is submitted through the REST API, the task manager submits the task through the Python Celery framework. The component monitors the progress of the task and returns the results by updating the Task instance (Django model).
- *Predictive Analytics*: This component trains and applies AI pipelines to the data stored in the Relational Database of O-V2X-MP (see Figure 2) to extract insights from the registered users and EV chargers. As an example, consider the task examined in Section 6.2, where the goal is to forecast the future energy consumption of an EV charger. The component receives the options about the preferred AI model and the ID of the EV charger. Then, pre-defined Django queries are executed to retrieve recent historical data, on which the model is trained. As output, the component provides its forecasts.
- *Log Handler*: This component sends asynchronously all Python logs to Logstash, utilising the `python-logstash-async` Python library[20].

5. Frontend

The O-V2X-MP dashboard is a dynamic and multifunctional web application that leverages the Node.js runtime environment[21] along with the Vue.js JavaScript framework[22]. These frameworks were combined through the VRISTO template[23]. To cover all functional requirements, the dashboard incorporates the following libraries:

- Leaflet[24] for rendering maps,
- Leaflet Routing Machine[25] for adding routing capabilities,
- Windy[26] to display weather information on the map,
- Axios[27] for handling HTTP GET/POST requests, and
- the AES library from CryptoJSf[28] for encrypting sensitive content in GET/POST methods using Axios.

By integrating these libraries and frameworks, the O-V2X-MP dashboard ensures a robust, secure, and user-friendly experience for two types of accounts: (i) the **EV drivers**, and (ii) the **CPOs**. For short, the former accounts are simply called **users** and the latter **administrators**. In this context, the functional requirements are distinguished into three categories: (i) those targeting both user types, (ii) those applying exclusively to EV drivers, and (iii) those crafted for CPOs. We elaborate on each category in the following.

[20] `https://pypi.org/project/python-logstash-async`
[21] `https://nodejs.org`
[22] `https://vuejs.org`
[23] `https://vristo.sbthemes.com`
[24] `https://leafletjs.com`
[25] `https://www.liedman.net/leaflet-routing-machine`
[26] `https://www.windy.com/`
[27] `https://axios-http.com`
[28] `https://www.npmjs.com/package/crypto-js`

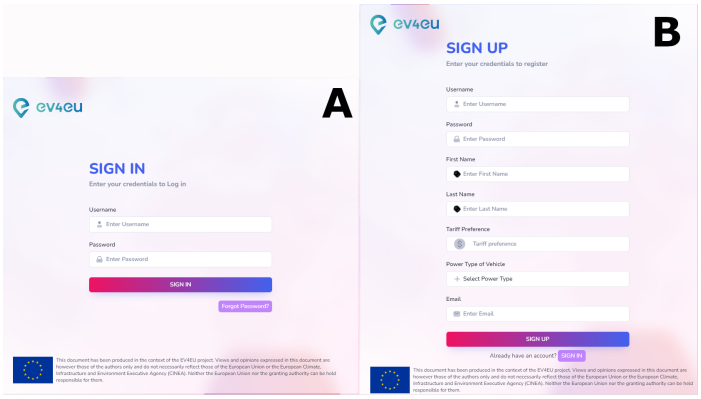

Figure 4: The platform's (A) sign in, and (B) sign up forms.

5.1. Generic functionalities

The following operations are common to both administrators and common users:

- *Account authentication*: The basic login functionality for accessing the platform. Password reset is supported. The corresponding menu is shown in Figure 4A.
- *Charging station overview*: Any account can view and select any of the available charging stations to inspect its precise location and technical characteristics (i.e., number, type and nominal power of sockets). Quite crucial are also the following two aspects:

  1. The *tariff schedule*, which determines the overall charging cost. Tariffs are dynamic, with a different value for each hour within the day, because they depend on the power generation of renewables (RES). In fact, in demand response scenarios, the objective is to promote green charging, aligning the charging load of EVs with RES power generation by lowering the tariffs accordingly.
  2. The *capacity schedule*, which determines the power that can be delivered to an EV per hour of the day. Note that the DSO can save capacity for other services by procuring and activating contracts with the CPO that reduce the capacity of specific charging stations for the day ahead or even in real-time (with a pre-determined cost). This way, the DSO reduces the constraints on the distribution system in cases with excessively high loads. For example, this might involve halving the power of EV chargers (e.g., from 22kW to 11kW), which inevitably doubles the duration of the charging sessions.

  The O-V2X-MP GUI provides this information by devoting the central, largest part of the screen to a map, centralized to the location of the user, as it is determined by the (approximate) information provided by the browser based on the client IP. See Figure 5A for an example. Note that every charger near this location is indicated through a marker, whose colour indicates its status: green denotes active charging stations, orange indicates charging stations that are unavailable, i.e., occupied by another EV driver, and red stands for charging stations that are out of order or out of capacity.

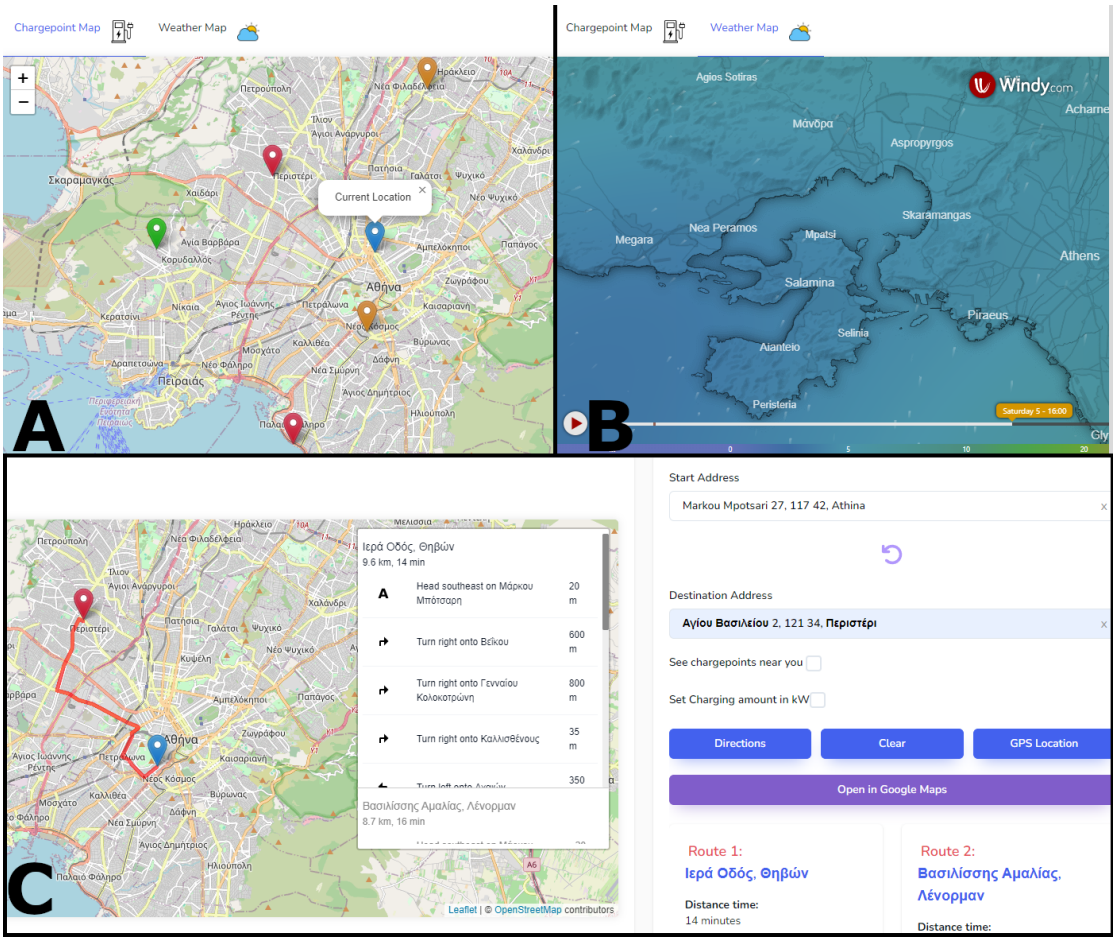

Figure 5: The O-V2X-MP dashboard's (A) main menu, (B) weather prediction, and (C) routing instructions.

- *Weather prediction*: The map visualizing the location of the charging stations can be enriched with a prediction of weather conditions, as shown in Figure 5B. These conditions, which are extracted for free from Windy[29], are critical, because they typically affect the capacity of the battery and, thus, the charging time.

These generic operations complement the type-specific ones described below, ensuring a holistic set of operations for both user types.

5.2. *User functionalities*

The main operations offered to EV drivers are the following:

- *Sign Up*: Users can easily register to the platform for free. To minimize the volume of sensitive information that is processed by the O-V2X-MP platform, the registration requires only the following information from the user, as shown in Figure 4B: (i) e-mail address, (ii) password, (iii) tariff preference, which indicates the maximum charging cost that the EV driver is willing to pay (i.e., chargers with a higher cost are automatically excluded from the charging point overview), and (iv) EV power type, which indicates the charging connector type of the user's EV (e.g., CCS2 or CHAdeMo). Optionally, the users can also provide their first and last names. Even though no personal information is required, the user content is stored in encrypted form by the User Management Component (cf. Section 4.2), using the AES standard [9], which offers high levels of security.

- *User preferences*: Users can update the main personal preferences that were filled in during registration, i.e., the tariff preference and the EV power type (see Figure 6A). In general, all user information can be updated, except for the email, which acts as the unique identifier of user accounts.

- *Routing Options*: Users can navigate to the most suitable charging station. Three characteristics of the available chargers determine this decision:

  1. their tariffs,
  2. their capacity, and
  3. their location.

  The first two characteristics are provided through the charging station overview presented in Section 5.1. For the third

[29] `https://www.windy.com/`

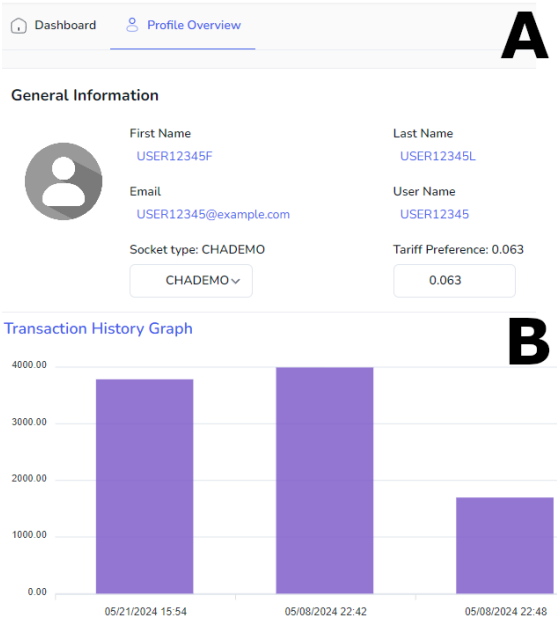

Figure 6: (A) Updating user information, and (B) visualizing user charging statistics.

characteristic, the O-V2X-MP dashboard offers routing instructions between any two points indicated by the user: the start location can be selected either automatically, through the location lookup based on the IP address, or manually, by clicking on the desired location. Similarly, the destination location is selected by a subsequent click on the desired location that does not necessarily correspond to an EV charger. Alternatively, the user can type in an exact address in the Start and Destination fields respectively. As shown in Figure 5C, the routing instructions are quite detailed, including the overall distance and the estimated travelling time, thus facilitating users to plan their trip. Live routing instructions can also be provided through Google Maps: after selecting the start and destination locations, the user simply presses the "*Open in Google Maps*" button, which initiates the GPS-based navigation.

- *User history*: Users can observe their charging statistics through intuitive diagrams, such as those in Figure 6B. Charging history is instrumental for EV drivers that rely on public charging stations with varying prices. By monitoring their charging costs, they can budget for their mobility expenses and compare the costs across different stations or times of day in order to identify the most affordable options. Understanding when and where charging is most cost-effective can help drivers to optimize their charging schedules. A detailed charging history also facilitates EV drivers to identify the root cause of charging issues that may arise, e.g., due to a faulty charger, a problem with their EV or their batteries, or even an error in the charging settings.

All user interactions are highlighted in Figure 7.

5.3. Administrator functionalities

The main operations offered to CPOs are the following:

1. *Charging station management*: CPOs can view, modify, add and delete entire charging stations as well as individual sockets (i.e., connectors). The information about the available charging points is automatically updated whenever a new EV charger connects to the OCPP server. For safety reasons, this can be manually carried out, too. To this end, the O-V2X-MP dashboard allows for updating all fields describing a particular EV charger. The fields are organized into two groups: (i) those pertaining to the location of the charger, shown in Figure 8A, and (ii) those pertaining to its technical characteristics, shown in Figure 8B. The administrator can alter any of these fields.

2. *Real-time monitoring*: CPOs need to continuously monitor the status of the charging stations and their charging sessions in order to ensure their availability and uptime, preventing downtimes that frustrate EV drivers and loose revenues for the charging station owners. Monitoring also helps CPOs to identify and address issues with charging speeds, power delivery, and overall efficiency with the aim of improving the experience of EV drivers. Indeed, customer satisfaction and loyalty can be enhanced by quickly identifying and resolving issues faced by users (like charging errors, network problems, and payment failures). Finally, monitoring contributes to the detection and prevention of cyberattacks, protecting user data and financial information. An example of the monitoring diagrams provided by the O-V2X-MP platform is depicted in Figure 8C.

3. *Tariff management*: CPOs can adjust tariffs in real-time, a crucial operation in the context of smart charging infrastructure that relies on variable tariffs. By specifying the tariffs for particular charging stations and regions, as shown in Figure 9, CPOs can examine the impact of charging cost on the behaviour of EV drivers. This is crucial for attracting more users and maximizing revenue.

4. *DSO integration*: CPOs are responsible for maintaining a real-time connection with DSO support systems to implement the smart charging scenarios on demand. For example, in case there is increased solar or wind power generation in a specific area that exceeds the corresponding demand, the DSS can notify the O-V2X-MP platform to lower the tariffs in nearby charging stations in order to maximize the green energy consumption.

5. *User management*: CPOs can create, view, update and delete user accounts.

6. *Data Analytics*: CPOs have access to machine and deep learning pipelines that process the data gathered from the charging sessions in order to make predictions, facilitating decision making. As an example, consider Figure 10, which demonstrates the prediction for the power delivered (in KWh) by a specific charger (Charger 4 in Table 2) per

Figure 7: Summary of the main user functionalities: (A) log in, (B) search for the nearest EV chargers, (C) get directions for the selected charger, (D) start charging, after having plugged the EV, and (E) complete the charging session.

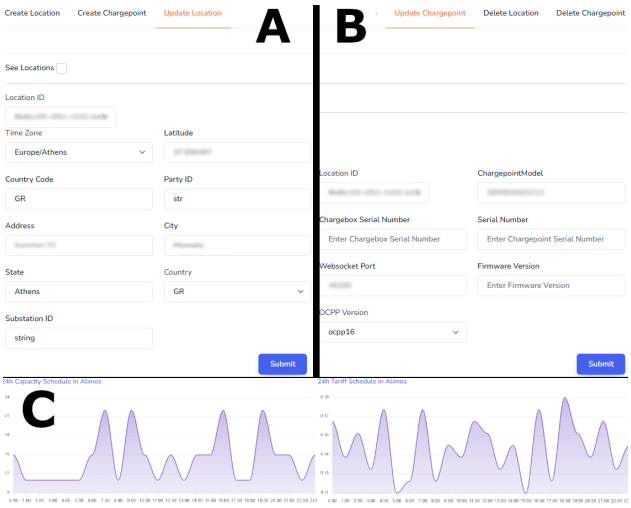

Figure 8: (A) Updating the location information of a specific charger, (B) updating the technical characteristics of a specific charger, and (C) monitoring the power to be delivered by a specific charger.

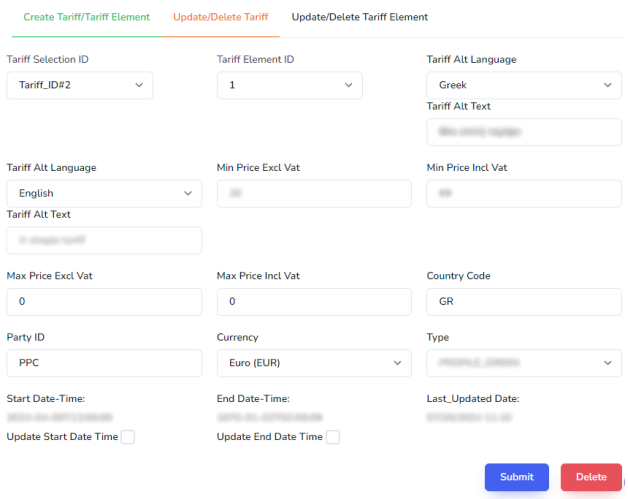

Figure 9: Menu for tariff declaration by CPOs.

day in January 2024, using as training data all its past charging sessions (i.e., up to December 31st, 2023). Note that the blue line indicates the actual values measured for this period, thus illustrating the high accuracy of our LSTM-based prediction model (see Section 6.2 for more details).

These functionalities are designed to facilitate the efficient management and maintenance of the O-V2X-MP backend through a reliable and user-friendly interface. Note that there is no interface for registering administrators, unlike the sign-up form for end users, because CPOs are added only through the O-V2X-MP backend, for security reasons.

6. Experimental Analysis

We now demonstrate the capabilities of the data analytics module of the O-V2X-MP platform. We actually examine two tasks: (i) the clustering of EV user profiles in Section 6.1, and (ii) the prediction of load per EV charger in Section 6.2.

Dataset. In both tasks, we employ the following four datasets: (i) D_{PPC} comprises all charging sessions from June 2022 to January 2024 in a network of 1,116 EV chargers throughout Greece, owned by Public Power Corporation. (ii) D_{PSDN} collects data from a research institution located in Pasadena, California, which has installed 55 EV chargers in a campus garage open to the public, with faculty, staff, and students accounting for most of the usage [10]. (iii) D_{LCND} gathers

Table 1: Dataset with real charging sessions.

	Original D_{PPC}	Clean D_{PPC}	D_{PSDN}	D_{LCND}	D_{SV}
Time period	May 31, 2022 - January 31, 2024		April 25, 2018 - September 14, 2021	September 5, 2018 - September 14, 2021	March 3, 2019 - September 14, 2021
#chargers	1,116	1,063	55	52	8
#municipalities	234	198	N/A	N/A	N/A
#charging sessions	131,535	131,186	31,424	33,640	1,683
Overall load consumed	2.021 GWh	2.015 GWh	0.291 GWh	0.473 GWh	0.029 GWh
Overall duration	268,060 hrs	250,388 hrs	484,680 hrs	568,008 hrs	34,992 hrs

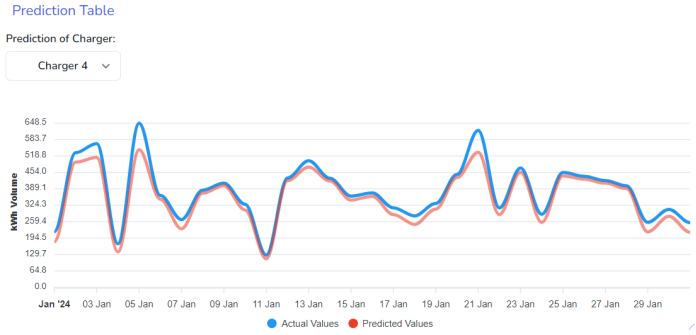

Figure 10: Delivered power (in KWh) per day in January 2024, both in reality (blue line) and as a forecast (orange line).

data from a national research lab located in La Canada, California, with 52 EV chargers available exclusively to employees [10]. (iv) D_{SV} consists of data from an office building in the Silicon Valley area, where 8 EV chargers are used solely by employees [10]. The technical aspects of these datasets are summarized in Table 1.

To ensure the robustness and reliability of our analysis, we cleaned the data by removing the chargers that were operational for a single day. We also remove the transaction records with missing, noisy or outlier values. As such, we consider the records where the delivered power is higher than that of the corresponding EV charger or higher than the maximum battery capacity in EVs sold in Europeand the USA. These steps prevent skewed results caused by outliers or incomplete data. For D_{PPC}, the cleaning reduced the dataset to 1,063 chargers with reliable and representative data for our experimental study. This cleaning had a minor impact on the number of charging sessions, on the overall duration and on the consumed load, as shown in the third column of Table 1. There was no change in the other datasets, given that they were curated before their release [10].

6.1. EV profiles clustering

6.1.1. Problem Definition

A useful value-added service offered by the O-V2X-MP platform is the support for running data analytics on its transaction data to gain insights into the behavior of EV users. This is illustrated through the unsupervised task of clustering charging transactions into user profiles.

More formally, this task can be defined as follows:

Problem 1 (User Profile Clustering). *Given the set of charging sessions gathered by O-V2X-MP, group them into a limited set of balanced, disjoint clusters with interpretable behavior.*

The above definition aims to avoid two kinds of results:

1. A large set of cohesive clusters, which are hard to comprehend, due to the minor differences in their feature/behavior.

2. A limited set of cohesive clusters, where one of them contains the vast majority (i.e., $\gg$90%) of the charging sessions, dominating all others.

Both results are common when using common effectiveness measures (see below), but provide no insights into the behavioral patterns of EV users. Instead, the resulting profiles should be relatively balanced, with a substantial portion of charging sessions per cluster, while involving a significant deviation in the considered features, and while being interpretable, so that they allow for an a-priori prediction of the power demand throughout the network of EV chargers maintained by the CPO.

Evaluation Measures. To assess the effectiveness of the resulting clusters, we use the following two measures [11]:

- For the *Davies-Bouldin index*, smaller values, closer to zero, mean better performance, as clusters are well separated and each one is well represented by its centroid.

- The *Silhouette Coefficient* takes values from -1 to $+1$, with higher values indicating more precise clustering, i.e., most records are well matched to their own clusters and poorly matched to neighboring ones.

6.1.2. Solution

To address Problem 1, O-V2X-MP implements a Machine Learning pipeline that leverages the Python data science ecosystem, comprising the following steps:

- *Data cleaning* excludes transactions with missing, noisy or outlier values, as described above.

- *Feature engineering* defines the following attributes to be used by the clustering algorithms [12]:

  F1) the timestamp of the plug-in time, also termed "connect time"

  F2) the timestamp of the plug-out time, also termed "disconnect time"

  F3) the total plug-in duration, i.e., $F2 - F1$, the time the EV was parked and plugged into the EVSE (also called sojourn time),

  F4) the total volume of delivered power (in KWh),

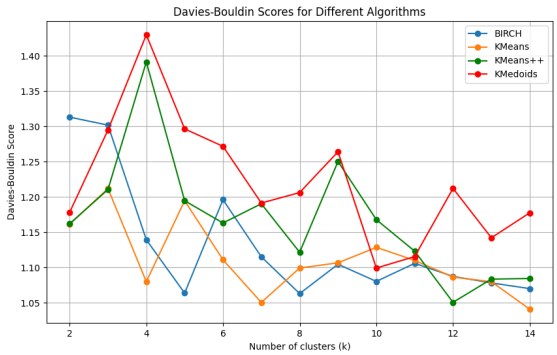

Figure 11: Davies-Bouldin index scores per clustering algorithm over D_{PPC} for different number of clusters (k).

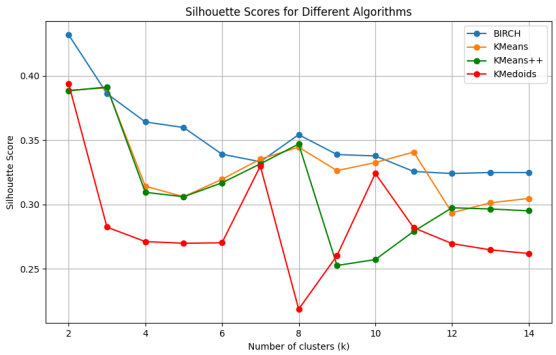

Figure 12: Silhouette scores per clustering algorithm over D_{PPC} for different number of clusters (k).

F5) the idle time, when the EV was parked and plugged into the charging station, without consuming energy, i.e., $F3 - F6$.

F6) the charging time, which is indirectly computed in the case of D_{PPC} as $\frac{\text{Energy Delivered}}{\text{max\_Power\_per\_EVSE}\times 0.8}$, while being directly estimated in the case of the other datasets, based on the timestamp at the end of charging,

F7) max P per EVSE (kW), the maximum power capacity of the EVSEs (this is available only in D_{PPC}).

Note that we disregard features that do not contribute to the detection of patterns in EV user behavior like user names or ids. Note also that we applied *min-max normalization* to all the features, restricting them to the [0-1] range.

- *Clustering* is applied to the resulting dataset to split it into disjoint clusters. We opted for the following four clustering algorithms, implemented by scikit-learn[30], version 1.3.2:

A1) K-Means [13] randomly selects an initial set of cluster centers, assigns each instance to its closest center and iteratively updates the cluster centers and the cluster assignments until convergence. It is the top performer algorithm for this task in [12].

A2) K-Means++ [14] selects better initial cluster centers, improving the initialization step before proceeding with the standard K-Means optimization iterations.

A3) K-Medoids [15] enhances K-Means by using actual points as cluster centers, instead of the mean of the cluster instances. It is more robust to noise and outliers, minimizing the sum of dissimilarities rather than squared distances.

A4) Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [16] is a Hierarchical clustering algorithm designed to efficiently handle large datasets. It builds a tree structure that summarizes the data points, allowing for quick, hierarchical clustering.

[30] `https://scikit-learn.org`

Note that the functionality of the above algorithms is quite diverse, but is solely configured by k, i.e., the number of the final clusters, which is given as input. Therefore, we optimize their performance through a grid search for the best k. Note also that for all other configuration parameters, we employ the default values provided by the scikit-learn implementation. The only exception is that we set the threshold of the Birch algorithm to 0.2 – this parameter dictates that the radius of the subcluster obtained by merging a new sample and the closest subcluster should be less than the threshold, otherwise a new subcluster is started. Finally, it is worth stressing that in datasets with a high number of charging sessions, the KMedoids algorithm is applied to a random sample of 25,000 instances, due to memory limitations.

6.1.3. Experimental Results

In the following, we applied our pipeline through the O-V2X-MP platform on the datasets of Table 1, using grid search to consider all values for k in the range of $[2, 15]$ – we omit D_{PSDN}, due to the quite unstable behavior with respect to k that is exhibited by all clustering algorithms.

D_{PPC}. Figure 11 reports the Davies-Bouldin index.

KMedoids and KMeans++ generally yield larger scores than the other methods, particularly for smaller k values, suggesting relatively poor clustering performance. In contrast, KMeans and BIRCH retain lower Davies-Bouldin scores for most cluster sizes, while showing a more steady decreasing trend as k rises.

The Silhouette Coefficient in Figure 12 verifies these results, with BIRCH achieving the greatest silhouette scores for smaller cluster counts (2-4), while having the most stable trend and the best average score. BIRCH is followed by KMeans and KMeans++, which show very similar performance with each other and perform relatively well for most k values. KMedoids exhibits again the lowest silhouette scores overall, especially at $k = 8$, when its performance significantly declines.

Overall, we can conclude that BIRCH typically outperforms the other clustering algorithms, with KMeans following in close distance. It is worth gaining insights into its best clusters, i.e., those defined for $k = 4$. Apart from their high Silhouette Coefficient and their low Davies-Bouldin index, another advantage

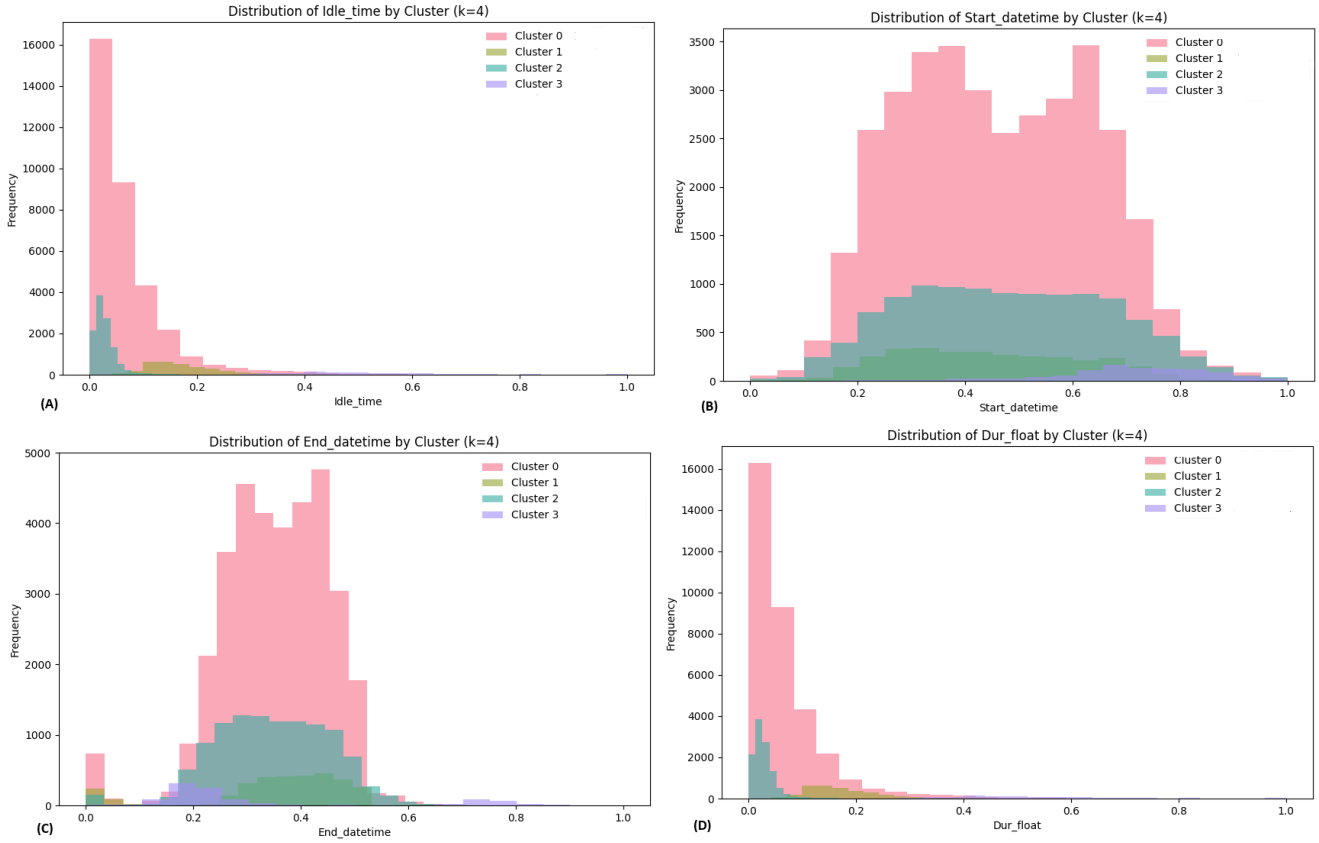

Figure 13: The distribution of charging sessions belonging to the four clusters created by BIRCH for $k = 4$ in D_{PPC} with respect to: (a) the idle time – feature F5, (b) the starting time – feature F1, (c) the ending time – feature F2 and (d) the duration of charging sessions – feature F6.

is that their number is low, while the distribution of sessions among them is not extremely skewed: the cluster sizes range from 2.0% and 6.4% for Clusters 3 and 1, respectively, to 22.4% and 69.2% for Clusters 2 and 0, respectively. Most importantly, these clusters are interpretable, as shown in Figure 13.

Starting with Figure 13(a), we observe that Cluster 2 exhibits very low idle time (feature F5), Clusters 0 and 1 much larger idle times, and Cluster 3 the longest by far idle times. Figure 13(b) shows the distribution for feature F1. We observe that all clusters typically charge their EVs during daytime, except for Cluster 3 whose charging sessions typically start late in the day. Similar patterns appear in Figure 13(c), which corresponds to feature F2, i.e., the end time of charging sessions. The only exception is that most sessions of Cluster 3 terminate early in the morning, which indicates overnight charging, while a small part of the sessions belonging to the other clusters terminate at midnight. Regarding the duration of the charging sessions (i.e., Feature F6), Figure 13(d) shows that the shortest correspond to Cluster 2, followed by Cluster 0 and 1, with Cluster 3 corresponding to the largest by longest charging sessions, as expected. The same patterns correspond to the total plug-in duration and, thus, the distribution for feature F3 is omitted for brevity. No clear patterns appear in the other two features, which are also omitted for brevity.

On the whole, we can conclude that Cluster 3 corresponds to long, typically overnight charging sessions, with high idle, in contrast to Cluster 2, which includes very short charging sessions during daytime, with almost no idle time (e.g., during shopping). Cluster 0 involves longer charging sessions with higher idle time during daytime, probably during working hours. Cluster 1 pertains to longer charging sessions with more idle time later in the day, e.g., during post-work activities.

D_{LCND}. Figure 15 reports the Davies-Bouldin index. We observe that BIRCH consistently obtains the lowest scores and, thus, the highest effectiveness for $k > 4$. KMeans performs better only for the lower number of clusters, i.e., $k \in [2, 4]$. KMeans++ is quite unstable for $k \leq 6$, exhibiting the highest scores in most of these cases. It stabilizes, though, as the number of clusters increases, exhibiting scores that are close to KMeans, but lower than KMedoids. Finally, KMedoids stabilizes to relatively high scores from $k = 4$ on, which eventually become the lowest ones after $k = 7$. This indicates that it creates least cohesive clusters among all algorithms.

Similar patterns arise from the Silhouette Coefficient in Figure 16. BIRCH exhibits again the highest effectiveness for $k \geq 4$, with the best score achieved for $k = 4$. KMeans also performs well, especially for $k \in [4, 10]$, where it occasionally produces scores comparable to BIRCH (e.g. for $k = 8$), exceeding KMeans++ and KMedoids. The scores of KMeans++ scores fluctuate significantly, particularly for lower k values, indicating less consistent performance; for $k > 6$, its scores slightly

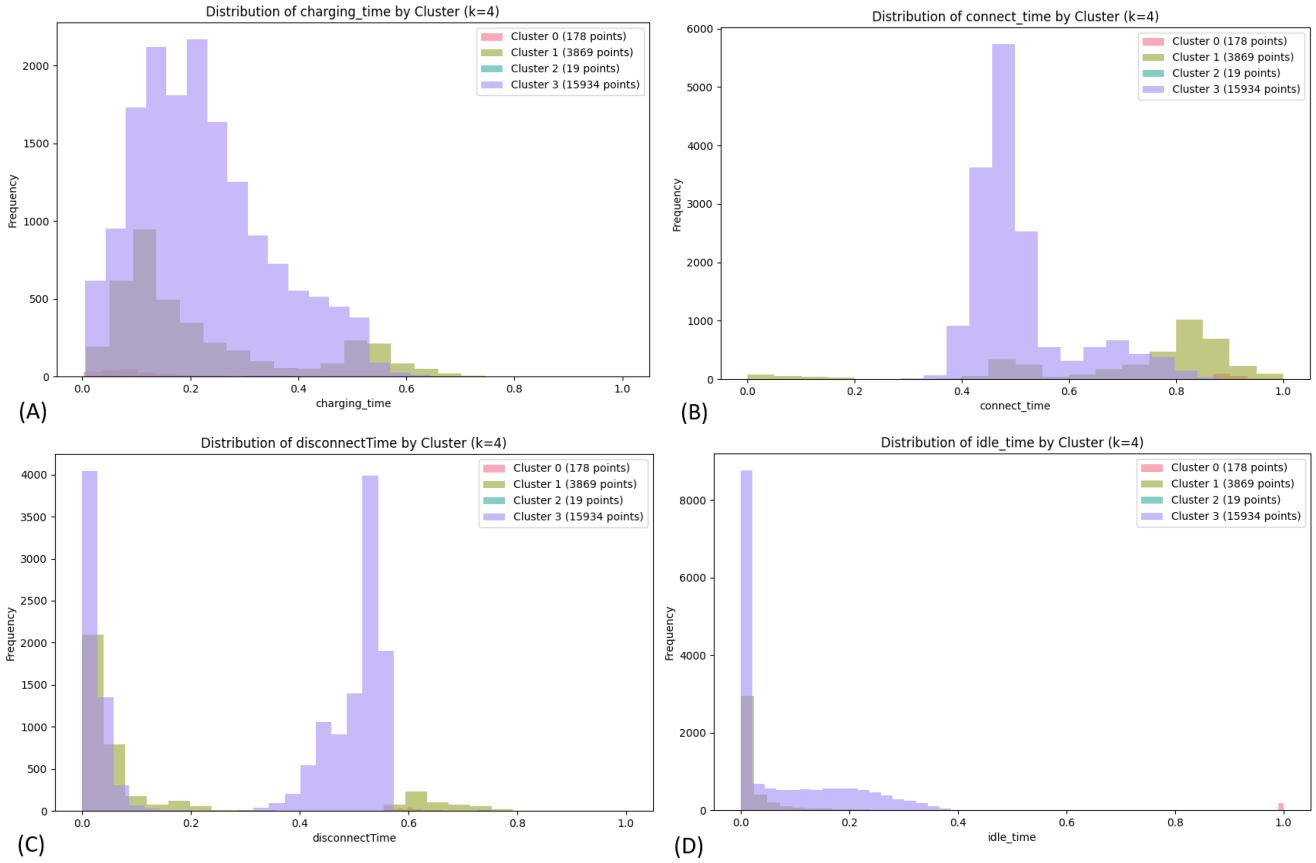

Figure 14: The distribution of charging sessions belonging to the four clusters created by BIRCH for $k = 4$ in D_{LCND} dataset, with respect to: (a) the charging time – feature F6, (b) the "connect time" – feature F1, (c) the "disconnect time" – feature F2, (d) the idle time – feature F5.

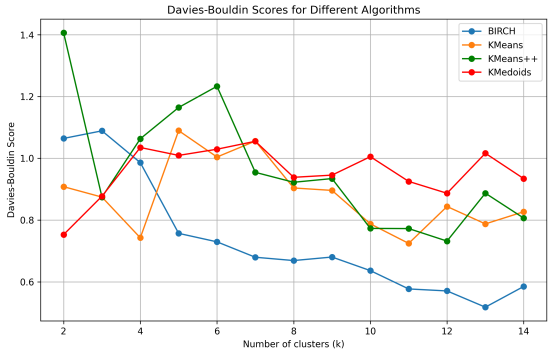

Figure 15: Davies-Bouldin index scores per clustering algorithm over D_{LCND} for different number of clusters (k).

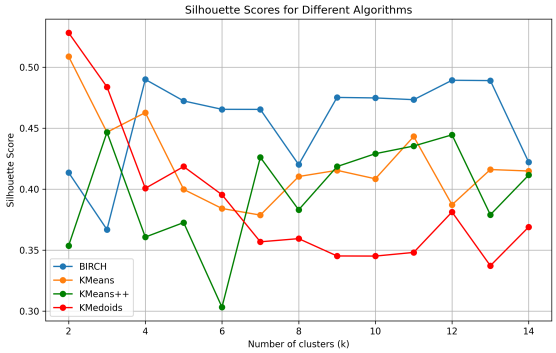

Figure 16: Silhouette scores per clustering algorithm over D_{LCND} for different number of clusters (k).

improve, obtaining competitive results at higher values. KMedoids exhibits the highest score overall (for $k = 2$), albeit for an utterly unbalanced clustering that provides no real insights. For $k > 3$, its scores drop significantly, consistently ranking last for $k \in [7, 15]$. In other words, it generates the least well-defined clusters in this dataset.

Overall, BIRCH produces the most effective clusters throughout a wide range of cluster numbers (k), exhibiting a relatively steady performance for $k \geq 4$. It is typically followed in close distance by KMeans, especially for intermediate cluster numbers. KMeans++ and KMedoids are less successful, especially for lower values of k. These patterns are consistent across both evaluation measures.

To gain more insights into the best clusters, we define as optimal configuration the BIRCH algorithm for $k = 4$. This configuration exhibits a relatively low score for the Davies-Bouldin index in combination with a very high Silhouette Coefficient, while the configurations with an even higher score produce ex-

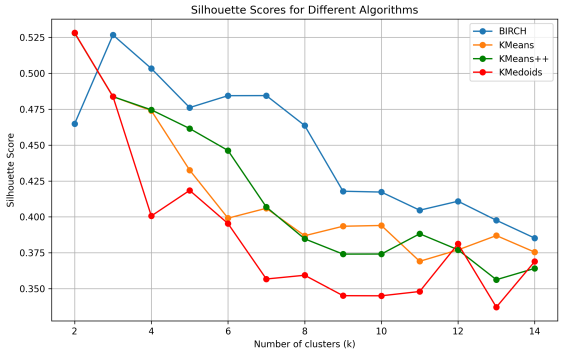

Figure 17: Silhouette scores per clustering algorithm over D_{SV} for different number of clusters (k).

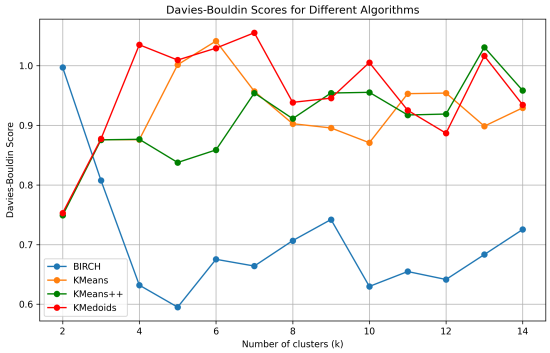

Figure 18: Davies-Bouldin index scores per clustering algorithm over D_{SV} for different number of clusters (k).

tremely skewed clusters. This is not the case with the selected configuration, since the cluster sizes range from 0.1% and 0.9% for Clusters 2 and 0, respectively, to 19.3% and 79.7% for Clusters 1 and 3, respectively. Most importantly, these clusters are interpretable, as shown in Figure 14.

We observe that for the charging time (F6), the two smaller clusters (0 and 2) correspond to rather short sessions. In contrast, the largest cluster (3) exhibits a right-skewed distribution concentrated between 0.1-0.4, indicating that most charging sessions are relatively short to medium length. The medium cluster (1) exhibits a long right tail, extended beyond 0.6, which represents less frequent but longer charging sessions. Given that the same cluster conveys a concentration around 0.1, its users reflect a mixture of "top-up" charging behavior of short duration, and occasional full charging sessions of longer durations. The right skew might also indicate that while most users need relatively quick charges, there's still a significant number who require longer charging times, possibly due to different battery capacities or starting charge levels.

Major differences between the four clusters are also observed in terms of the "connect time" (F1). The charging sessions of the largest cluster peak at 0.5, with an almost normal distribution around it. This suggests a very consistent behavior, where a large number of users tend to connect their vehicles at similar times, indicating a mid-day charging behavior (around noon). For the rest of the clusters, there's also a notable concentration around 0.8-0.9, which represents evening charging sessions.

The typically short charging sessions of the largest clusters are reflected in the "disconnect time" (F2) and the idle time (F5). The former is concentrated shortly after noon, signaling that most sessions are terminated at most a couple of hours after their initialization. That's why the idle time is zero for the almost 2/3 of these charging sessions. Nevertheless, 1/3 of the sessions terminate early in the morning, accounting for larger idle times. The other clusters exhibit lower idle times, even though most of their sessions terminate early in the morning.

In short, our approach yields two rather small clusters that correspond to outliers and two larger clusters with distinctive behavior. Most EV drivers of the larger cluster charge their EVs in short time, around noon, while the rest are more likely to charge for a long time, later in the evening.

D_{SV}. The Davies-Bouldin index is reported in Figure 18. BIRCH consistently achieves the lowest (by far) scores for almost all k values, except for $k = 2$. Its scores are particularly low at $k = 4$ and remain steadily low, indicating a rather effective separation of charging sessions into meaningful clusters. KMeans performs also well, with generally stable scores close to 1. Its lowest value also corresponds to $k = 4$, but remains much higher (and thus worse) than that of BIRCH.

KMeans++ follows a similar pattern as KMeans, but is less stable, exhibiting slightly greater fluctuation, particularly at lower values for k. KMedoids underperforms all other algorithms for most cluster numbers, exhibiting the lowest effectiveness. These patterns are verified by the Silhouette Coefficient in Figure 17. Kmeans, Kmeans++ and KMedoids achieve the same highest score for $k = 2$, which for the last two stems from a trivial, highly unbalanced clustering. For $k \geq 3$, though, BIRCH consistently outperforms them, with KMedoids ranking last in all cases, while KMeans and KMeans++ are consistently located in the middle of these two extremes.

Overall, D_{SV} shows a pattern similar to D_{PPC} and D_{LCND}: BIRCH typically outperforms the other clustering algorithms, with KMeans following in close distance, while KMeans++ and KMedoids exhibit consistently lower cluster cohesiveness.

As the optimal configuration, we designate KMeans with $k = 2$, because it yields the highest Silhouette score overall and a relatively low Davies-Bouldin index score, while producing relatively balanced clusters – the larger one involves 69% of all charging sessions. Most importantly, the resulting clusters are interpretable, as shown by the feature distribution in Figure 19.

We see that the "connect time" (F1) differs substantially, with the EV drivers of the large cluster typically starting their charging sessions around noon - shortly after 12.00. Instead, most users of the small cluster usually start charging in the afternoon or even in the evening. In terms of the delivered power (F4), the large cluster shows a strong peak near the lower end, indicating sessions with minimal energy delivery. In contrast, the smaller cluster has a broader spread, suggesting more varied energy demands. Idle time (F5) peaks at zero in the vast majority of charging sessions, especially for those of the large cluster, indicating that users disconnect quickly after charging. The smaller cluster, though, includes customers who keep their vehicles in-

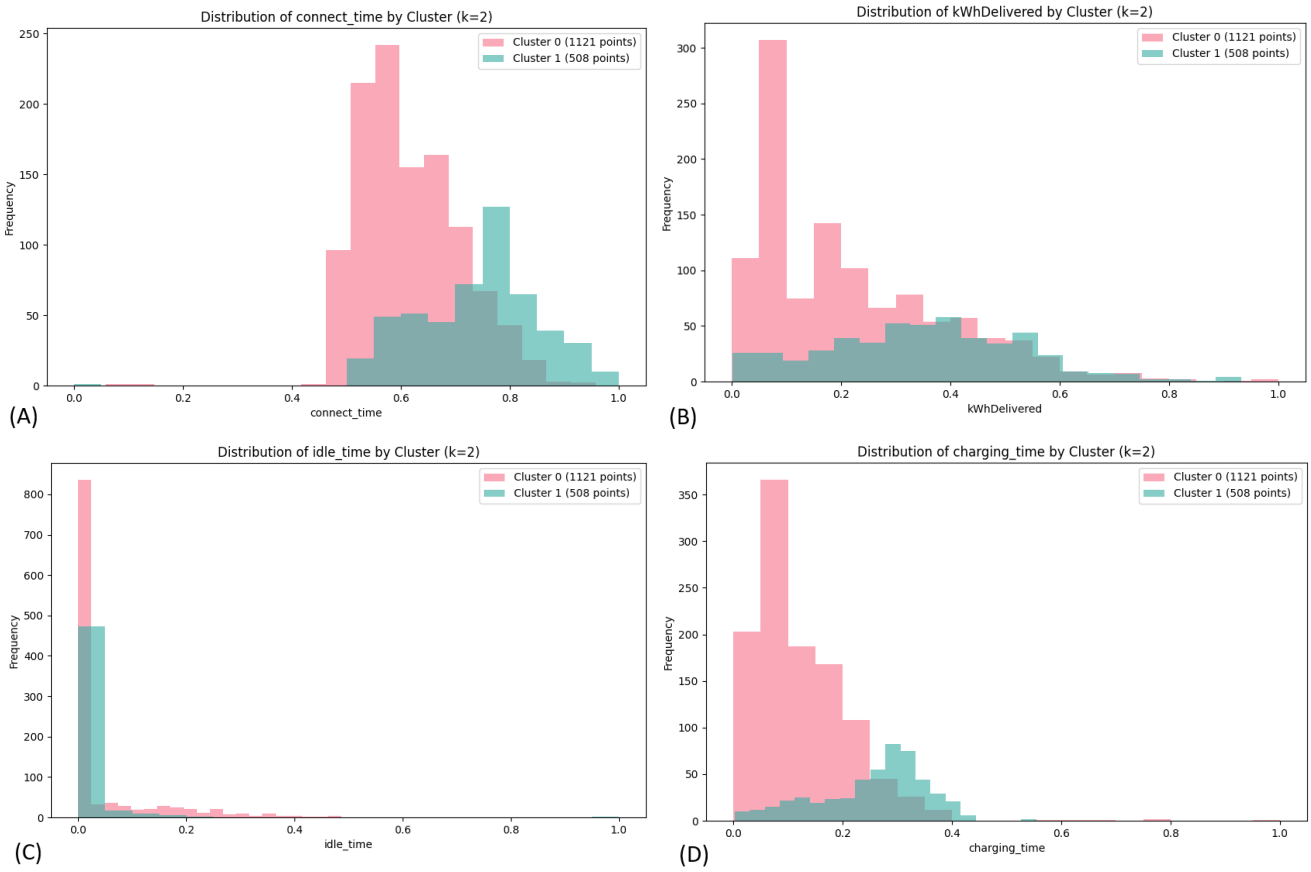

Figure 19: The distribution of charging sessions belonging to the two clusters created by KMeans for $k = 2$ in D_{SV} with respect to: (a) the "connect time" – feature F1, (b) the delivered power – feature F4, (c) the idle time – feature F5, and (d) the charging time – feature F6.

active for longer periods of time. Finally, the charging time (F6) reveals that the users of the large cluster consistently have rather short sessions, whereas those of the small cluster have a wider spread, probably due to a mix of regular short charges and occasionally longer charges.

Overall, there are two different user behaviors: the large cluster represents EV drivers who charge early, briefly, efficiently and for less power, whereas the small cluster contains users with much longer periods of time that consume more energy.

6.1.4. Conclusions

Our clustering methodology allows for extracting useful insights from the charging sessions stored in the O-V2X-MP database. It yields user profiles that reveal when EVs are typically charged, for how long, how much energy every charging session typically consumes, how long an EV occupies the charger even without charging etc. This information facilitates CPOs to schedule various aspects of charger maintenance, from repairing them to predicting the energy demand per hour.

6.2. Charging Load Prediction

6.2.1. Problem Definition

The goal of this task is to forecast the demand per individual EV charger based on its past charging sessions. This forecast is critical, because it allows CPOs to schedule the maintenance of the charging stations and, most importantly, to predict the idle capacity per day in a network of charging stations, thus laying the basis for dynamic capacity contracts with the DSO [17]. For example, in days with an extremely high overall load, the DSO can ensure the stability of the electrical grid by procuring and activating the capacity contracts to lower the power delivered by each charging station (e.g., from 22kW to 11kW).

More formally, we define this prediction task as follows:

Problem 2 (Day-ahead Charging Load Prediction). *Given the energy delivered by a particular charging station on a daily basis for a specific time period, predict the load demand for the day ahead so as to minimize the difference between the prediction and the actual value.*

Note that this is a univariate time series prediction task, given that it exclusively considers the historical energy consumption data (in kWh per day). We address it as *a regression task that uses as features the load of the last n days* (i.e., n time steps). In theory, we can apply a regression algorithm to the consumption data of any charging station, but in practice, a forecast is possible only for chargers with a sufficiently large transaction history.

Evaluation Measures. All the above regression models are evaluated with respect to the following measures [18, 19]:

- The Mean Absolute Error (**MAE**) estimates the absolute difference between the actual and predicted load across all days of the test set. In other words, it measures the average of the prediction residuals.

- The Mean Squared Error (**MSE**) is the average of the squared difference between the actual and predicted values across all testing days. It measures the variance of the prediction residuals.

- The Normalized Root Mean Squared Error (**NRMSE**) divides the square root of MSE with the range of the predicted values (i.e., maximum daily load - minimum daily load). This is a normalized measure of the standard deviation of prediction residuals.

- $\mathbf{R^2}$ is defined as 1 - MSE(prediction model)/MSE(trivial model), where the trivial model always predicts the mean of the target variable. Therefore, R^2 represents the proportion of the variance in the dependent variable, which is explained by the prediction model.

The first three evaluation measures take positive values, with lower scores indicating higher prediction accuracy. In contrast, R^2 always takes values lower than 1, with higher values indicating higher accuracy.

6.2.2. Solution

We address Problem 2 through the following regression algorithms:

- Linear Regression (LR) [20], a basic, parameter-free statistic model that estimates a linear relationship between the value to be predicted and the features used.

- Long Short-Term Memory (LSTM) [21], a type of recurrent deep neural network that is crafted for capturing long-term dependencies and complex patterns from long sequences.

- Extreme Gradient Boosting (XGBoost) [22], an established algorithm for learning a weighted ensemble of decision trees through gradient boosting.

- Autoregressive integrated moving average (ARIMA) [23], a common model of high effectiveness in short-term forecasting over non-stationary time series, due to its ability to capture temporal trends.

These four models are quite diverse in nature, because our goal is twofold: (i) to identify the most accurate prediction per EV charger, and (ii) to gain insights into how the characteristics of each time series affect the performance of each regression algorithm. For all algorithms, we used the implementation provided by scikit-learn[31], version 1.3.2.

Table 2: The charging stations from D_{PPC} selected for load prediction.

Charger	Total Volume (kWh)	Total Duration (hrs)	Total Sessions	Total Days	#Inactive Days	
C1	42,849	2,775	1,748	610	45	(7.4%)
C2	39,730	719	1,376	604	126	(20.9%)
C3	67,231	5,363	4,156	610	7	(1.2%)
C4	123,044	3,494	4,623	610	10	(1.6%)
C5	32,212	5,823	2,757	610	17	(2.8%)
C6	43,892	1,079	1,743	608	61	(10.0%)
C7	48,009	3,124	2,694	610	34	(5.6%)
C8	28,684	6,104	2,568	610	11	(1.8%)
Total	425,652	28,481	21,665			

Hyperparameter fine-tuning. The most critical decision is the number of time steps that should be used as features for tackling Problem 2. We performed preliminary experiments with

four time representative time steps: 1, 3, 7 and 15. The experimental results suggested that the best performance for each regression algorithm and EV charger is achieved by the smallest time step. Therefore, the input to Problem 2 for each EV charger is its overall charging load during the previous day.

For the hyperparameter tuning of the considered regression algorithms, we employed Optuna [24], an open-source library with generic techniques to efficiently explore the configuration space and optimize the parameters of any forecast model. The only exception is Linear Regression, which is parameter-free. The parameters that were considered in the fine-tuning process along with the parameters that were selected for each charging station are reported in Tables 3 and 7 (please refer to the scikit-learn documentation[32] for more details on each parameter).

Note that our LSTM consists of at least one and up to three hidden layers as well as a final dense layer with a single unit, which aggregates the outputs of the hidden layers. Therefore, only the units of the hidden layers are fine-tuned as hyperparameters. Note also that there is a different domain for the units of the first two hidden layers than those of the last one.

6.2.3. Experimental Results

$\mathbf{D_{PPC}}$. We selected the top-10 charging stations with the most transactions and then merged the charging stations with the same address, because they correspond to different sockets of the same charger or to multiple chargers in the same location (e.g., a parking lot, an airport etc). In both cases, the charging load is randomly distributed among them and, thus, a prediction model should treat them as a whole.

The end result of this cleaning process yielded the eight most popular charging stations, whose technical characteristics are summarized in Table 2. Note that they all involve at least 604 days of activity and that overall, they account for 21%, 11% and 17% of the total load consumed, duration and number of charging sessions, respectively, of the entire dataset in Table 4.

Finally, we define as testing set the 31 days in January, 2024, with all prior information constituting the training set of each

[31] `https://scikit-learn.org`

[32] `https://scikit-learn.org/0.21/documentation.html`

Table 3: The configuration parameters examined with grid search per regressor (on the left), and the selected values per charging station in Table 2 (on the right).

Parameter	Domain	C1	C2	C3	C4	C5	C6	C7	C8
#layers	{2, 4}	4	3	3	4	3	4	3	3
#units per layer	{30, 50}, step = 10	[50, 30, 50]	[40, 50]	[50, 50]	[50, 40, 30]	[50, 40]	[40, 50, 40]	[50, 50]	[50, 40]
#units in last layer	[15, 50], step = 5	45	50	50	25	20	45	50	20
Dropout rate per layer	from 0.2 to 0.3	[0.27, 0.30, 0.29, 0.26]	[0.23, 0.23, 0.22]	[0.28, 0.23, 0.22]	[0.22, 0.29, 0.26, 0.29]	[0.30, 0.24, 0.23]	[0.25, 0.29, 0.21, 0.27]	[0.28, 0.23, 0.22]	[0.28, 0.30, 0.20]
Learning rate	from 10^{-6} to 10^{-3}	0.0006	0.0003	0.001	0.0004	0.0009	0.0008	0.001	0.00003
Epochs	from 10 to 250	77	86	76	183	237	178	199	150
Optimizer	(1):adam, (2):rmsprop, (3):sgd, (4):adagrad, (5):adamax	(1)	(2)	(1)	(2)	(2)	(1)	(1)	(5)
Batch size	[1, 5] with step = 1	5	4	5	2	4	1	5	4

(a) LSTM

Objective	(1): squarederror, (2): pseudohubererror	(2)	(2)	(2)	(2)	(2)	(2)	(2)	(2)
Evaluation metric	(1):rmse, (2):mae, (3):mphe	(3)	(3)	(3)	(3)	(3)	(3)	(3)	(3)
Tree method	(1):gpu_hist, (2):gpu_exact	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
Booster	(1):booster, (2):gblinear	(2)	(2)	(2)	(2)	(2)	(2)	(2)	(2)
Max depth	{9, 10}	10	10	9	10	9	10	10	9
Learning rate	from 0.25 to 0.40	0.36	0.37	0.28	0.37	0.40	0.39	0.37	0.40
#estimators	from 480 to 500	482	482	490	482	482	487	482	484
Subsample	from 0.9 to 1.00	0.95	0.95	0.92	0.95	0.93	0.94	0.95	0.96
Colsample by tree	from 0.9 to 1.00	0.99	0.99	0.94	0.99	0.95	0.91	0.99	0.91
Min. child weight	{3, 4}	3	3	4	3	3	4	3	3

(b) XGBOOST

p	from 0 to 10	10	3	7	10	9	7	0	10
d	from 0 to 4	0	0	4	0	0	2	0	0
q	from 0 to 10	3	1	2	1	6	10	4	9

(c) ARIMA

charging station. As a result, the training set of the selected chargers comprises 573 to 579 total values.

The regression algorithms were fine-tuned on each charger of Table 2 based on the maximization of the R^2 metric. The corresponding performance per algorithm and charger is reported in Table 4. We observe that no single model consistently outperforms all others across all datasets. Yet, for each charger, the top performing model excels with respect to all four evaluation measures.

Interestingly, despite hyperparameter tuning, Linear Regression outperforms the more complex models in several cases. In fact, it is the top performer for C1 and C2, with its difference from the rest of the models being statistically significant. The least complex of the remaining models, ARIMA, also exhibited high performance in many cases, dominating all other regression algorithms on two chargers, C5 and C6. Note though that in the latter charger, its difference from LSTM is rather minor ($\ll 1\%$ in most measures). In the rest of the charging stations, the top performer is LSTM, with its superiority being statistically significant, albeit limited for C3 and C7. XGBOOST does not surpass the other models in any of the charging stations, but it consistently achieves competitive results, being close to the top performance in most cases.

To illustrate the relative performance of the four regression algorithms, Figure 20 depicts their predictions across all days of the testing set (i.e., January 2024) for the two extremes of Table 4: C4, where most chargers make the most accurate predictions, and C6, where most chargers are the least accurate.

The plot for C4 (on the left) reveals a strong competition between Linear Regression and LSTM, with the former outperforming LSTM on several days. However, LSTM dominates Linear Regression on most days with high energy consumption. XGBoost follows them in close distance, but the ARIMA model struggled to follow these trends, predicting a value close to the average one, thus resulting in the weakest by far performance.

Conversely, in the more challenging case of C6, Linear Regression, LSTM and XGBoost demonstrate very similar performance, which is however significantly lower than the actual consumption in almost all days. Instead, the forecasts of ARIMA are consistently closer to the actual values, even though they basically predict the mean value of the time series, as indicated by the value of R^2 in Table 4.

Overall, the forecasting behavior of XGBoost, Linear Regression, and LSTM is closely related, with the first two producing nearly identical predictions. LSTM typically outperforms them, while the ARIMA model exhibits a significantly different behavior with nearly constant predictions.

ACF/PACF Analysis. To explain these patterns, we perform an analysis of the complexity of the time series formed by the overall load per EV charger in Table 2. Our goal is to identify the conditions that determine the performance of regression models so that we can *a-priori* determine the best performing one in each case. We actually compute the *AutoCorrelation Function* (**ACF**) [25] and the *Partial AutoCorrelation Function* [26] (**PACF**), which describe the overall and direct correlations of lagged values, respectively.

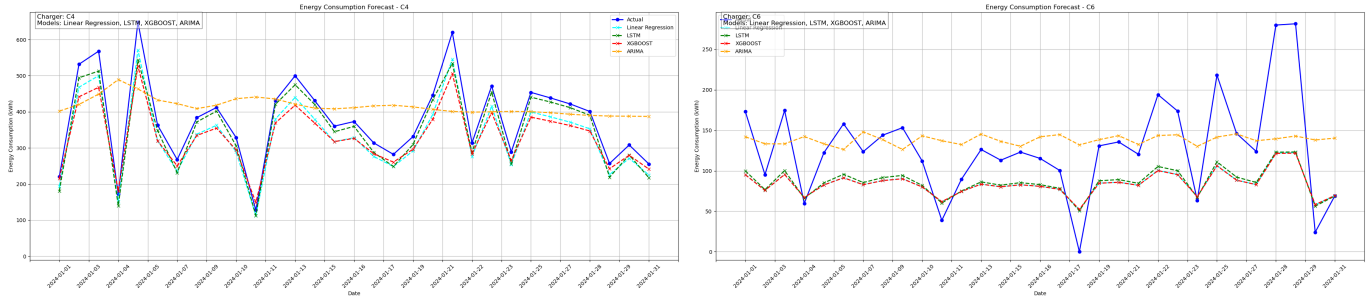

Figure 20: Daily predictions by all regression algorithms for the least and most challenging charging stations in Table 2, C4 (on the left) and C6 (on the right), resp.

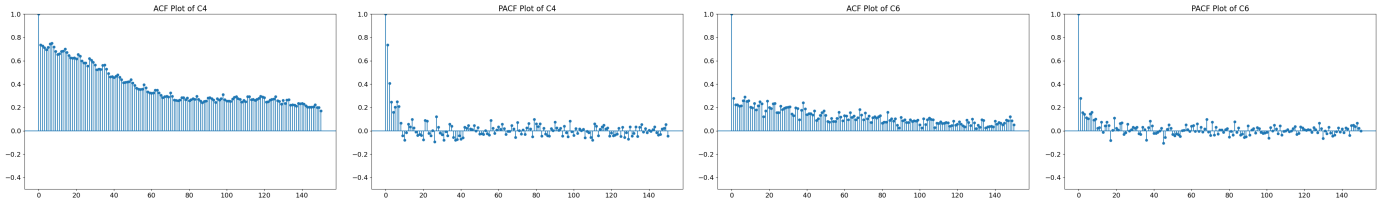

Figure 21: Auto-correlation (ACF) and partial autocorrelation (PCF) for the least and most challenging charging stations in Table 2, C4 (left) and C6 (right), resp.

Figure 21 depicts the ACF of C4 and C6, showing positive correlations across multiple lags in both charging stations. C4 stands out with high initial autocorrelation values that gradually decrease across all lags, indicating strong temporal dependencies, suggesting the presence of trends or seasonal patterns. These patterns justify along with the high effectiveness of most regression models in Table 4. In contrast, C6 exhibits a rapid decline in autocorrelation values, suggesting that its past values have less influence over time. This indicates the presence of shorter-term correlations rather than long-term dependencies. This probably accounts for the low prediction accuracy of all regression algorithms regarding C6, as shown in Table 4.

Regarding the PACF of both chargers, C4 shows a more structured pattern, with several spikes suggesting the influence of certain lags. This pattern indicates potential cyclic behavior within the data, as the partial autocorrelation values gradually decline over the time lags. This behavior contributes significantly to the high accuracy of the prediction models in Table 4. In contrast, the PACF of C6 shows a less consistent pattern, suggesting a more complex and potentially random data relationship, where the dependencies between time lags are not as clearly defined. This justifies the low prediction accuracy of all prediction models for C6 in Table 4.

ADF Analysis. These results suggested the need for further investigation into the data nature of the time series dataset and its impact on model performance. For this reason, we applied the *Augmented Dickey-Fuller* (**ADF**) test [27]. Its goal is to assess the stationarity (or not) of a time series by investigating its statistical properties and hypothesizing the presence of a unit root within the series, which serves as the null hypothesis. Note that a time series is *stationary* if its statistical properties (e.g., mean, variance, covariance, and standard deviation) do not vary with time or are not a function of time; in other words, time series with trends or seasonality are not stationary, because their values depend on the time of observation.

The ADF test provides a p-value that is compared to a significance level typically set at 0.05 [28] [29]. It also provides the ADF statistic and the critical values, which are also crucial for validating the null hypothesis: if the p-value is less than 0.05 and the ADF statistic is more negative than the critical values of 1%, 5% and 10%, the null hypothesis can be rejected, indicating that the series is stationary. The results in Table 5 indicate only C1 exhibits a stationary behavior. All other charging stations are non-stationary, which hampers their modeling, leading to unreliable forecasts, given that the relationships between data points changes over time.

Entropy Analysis. To shed more light on the complexity of the time series of overall load per EV charger, we follow [30] and compute the three entropy measures below:

- Spectral Entropy [31] transforms the given time series from the time domain to the frequency one through the discrete Fourier transform, providing information about the power of each frequency component. It reaches its maximum when the power is uniformly distributed across all frequencies, indicating a white noise signal.
- Approximate Entropy [32] quantifies the regularity or unpredictability in the time series, measuring the likelihood that subsequences of the same length remain similar as the series progresses.
- Sample Entropy [31] improves on approximate entropy by disregarding self-matches, i.e., measuring the likelihood that subsequences of the same length do not remain similar as the series progresses.

For all entropy measures, values close to zero indicate no variation (i.e., a stable time series), with positive values being proportional to the randomness in the observed data. That is, the higher all these entropies are, the higher is the complexity of the corresponding time series. For the implementation of all entropies we used the open-source Python library AntroPy[33].

Note that we considered more entropies indicative of the complexity of a time series like permutation [33] and shannon

[33] `https://github.com/raphaelvallat/antropy`

Table 4: Forecasting performance of the fine-tuned regression algorithms over the charging stations in Table 2. The best performance per evaluation measure and charger is highlighted in bold.

Charger	Regressor	R^2	MAE	RMSE	NRMSE
C1	LR	**0.348**	**36.3**	**49.9**	**0.537**
	LSTM	0.311	37.2	51.3	0.552
	XGBoost	0.342	36.5	50.1	0.539
	ARIMA	-0.003	46.2	61.9	0.666
C2	LR	**0.215**	**45.2**	**61.4**	**0.498**
	LSTM	-0.075	47.5	71.9	0.582
	XGBoost	0.206	45.4	61.7	0.500
	ARIMA	-0.005	56.0	69.5	0.563
C3	LR	0.603	29.7	39.1	0.324
	LSTM	**0.680**	**26.3**	**35.2**	**0.291**
	XGBoost	0.647	27.8	36.9	0.306
	ARIMA	0.028	49.8	61.3	0.508
C4	LR	0.843	45.2	47.4	0.125
	LSTM	**0.915**	**27.7**	**35.0**	**0.093**
	XGBoost	0.765	49.7	58.1	0.154
	ARIMA	-0.124	100.1	127.1	0.336
C5	LR	-0.098	44.5	56.9	0.433
	LSTM	-0.133	43.5	57.8	0.439
	XGBoost	0.001	41.8	54.2	0.413
	ARIMA	**0.115**	**41.0**	**51.0**	**0.388**
C6	LR	-0.075	52.2	64.3	0.500
	LSTM	0.001	49.7	62.0	0.482
	XGBoost	-0.075	52.2	64.3	0.500
	ARIMA	**0.009**	**46.6**	**61.7**	**0.480**
C7	LR	0.376	22.6	30.0	0.377
	LSTM	**0.390**	**22.2**	**29.7**	**0.373**
	XGBoost	0.372	22.7	30.1	0.379
	ARIMA	0.003	28.5	38.0	0.477
C8	LR	0.012	21.2	24.7	0.739
	LSTM	**0.097**	**20.3**	**23.6**	**0.706**
	XGBoost	0.007	21.3	24.8	0.740
	ARIMA	0.082	19.4	24.8	0.712

entropy [34], 2-Regimes Complexity [35] as well as the frequentist binning approach [36]. Yet, they all yield almost identical values across all chargers of Table 2, providing no insights to our analysis. Therefore, we exclude them for brevity.

The values of the selected entropy measures are presented in Table 6. We observe that they indicate significant irregularity across all datasets, with complex patterns characterized by nearly random values. To extract more conclusions from these findings, we define a two dimensional space formed by spectral and approximate entropy on the horizontal and vertical axis, respectively. We selected these two entropies, as they yield more distinctive values than sample entropy. The resulting scatter plot, which appears in Figure 22, can be used for a-priori identifying the best performing regression model per EV charger. Note that each point (i.e., time series of a charging station) is associated with the two top performing regression models. This plot leads us to the following conclusions:

- Linear Regression works best in chargers with low approximate entropy and high spectral entropy. That is, it excels in data with regular patterns but rich dynamics, which makes it predictable in the short term despite having complex values across different time scales.
- LSTM works best in chargers with high approximate entropy and relatively low spectral entropy. This indicates that while the data may seem unpredictable or chaotic in the short term, LSTM can capture underlying periodic or longer-term trends.
- XGBOOST performs moderately well under many settings, especially with high approximate and spectral entropy.
- ARIMA performs well in time series with moderate complexity and high irregularity, i.e., with relatively medium approximate entropy and high spectral entropy values.

Table 5: The outcomes of the ADF test over the charging stations in Table 2. CV stands for Critical Value.

Charger	ADF statistic	p-value	CV1%	CV5%	CV10%
C1	-6.06	$\ll$0.00	-3.44	-2.57	-2.87
C2	-2.17	0.22	-3.44	-2.57	-2.87
C3	-2.58	0.10	-3.44	-2.57	-2.87
C4	-1.38	0.59	-3.44	-2.57	-2.87
C5	-1.47	0.55	-3.44	-2.57	-2.87
C6	-2.11	0.24	-3.44	-2.57	-2.87
C7	-2.67	0.08	-3.44	-2.57	-2.87
C8	-3.00	0.04	-3.44	-2.57	-2.87

Table 6: Complexity measures of the overall load per EV charger in Table 2.

Entropy	C1	C2	C3	C4	C5	C6	C7	C8
Spectral	7.339	6.923	6.448	4.151	6.619	7.115	7.317	7.147
Approx.	5.467	4.895	5.830	6.000	5.515	5.383	5.500	5.269
Sample	1.813	1.382	1.901	1.116	1.854	1.850	2.069	2.017

D_{PSDN}, D_{LCND} *and* D_{SV}. We applied the above methodology to a subset of the charging stations included in the rest of the datasets. More specifically, we selected the top-3 chargers with the most transactions from D_{PSDN}, which are labeled as SID1, SID2 and SID3 in the following. We did the same in D_{LCND}, selecting three more chargers: SID4, SID5 and SID6. For D_{SV},

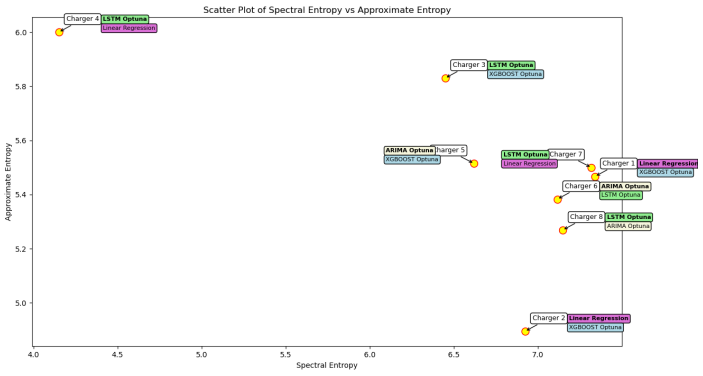

Figure 22: The relationship between the chargers in Table 2 and two entropy measures (Spectral and Approximate Entropy). Each point represents a charger, with its position determined by its entropy values. Additionally, for each point, the two dominant forecasting models are highlighted.

Table 7: The configuration parameters examined with grid search per regressor (on the left), and the selected values per charging station in Table 8 (on the right).

Parameter	Domain	SID1	SID2	SID3	SID4	SID5	SID6	SID7
#layers	{2, 4}	4	3	4	3	3	4	3
#units per layer	{30, 50}, step = 10	[30, 50, 30]	[40, 50]	[30, 40, 40]	[50, 40]	[40, 30]	[50, 40]	[50, 40]
#units in last layer	[15, 50], step = 5	50	30	50	20	15	20	20
Dropout rate per layer	from 0.2 to 0.3	[0.23, 0.29, 0.25, 0.21]	[0.24, 0.20, 0.21]	[0.25, 0.29, 0.23, 0.25]	[0.28, 0.30, 0.20]	[0.23, 0.23, 0.27]	[0.28, 0.30, 0.20]	[0.28, 0.30, 0.20]
Learning rate	from 10^{-6} to 10^{-3}	0.0002	0.0009	0.0010	0.0035	0.0002	0.0038	0.0039
Epochs	from 10 to 250	112	200	14	105	196	68	97
Optimizer	(1): adam, (2): rmsprop, (3): sgd, (4): adagrad, (5): adamax	(1)	(3)	(2)	(1)	(2)	(1)	(1)
Batch size	[1, 5], step = 1	1	5	1	5	1	5	5

(a) LSTM

Objective	(1): squarederror, (2): pseudohubererror	(1)	(2)	(1)	(1)	(2)	(1)	(2)
Evaluation metric	(1): rmse, (2): mae, (3): mphe	(2)	(3)	(2)	(2)	(2)	(3)	(1)
Tree method	(1): gpu_hist, (2): gpu_exact	(2)	(2)	(2)	(2)	(1)	(2)	(2)
Booster	(1): gbtree, (2): gblinear, (3): dart	(2)	(2)	(2)	(2)	(2)	(3)	(3)
Max depth	{9, 10}	10	9	9	9	9	10	10
Learning rate	from 0.25 to 0.40	0.36	0.35	0.33	0.38	0.27	0.37	0.36
#estimators	from 480 to 500	493	500	497	499	491	499	485
Subsample	from 0.9 to 1.00	0.94	0.94	0.90	0.90	0.93	0.95	0.95
Colsample by tree	from 0.9 to 1.00	0.93	0.93	0.90	0.99	0.93	0.96	0.92
Min. child weight	{3, 4}	3	3	4	4	3	3	3

(b) XGBOOST

p	from 0 to 10	4	2	7	7	5	5	8
d	from 0 to 4	1	2	2	2	2	1	0
q	from 0 to 10	0	6	8	8	6	8	7

(c) ARIMA

Table 8: The charging stations from D_{PSDN}, D_{LCND} and D_{SV} selected for load prediction.

Charger	Total Volume (kWh)	Total Duration (hrs)	Total Sessions	Total Days	#Inactive Days
SID1	12,812	7,802	1,412	1,237	490 (39.6%)
SID2	15,714	8,922	1,808	1,238	422 (34.1%)
SID3	25,053	934	1,130	300	29 (9.7%)
SID4	13,704	8,335	1,449	1,097	353 (32.2%)
SID5	14,619	7,342	1,268	1,096	353 (32.2%)
SID6	12,266	6,834	1,134	1,098	387 (35.3%)
SID7	29,072	10,799	1,683	904	310 (34.3%)
Total	123,241	50,966	9,884		

due to the limited transaction history across its 8 chargers and the co-location of all chargers in the same building, the entire dataset was treated as a single charger, labeled as SID7. The technical specifications of these chargers appear in Table 8.

Following the setup used for D_{PPC}, the testing set for each charger was set to the last month (in particular the last 31 days) of its transaction history. Note that this resulted in nearly identical testing periods across all seven chargers, from the middle of August, 2021 to the middle of September, 2021. The four regression algorithms were fine-tuned with Optuna so as to maximize the R^2 metric. The selected configurations are reported in Table 7 and their performance in Table 9.

We observe that Linear Regression is the top performer in five charging stations with respect to R^2 and usually the other evaluation measures, too. The only exceptions are SID3 and SID4, where the highest performance corresponds to XGBoost and LSTM, respectively. Even in these cases, though, Linear Regression follows in close distance, with its R^2 being lower by (far) less than 0.01. In SID4, Linear Regression actually exhibits the best scores for RMSE and NRMSE.

Among the remaining regression algorithms, XGBoost is proven to be highly competitive, typically achieving the second best scores across most evaluation measures. It actually outperforms Linear Regression in SID1 and SID2 with respect to RMSE and NRMSE, albeit to a minor extent. LSTM typically ranks third, mostly due to the lack of a large training dataset that allows for extracting complex patterns. Finally, ARIMA typically exhibits the worst performance with respect to all evaluation measures. The reason is that its predictions are very close to the average load, as indicated by its R^2 score. The only ex-

Table 9: Forecasting performance of the fine-tuned regression algorithms over the charging stations in Table 8. The best performance per evaluation measure and charger is highlighted in bold.

Charger	Regressor	R^2	MAE	RMSE	NRMSE
SID1	LR	**0.419**	**5.730**	9.633	0.685
	LSTM	0.200	7.204	10.515	0.747
	XGBoost	0.403	6.173	**9.107**	**0.647**
	ARIMA	0.013	7.932	11.530	0.820
SID2	LR	**0.571**	**3.513**	5.334	0.488
	LSTM	0.363	5.420	5.986	0.547
	XGBoost	0.505	4.090	**5.281**	**0.483**
	ARIMA	0.021	5.562	7.762	0.709
SID3	LR	0.409	35.238	42.236	0.319
	LSTM	0.170	39.082	50.072	0.379
	XGBoost	**0.414**	**34.984**	**42.059**	**0.318**
	ARIMA	-0.018	44.522	55.453	0.419
SID4	LR	0.541	5.393	**5.529**	**0.460**
	LSTM	**0.548**	**5.138**	6.127	0.510
	XGBoost	0.370	5.974	7.342	0.611
	ARIMA	0.178	6.047	9.059	0.753
SID5	LR	**0.376**	**7.425**	10.019	0.625
	LSTM	0.319	8.512	10.383	0.648
	XGBoost	0.346	7.803	**9.909**	**0.618**
	ARIMA	0.023	8.605	13.496	0.842
SID6	LR	**0.439**	**6.340**	**7.398**	**0.531**
	LSTM	0.365	7.259	7.538	0.541
	XGBoost	0.413	5.802	7.718	0.554
	ARIMA	0.168	6.972	9.974	0.716
SID7	LR	**0.657**	**10.967**	**11.564**	**0.324**
	LSTM	0.429	14.598	15.535	0.436
	XGBoost	0.202	16.802	18.973	0.532
	ARIMA	0.429	11.133	18.275	0.513

ception pertains to SID7, which should be attributed to the non-stationary data it conveys, for which ARIMA is crafted.

To shed more light into the relative performance of the four regression algorithms, Figure 23 presents the exact predictions of each model per testing day for SID7, which corresponds to the highest average R^2 score (0.429), and SID3, which corresponds to the lowest one (0.266).

For SID7, LSTM is more accurate in predicting mean load values, but fails to capture the peaks and valleys of the actual loads. ARIMA, while not fully aligned with the overall load pattern, performs better than the other models in predicting the lower load values. Linear Regression demonstrates a more balanced prediction capability, effectively capturing peak loads and maintaining momentum across mean and low loads. Despite its competitive performance in all other chargers, XGBoost is the least effective model in this case, failing to provide accurate predictions in any of the considered days.

For SID3, Linear Regression and XGBoost produce dominant, highly similar load predictions that closely follow the actual data patterns, but do not capture the exact consumption levels across all days. LSTM, while relatively close to the leading models, tends to predict values near the overall mean of the time series, rather than aligning with the actual load. ARIMA, demonstrates the lowest accuracy, showing minimal adaptation to the actual load, with its predictions centered around the mean values for the entire forecasting period.

Table 10: The outcomes of the ADF test over the charging stations in Table 8. CV stands for Critical Value.

Charger	ADF statistic	p-value	CV1%	CV5%	CV10%
SID1	-2.63	0.09	-3.44	-2.86	-2.57
SID2	-2.60	0.09	-3.44	-2.86	-2.57
SID3	-2.86	0.05	-3.45	-2.87	-2.57
SID4	-2.92	0.04	-3.44	-2.86	-2.57
SID5	-1.93	0.04	-3.44	-2.86	-2.57
SID6	-2.69	0.08	-3.44	-2.86	-2.57
SID7	-2.21	0.20	-3.44	-2.86	-2.57

Overall, the forecasting performance of Linear Regression and XGBoost is closely related, with the former emerging as the most accurate forecasting model overall, consistently delivering strong and competitive results across all chargers in Table 8. On the other extreme lies the ARIMA model, which typically underperforms all other models, while LSTM lies in the middle. Below, we examine the conditions determining these patterns.

ACF/PACF Analysis. Figure 24 shows the ACF and PACF plots for SID7 and SID3. Both charging stations exhibit periodic patterns with respect to ACF, while the PACF plots reveal more complex behavior across time lags.

Regarding ACF, SID7 stands out due to its strong periodic correlation patterns in positive and negative time lags, which indicate the presence of seasonal patterns. Similar, albeit weaker, periodic correlation patterns appear in the case of SID3.

Regarding PACF, SID7 exhibits a more structured pattern with several positive and negative peaks, highlighting the influence of certain lags. This pattern suggests a periodic behavior within the data, with the partial autocorrelation values gradually declining over time lags. In contrast, SID3 shows less consistent periodicity, with weak relationships across time lags, indicating the presence of more random behavior in the data, apart from the limited transaction days available in the dataset.

The strong periodicity patterns account for the high average scores of SID7 with respect to most evaluation measures, especially R^2. The weaker periodicity patterns of SID3 along with the considerably fewer training and inactive days than all other chargers (cf. Table 8) yield a much lower performance.

ADF Analysis. The ADF test was applied to all chargers, with the results presented in Table 10. The p-value and the critical values indicate that three chargers are stationary, exhibiting periodic patterns: SID3, SID4 and SID5. Despite these patterns, SID3 is quite challenging, due to the lack of sufficient training data and the high number of active days. The rest of the chargers, including SID7, display non-stationary data, suggesting more challenging relationships within the data that change over time. Still, the periodicity revealed by the ACF/PACF analysis allows for a much higher predictive accuracy.

Entropy Analysis. To further justify the performance of each regression model per charging station in Table 8, we computed Spectral Entropy (defined above) along with:

- The Shannon Entropy measures the uncertainty or unpredictability within a dataset. In time series analysis, it quan-

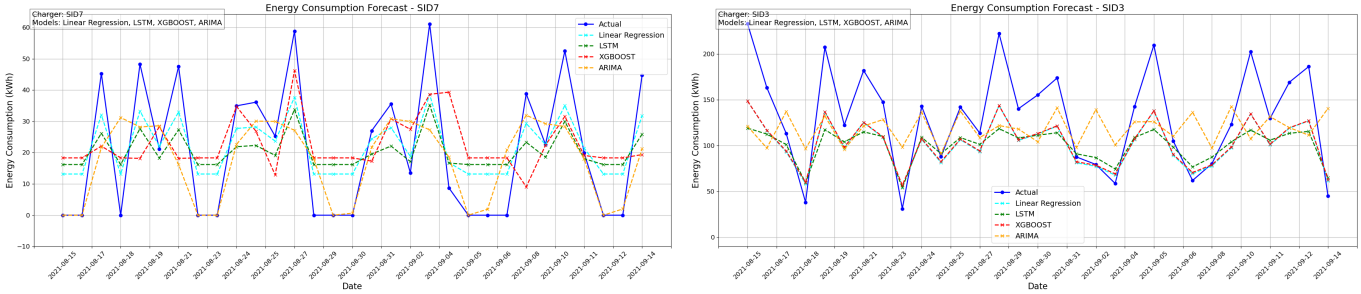

Figure 23: Daily predictions by all regression algorithms for the charging stations in Table 8 with the highest and the lowest average R^2 score, i.e., SID7 (on the left) and SID3 (on the right), respectively.

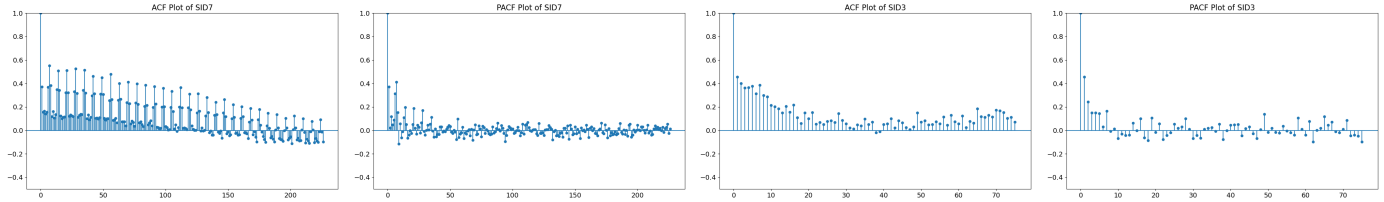

Figure 24: Auto-correlation (ACF) and partial autocorrelation (PCF) for the charging stations in Table 8 with the highest and the lowest average R^2 score, i.e., SID7 (on the left) and SID3 (on the right), respectively.

Table 11: Complexity measures of the overall load per EV charger in Table 8.

Entropy	SID1	SID2	SID3	SID4	SID5	SID6	SID7
Spectral	7.457	7.141	5.708	7.315	7.672	7.377	6.299
Shannon	1.900	2.083	2.905	2.303	2.183	2.291	2.310
2-Regimes	1.854	2.189	2.994	2.078	2.089	2.017	2.201

tifies the amount of information generated by a stochastic or random process. Higher Shannon Entropy suggests a more complex or "disordered" system, while lower entropy indicates greater regularity or predictability in the data.

- The 2-Regimes Entropy measures the complexity of a time series by dividing it into two distinct regimes, each representing different states or patterns in the data. It then calculates the entropy in each regime to capture how the time series' behavior changes across these phases, such as shifts in volatility or variability. A high 2-Regimes Entropy indicates that each regime has unique and complex characteristics, suggesting a non-stationary or multi-state time series.

Note that we considered more entropy measures related to time series complexity, namely Permutation, Approximate, Frequentist, and Sample Entropy. However, they exhibit much lower distinctiveness than the three selected ones, providing no further insights for our analysis. We omit them for brevity.

The values of the selected entropy measures are presented in Table 11. We observe that despite its nonstationary behavior, SID7 exhibits relatively low complexity across all entropies, especially the Spectral one, where it has the second lowest value. This confirms the results of the ACF/PACF analysis. In combination with its larger training set, these settings enhance the accuracy of regression algorithms in capturing the consumption patterns of SID7. In contrast, SID3 exhibits high unpredictability and complexity, scoring the highest values across all seven chargers for the Shannon and the 2-Regimes Entropies, despite

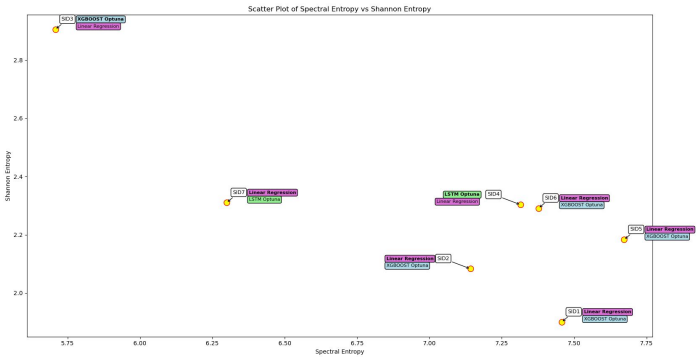

Figure 25: The relationship between the chargers in Table 8 and two entropy measures (Spectral and Shannon Entropy). Each point represents a charger, with its position determined by its entropy values. Additionally, for each point, the two dominant forecasting models are highlighted.

its stationary behavior. This behavior along with its limited training and inactivity days result in the (surprisingly) lowest score for Spectral Entropy.

To better understand the relationship between the entropy values and the dominant regression models per charger, we created a diagram with Spectral Entropy on the horizontal axis and Shannon Entropy on the vertical one. We then added a point per charging station, annotating it with the two top-performing regression models. The resulting scatter plot appears in plot Figure 25, leading to the following conclusions:

- Linear Regression performs best with chargers that have low Shannon Entropy and high Spectral Entropy. This means that it excels in handling data with versatile complexities and relatively low random patterns across different time scales.

- LSTM works best with data exhibiting medium Shannon Entropy and medium to high Spectral Entropy. This means that it is well-suited for capturing more complex patterns with high variability and randomness.

- XGBoost performs best with highly complex data that yields high Shannon and low Spectral Entropy. It typically ranks second, because it can capture noise in the data patterns, while adapting better than LSTM to smaller training sets.
- ARIMA performs well in time series with relatively low Spectral Entropy and low Shannon Entropy values. This means that it struggles with high data variability, but is effective in capturing periodic patterns.

6.2.4. Conclusions

Overall, three key factors contributed to the relatively low performance of the selected regression models in most cases:

1. The data recorded per charger are limited. The training set of the chargers in Table 2 has up to ~570 instances, while most chargers in Table 8 have double as many. Note, though, that in the latter case, 1/3 of the training instances corresponds to days of inactivity, with zero consumed energy (the corresponding portion in Table 2 is 10% or lower). These training datasets do not suffice for sophisticated models like LSTM to learn complex patterns effectively. More data is essential to unlock the full potential of deep learning models.
2. The ADF, ACF and PACF analyses highlight the non-stationarity and near-random nature of the time series, explaining why the regression models achieve the best results with a time step (i.e., lag) of 1. In other words, the high irregularity and complexity of the patterns make it difficult for the regression models to capture long-term dependencies, with small time steps yielding higher effectiveness.
3. The entropy measures are useful for a-priori assessing the complexity of each charger's time series data. Chargers like C4 and SID3 exhibit high entropy values, suggesting a highly unpredictable and complex time series. These properties make it more challenging for the regression models to capture the underlying data patterns, particularly over longer time horizons. In contrast, chargers like C4 and SID7 show more predictable patterns, with lower entropy values, which indicate more stable and less complex time series, enabling simpler models like Linear Regression to achieve high accuracy. These patterns emphasize the role of entropy in a-priori estimating the performance of regression model.

These settings lead to the "*No Free Lunch*" theorem [37], which dictates that no single forecasting model consistently outperforms others across all EV chargers. An increase in the quantity of training data might increase the performance of the more sophisticated regressions algorithms like LSTM.

Another conclusion of our analysis is that the characteristics of time series are indicative to a large extent of the performance of prediction algorithms. Typically, the stationary time series, as indicated by the ADF test, pose relatively easy prediction tasks for sophisticated regression models like LSTM (e.g., C1). The complexity of non-stationary time series depends on their (partial) autocorrelation as well as on their entropy. Higher ACF values for longer periods indicate longer-term correlations, which thus facilitate high prediction accuracy, especially when accompanied by low values for spectral, approximate and sample entropy, which are indicative of a less complex (i.e., unpredictable) behavior. These settings characterize C4, where most regression models achieve their highest effectiveness. The opposite settings pertain to C6 and SID3, where most exhibit very poor forecast accuracy.

7. Conclusions & Future Work

We have presented the backend and the frontend of O-V2X-MP, an open-source platform that covers all aspects of charging stations, the backbone of e-mobility. Charging point operators can use the platform to maintain a network of EV chargers, to run diagnostics, to extract insights from the recorded charging sessions and to apply V2G services like flexible capacity contracts. EV drivers can use the platform to detect and navigate to the closest charging session, to manage their charging history and make the most of V2X services (e.g., demand response services that reduce prices whenever the renewables reach their peak). Special care was taken to demonstrate the data analytics capabilities of the platform through two use cases: (i) the clustering of charging sessions into charging profiles, which allows for predicting EV drivers behavior, and (ii) the prediction of the overall load for any EV charger.

In the future, we plan to extend the data analytics capabilities of the O-V2X-MP with more advanced techniques (e.g., multivariate forecasts for EV load leveraging seasonality). We also plan to enhance the billing engine with support for more advanced payment protocols, such as the Lightning Network[34], which solves the bitcoin scalability problem.

Acknowledgement

This document is the result of the research project funded by European Union´s Horizon Europe R&I programme under grant agreement no. 101056765. Views and opinions expressed in this document are however those of the authors only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the grating authority can be held responsible for them.

References

[1] S. Islam, A. Iqbal, M. Marzband, I. Khan, A. M. Al-Wahedi, State-of-the-art vehicle-to-everything mode of operation of electric vehicles and its future perspectives, Renewable and Sustainable Energy Reviews 166 (2022) 112574.

[2] L. Noel, G. Z. De Rubens, J. Kester, B. K. Sovacool, Vehicle-to-grid, Springer (2019).

[3] E. L. Karfopoulos, N. D. Hatziargyriou, Distributed coordination of electric vehicles providing v2g services, IEEE Transactions on Power Systems 31 (1) (2015) 329–338.

[4] C. Gschwendtner, S. R. Sinsel, A. Stephan, Vehicle-to-x (v2x) implementation: An overview of predominate trial configurations and technical, social and regulatory challenges, Renewable and Sustainable Energy Reviews 145 (2021) 110977.

[34] `https://lightning.network`

[5] Open-Charge-Alliance, Open charge point protocol 2.0.1 (2020).
[6] Open-Charge-Alliance, Open charge point protocol 1.6 (2015).
[7] S. Brown, Software architecture for developers, Coding the Architecture (2013).
[8] A. Lekidis, H. Morais, Open V2X management platform cyber-resilience and data privacy mechanisms, in: ARES, 2024, pp. 139:1–139:8.
[9] J. Daemen, V. Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard, Information Security and Cryptography, Springer, 2002.
[10] Z. J. Lee, T. Li, S. H. Low, ACN-Data: Analysis and Applications of an Open EV Charging Dataset, in: Proceedings of the Tenth International Conference on Future Energy Systems, e-Energy '19, 2019.
[11] J. Han, M. Kamber, J. Pei, Data mining concepts and techniques third edition, The Morgan Kaufmann Series in Data Management Systems (2012).
[12] A. Lekidis, et al., Deliverable D3.3: EVs use Clustering results report, EV4EU: Electric Vehicles Management for carbon neutrality in Europe, 2023.
[13] A. Likas, N. Vlassis, J. J. Verbeek, The global k-means clustering algorithm, Pattern Recognit. 36 (2) (2003) 451–461.
[14] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: SODA, 2007, pp. 1027–1035.
[15] H. Park, C. Jun, A simple and fast algorithm for k-medoids clustering, Expert Syst. Appl. 36 (2) (2009) 3336–3341.
[16] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: an efficient data clustering method for very large databases, in: SIGMOD, 1996, pp. 103–114.
[17] T. Unterluggauer, F. Hipólito, P. B. Andersen, J. Rich, M. Marinelli, Conditional connection agreements for ev charging: Review, design, and implementation of solutions, Design, and Implementation of Solutions (2024).
[18] V. Plevris, G. Solorzano, N. P. Bakas, M. E. A. Ben Seghier, Investigation of performance metrics in regression analysis and machine learning-based prediction models, in: ECCOMAS Congress, 2022.
[19] D. Chicco, M. J. Warrens, G. Jurman, The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation, Peerj computer science 7 (2021) e623.
[20] X. Su, X. Yan, C.-L. Tsai, Linear regression, Wiley Interdisciplinary Reviews: Computational Statistics 4 (3) (2012) 275–294.
[21] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[22] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: SIGKDD, 2016, pp. 785–794.
[23] J. Contreras, R. Espinola, F. J. Nogales, A. J. Conejo, Arima models to predict next-day electricity prices, IEEE transactions on power systems 18 (3) (2003) 1014–1020.
[24] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: KDD, 2019, pp. 2623–2631.
[25] J. H. F. Flores, P. M. Engel, R. C. Pinto, Autocorrelation and partial autocorrelation functions to improve neural networks models on univariate time series forecasting, in: IJCNN, 2012, pp. 1–8.
[26] F. L. Ramsey, Characterization of the partial autocorrelation function, The Annals of Statistics (1974) 1296–1301.
[27] R. Mushtaq, Augmented dickey fuller test (2011).
[28] S. Cook, Finite-sample critical values of the augmented dickey-fuller statistic: a note on lag order, ECONOMIC ISSUES-STOKE ON TRENT-6 (2) (2001) 31–46.
[29] E. Herranz, Unit root tests, Wiley Interdisciplinary Reviews: Computational Statistics 9 (3) (2017) e1396.
[30] M. Ponce-Flores, J. Frausto-Solís, G. Santamaría-Bonfil, J. Pérez-Ortega, J. J. González-Barbosa, Time series complexities and their relationship to forecasting performance, Entropy 22 (1) (2020) 89.
[31] J. S. Richman, D. E. Lake, J. R. Moorman, Sample entropy, in: Methods in enzymology, Vol. 384, 2004, pp. 172–184.
[32] S. M. Pincus, Approximate entropy as a measure of system complexity., Proceedings of the national academy of sciences 88 (6) (1991) 2297–2301.
[33] C. Bandt, B. Pompe, Permutation entropy: a natural complexity measure for time series, Physical review letters 88 (17) (2002) 174102.
[34] J. Lin, Divergence measures based on the shannon entropy, IEEE Transactions on Information theory 37 (1) (1991) 145–151.
[35] J. G. Brida, L. F. Punzo, Symbolic time series analysis and dynamic regimes, Structural Change and Economic Dynamics 14 (2) (2003) 159–183.
[36] S. Verdú, Empirical estimation of information measures: A literature guide, Entropy 21 (8) (2019) 720.
[37] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.