Autonomous Agents and Multi-Agent Systems (2024) 38:45
https://doi.org/10.1007/s10458-024-09671-8

Personalised electric vehicle charging stop planning through online estimators

Elnaz Shafipour[1] · Sebastian Stein[1] · Selin Ahipasaoglu[1]

Accepted: 7 August 2024

Abstract

In this paper, we address the problem of finding charging stops while travelling in electric vehicles (EVs) using artificial intelligence (AI). Choosing a charging station is challenging, because drivers have very heterogeneous preferences in terms of how they trade off the features of various alternatives (for example, regarding the time spent driving, charging costs, waiting times at charging stations, and the facilities provided at the charging stations). The key problem here is eliciting the diverse preferences of drivers, assuming that these preferences are typically not fully known a priori, and then planning stops based on each driver's preferences. Our approach to solving this problem is to develop an intelligent personal agent that learns preferences gradually over multiple interactions. This study proposes a new technique that utilises a small-scale discrete choice experiment as a method of interacting with the driver in order to minimise the cognitive burden on the driver. Using this method, drivers are presented with a variety of routes with possible combinations of charging stops depending on the agent's latest belief about their preferences. In subsequent iterations, the personal agent will continue to learn and refine its belief about the driver's preferences, suggesting more personalised routes that are closer to the driver's preferences. Based on real preference data from EV drivers, we evaluate our novel algorithm and show that, after only a few queries, our method quickly converges to the optimal routes for EV drivers [This paper is an extended version of an ECAI workshop short paper (Shafipour Yourdshahi et al., in: ECAI 2023 workshops, Kraków, Poland, 2023)].

✉ Sebastian Stein
ss2@ecs.soton.ac.uk

Elnaz Shafipour
e.shafipour@soton.ac.uk

Selin Ahipasaoglu
s.d.ahipasaoglu@soton.ac.uk

1 School of Electronics and Computer Science, University of Southampton, Southampton, England, UK

Published online: 30 September 2024

1 Introduction

In order to combat climate change and achieve net-zero emissions, electric vehicles (EVs) are becoming increasingly important, and many governments are supporting their widespread introduction. There is a plan in place by the UK government to stop selling newly manufactured petrol and diesel vehicles by 2035 [17], paralleled by similar objectives within the EU aiming for the same target by 2035 [15]. With the anticipated growth of EVs in the future, there will be a need for better infrastructure which will facilitate the charging of EVs. Nevertheless, according to the United Nations (UN) Sustainable Development Goals (SDGs) #11 and #7, it is essential to achieve a transportation system that is affordable and sustainable by 2035 to achieve a sustainable world.

There is, however, a lot of work to be done in order to develop this infrastructure and, for now, there is a lack of charging stations available to drivers. When it comes to travelling long distances, this can be particularly challenging, since it is likely that drivers have to stop at rapid charging stations, possibly multiple times, on their journey to reach their destination. Hence, the charging problem is still a major obstacle to switching to EVs in the future [10]. In order to gain a deeper understanding of the problems faced by current EV drivers and their behaviour regarding choosing charging stops, we surveyed 1278 EV drivers in April 2022 [19]. In this survey, we found that over a third of the respondents were not satisfied with their charging experience on such long journeys. Another survey conducted by the UK government's Department for Transport [24] supports our findings, indicating that more than a third of EV drivers express dissatisfaction with the public charging infrastructure.

Additionally, we noticed that there was a large variety of preferences of drivers when it came to charging stations. Some prioritised time, whereas others considered the overall cost of charging stations or even the availability of specific facilities as more important.

This paper explores, based on our findings from the survey, how a personal intelligent agent can help EV drivers manage the limited charging infrastructure that is currently available for long trips involving EVs. In particular, since the drivers have very diverse preferences when it comes to choosing charging stations, we take a driver-centric approach and are interested in using artificial intelligence (AI) tools to plan the drivers' stops over long distances, while taking into account their individual preferences (e.g., balancing cost, travel time, charging time, and facilities at charging stations). In spite of the fact that existing work [12, 18, 22] has examined personalised routing, it does not address dynamic preference elicitation.

To minimise reliance on existing data, we assume that the personal agent is not able to obtain any information about the driver's preferences in the beginning, and instead is only able to acquire them through subsequent interactions. A novel method is proposed in this paper, called *Online Estimators for Preference Elicitation (OEPE)*. The objective of this method is to find a route with charging stops that are aligned with the driver's preferences (i.e., a route that will maximise the utility of the driver). As we demonstrate in the results of our research, it is possible to learn a driver's preferences well enough with only a few interactions, so that we can suggest a route which is close to the most preferred route of the driver. We specifically contribute to the state of the art by making the following contributions:

1. To enhance the driving experience of electric vehicles, we developed a personalised, driver-centric routing technique that incorporates information about the driver to find

more convenient charging stations. We applied the preference elicitation concept in this domain for the first time.

2. Our research advances the state of the art like Reinforcement learning [25] and Adversarial Learning [13] with the introduction of OEPE (Online Evolution of Preferences Estimation), a pioneering method designed to estimate driver preferences in the absence of any prior information. This innovative approach enables us to formulate initial hypotheses regarding driver inclinations and to iteratively enhance them through ongoing interactions with the driver. What sets OEPE apart is its lightweight nature. Unlike conventional methods reliant on pre-trained models or the retention of knowledge between executions, our algorithm operates by recalculating estimations from scratch with each run. This ensures real-time adaptability and accuracy without the burden of historical data or preconceived biases. In leveraging OEPE, we offer a dynamic and responsive solution for understanding and accommodating driver preferences in diverse contexts that go beyond the limitations of traditional preference learning techniques.
3. In this paper, we implemented Discrete Route Choice (DRC) for the first time to solve the preference elicitation problem in this domain. Using this technique, drivers are offered multiple routes from their origin to their destination. Then, given the number of choices available to them, they choose the one which is most suitable for them. The advantage of this is that we are able to gather more information from drivers with fewer interactions.

We evaluated our algorithms using real-world data which was collected from EV drivers. This data involves drivers' preferred charging points when on long journeys. Through an extensive evaluation, we show that our approach can converge to the optimal route within only a few iterations as a result of the evaluation.

The remainder of the paper discusses related work in Sect. 2, introduces our novel method (OEPE) in Sect. 3, presents the empirical evaluation in Sect. 4, discusses limitations in Sect. 5 and concludes with conclusions and future work in Sect. 6. As soon as the paper is published, all source codes and datasets needed for conducting and analysing the experiments will be publicly available under a license allowing free research use.

2 Related work

User-centric route planning and eliciting human preferences are very active research topics [4, 11, 12, 18, 22]. In particular, in recent works on dynamic user-centric route planning [12, 18, 22], the authors propose personalised route planning algorithms. These algorithms can provide users with routes that meet their needs, assuming these are known a priori.

Unlike their work, our method acquires a user's preferences without assuming prior information about that user. Rather, it builds beliefs about the user's preferences over time through interactions. There is other recent research [23] in which the authors are concerned with selecting an appropriate query at each stage of an interactive preference elicitation process for the user. Their method is to choose a query that minimises max setwise regret. In light of the fact that we currently do not generate questions, and that we already have a few possible stations, their solution does not fit our current work.

Furthermore, recent work [4] looked at obtaining the preferences of the user through interaction. Based on feedback from human preferences, the authors propose an interactive platform for performing grammar-guided symbolic regression. Specifically, they design an interface that provides the user with three distinct ways to state their preferences between multiple sampled symbolic expressions: categorising samples, comparing pairs, and suggesting improvements to a sampled symbolic expression. There are other works [1–3, 5, 11] where the authors suggest pairs to users and ask them to choose one in order to elicit feedback. In contrast, in our work, instead of interacting with users by eliciting preferences over pairs, we do this by using a method similar to the Discrete Choice Experiment (DCE) [6]. DCEs are used to estimate the parameters of a discrete choice model that captures the choice behaviour using the stated preferences of the users. In each experiment, a user performs a choice task, which is choosing one out of a small number of alternatives. Each alternative and each user have observable attributes that are included in the choice model. We use a similar approach called Discrete Route Choice (DRC). We provide the driver with more than two routes with charging stops where each choice has summarised information about the route and charging stops and allow drivers to choose their preferred one. With this method, we are able to elicit the preferences of the drivers with fewer interactions.

3 Methodology

To model our research, we assume that there is an EV driver who has some preferences for choosing charging stops and depending on the destination, the driver can stop at multiple charging stations on the route to charge his/her car. It is important to note that each selected station has different specifications, such as the cost of charging, and the speed of charging. Depending on which stations are chosen, driving time, waiting time, fees for charging, and access to facilities will differ. Therefore, based on the driver's preferences, the optimal stop would be different. We assign an intelligent agent to each driver taking into account that these preferences are initially unknown to the agent. The agent has initially a hypothesis about the driver's preferences and improves its belief after interacting with the driver multiple times. In more detail, Fig. 1 shows the interaction between the agent and the driver. The agent will keep suggesting some routes with different charging stops depending on the latest perception of the driver's preferences and based on the driver's chosen origin and destination. When the driver picks the route that is most convenient for them among all choices the personal agent updates its belief about the driver's preferences upon receiving

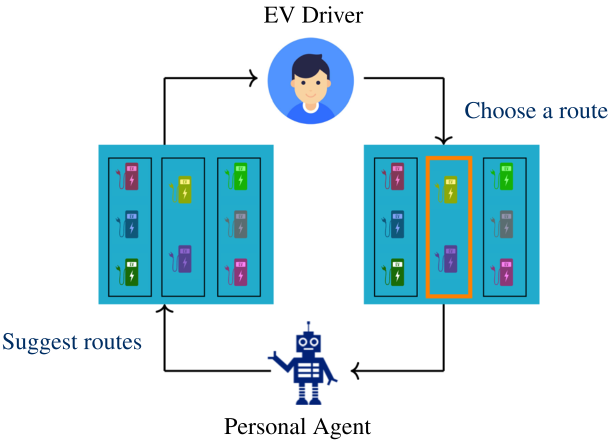


Fig. 1 According to the latest understanding of the driver's preferences, the agent suggests some routes for the journey. The agent updates its belief about the driver's preferences when the driver selects a route

the latest choice. For this method, we assume that $\mathbf{R}_{ab}$ is a set of all possible routes considering charging stops between the given origin a and destination b of the driver. Each $r \in \mathbf{R}_{ab}$ is defined with its feature $\mathbf{X}_r = \{x_1, x_2, ..., x_n\}$ where x_i is the i^{th} feature of the route r (related to EV charging stops and the route itself). As mentioned earlier, these features can include statistics such as mean values and deviations of charging costs, charging speed, waiting times, and availability of restaurants, restrooms or childcare facilities during the entire journey. Additionally, the marginal contribution of each feature to the driver's utility is denoted by weight $\mathbf{W} = \{w_1, w_2, ..., w_n\}$. Each element w_i is a real number between -1 and 1 expressing the importance of the respective feature to the driver. Here, we assume that each driver's preferences can be described by a utility function u:

$$u(\mathbf{W}, \mathbf{X}_r) = w_1x_1 + w_2x_2 + \cdots + w_nx_n.$$

To elicit the driver's preferences and offer better routes, in this paper, we introduce our novel method, *Online Estimators for Preference Elicitation (OEPE)*. Using this method, the most appropriate route is determined based on the driver's preferences regarding charging stop specifications on the route between the origin and destination. This is a driver-in-the-loop method that learns the driver's preferences interactively to identify the optimal route. The main idea behind our algorithm is to keep a set of *estimators*, which are the most probable hypotheses that are continuously revised over time. Each estimator contains potential weights $\mathbf{W}$ for calculating the utility u. These weights are used to predict the route selected by the driver from a given list of route choices. Estimators that are not able to make accurate predictions are removed, and replaced by estimators that are created using successful ones as a basis, or purely random selection.

OEPE has some fundamental concepts that are applied during preference elicitation. In order to provide a proper understanding of the method, we will first introduce the basics of it in Sect. 3.1 and, following that, we will explain the algorithm for eliciting preferences using OEPE in Sect. 3.2 in detail.

3.1 OEPE fundamentals

Set of estimators In OEPE, the personal agent keeps a set, $\mathbf{E}$, of N estimators. As we can see in Fig. 2, an estimator e is a tuple: $\{\mathbf{W}_e, \zeta_e, f_e\}$, where:

- $\mathbf{W}_e$ is a vector of estimated weights of the driver's utility function u;

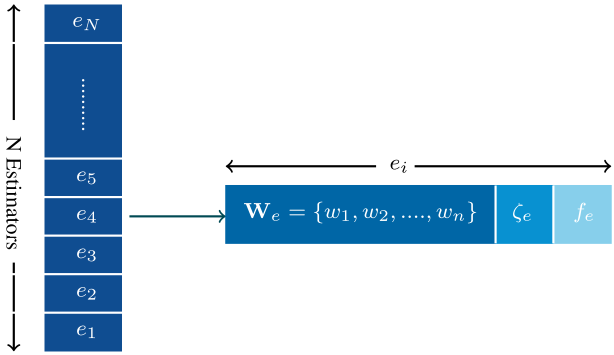


Fig. 2 Set of estimators and the structure of each estimator

- ζ_e holds the number of times that e was successful in predicting the chosen route by the driver;
- f_e keeps the count of failures in identifying the chosen route.

The estimators are initialised at the beginning of the process and evaluated whenever a route is chosen by a driver. The estimators that are not able to make accurate predictions after several trials are removed and replaced by estimators that are created using successful ones as a basis, or purely randomly, in a fashion inspired by genetic algorithms [9].

List of route choices As we mentioned before, based on the specified origin a and destination b, we have $\mathbf{R}_{ab}$ as possible routes from a to b. At each interaction with the driver, the personal agent recommends a set of route choices $\mathbf{R}^*_{ab} = \{r_1, r_2, ..., r_\theta\}$ selected from $\mathbf{R}_{ab}$ to the driver. According to the driver's preference, he/she chooses a route r^* that maximises the utility function. The process of generating $\mathbf{R}_{ab}$ and $\mathbf{R}^*_{ab}$ will be explained in more detail in Sect. 3.2, Algorithms 2 and 3. As part of OEPE, we keep a list of all routes proposed to the driver $\mathbf{R}^*_{ab}$ in all interactions as well as the route selected by the driver r^* in $\mathbf{C}$:

$$\mathbf{C} = \{(\mathbf{R}^*_{ab_1}, r^*_1), (\mathbf{R}^*_{ab_2}, r^*_2), ..., (\mathbf{R}^*_{ab_k}, r^*_k)\}$$

The reason for this is to check the history of suggested routes and select one by the driver for validating each estimator of the $\mathbf{E}$.

Bags of successful weights To predict $\mathbf{w}_e$ precisely and with less interaction, we define bags of successful weights. Each bag $\mathbf{B}_i$ keeps a set of each parameter w_i of vector $\mathbf{w}_e$ separately so that it can be used in the future for generating new estimators. Figure 3 shows an example of bags for each element of vector $\mathbf{w}_e$. Whenever $\mathbf{w}_e$ successfully estimates the correct route selected by the driver, we store each w_i of $\mathbf{w}_e$ in the corresponding bag. If one w_i is successful many times, it will be kept in the corresponding bag repeatedly. Therefore, the chance of selecting it for generating a new estimator will increase. Details of how we use bags in the generation step are mentioned in Algorithm 6.

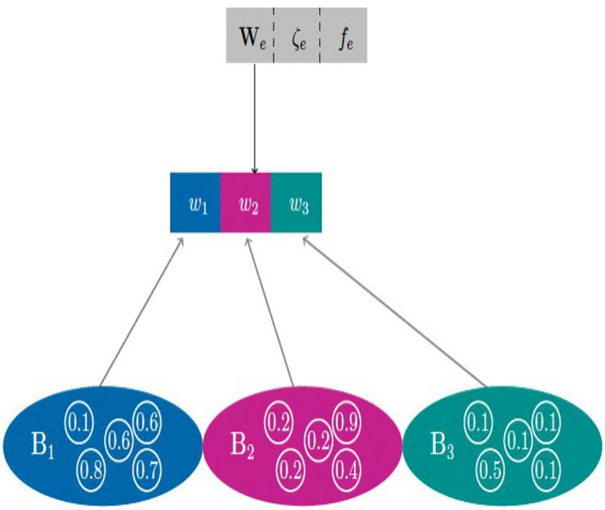


Fig. 3 The length of $\mathbf{W}_e$ is three so we have three sets of bags for each $\mathbf{w}_i$. The numbers inside the bags are those values that predicted the correct route selected by the driver successfully

3.2 Process of eliciting preferences

The process of eliciting preferences in OEPE has three main steps: *Initialisation*, *Evaluation* and *Generation* as shown in Fig. 4.

In more detail Algorithm 1 illustrates the process of interacting with the driver and eliciting his/her preferences.

Algorithm 1 Eliciting Preferences

1: **procedure** ELICITINGPREFERENCES($\theta, \sigma, N, m, \nabla$)
2: $\mathbf{E} \leftarrow Initialisation(N)$;
3: $\mathbf{C} \leftarrow \emptyset$;
4: $\mathbf{B} \leftarrow \emptyset$;
5: **repeat**
6: $a, b \leftarrow GetOriginAndDestination()$; ▷ Origin is a and destination is b.
7: $\mathbf{R}_{ab} \leftarrow GenarateRoutes(a, b, \Lambda, \nabla)$;
8: $\mathbf{W}_p \leftarrow PredictWeights(\mathbf{E})$;
9: $\mathbf{R}^*_{ab} \leftarrow GenerateChoices(\mathbf{R}_{ab}, \mathbf{W}_p, \theta)$;
10: $r^* \leftarrow GetSelectedRoute(\mathbf{R}^*_{ab})$;
11: $c \leftarrow (r^*, \mathbf{R}^*_{ab})$;
12: $\mathbf{C} \leftarrow \mathbf{C} \cup \{c\}$;
13: $\mathbf{E}_t, \mathbf{B} \leftarrow Evaluation(\mathbf{E}, \mathbf{C})$;
14: $\mathbf{E} \leftarrow Generation(\mathbf{E}_t, \mathbf{B}, \mathbf{C}, m)$;
15: **until** σ *iterations*
16: **end procedure**

The first step of the Algorithm 1 is *Initialisation* (Line 2) which is called to generate and initialise N estimators. As there is no prior information about the driver's preferences, all weights w_i of $\mathbf{W}_e$ for each estimator are initialised with a random value drawn uniformly at random from $[-1, 1]$. Finally, both ζ_e and f_e are set to zero. Then the interaction with the driver will be started for σ times and assume that in each interaction, the driver travels from a different origin a to a different destination b and will be asked to enter by the driver using *GetOriginAndDestination* Function (Line 6). Later, the *GenarateRoutes* function generates Λ random routes $\mathbf{R}_{ab}$ from a to b considering the stations on the route to charge. Details can be found in Algorithm 2.

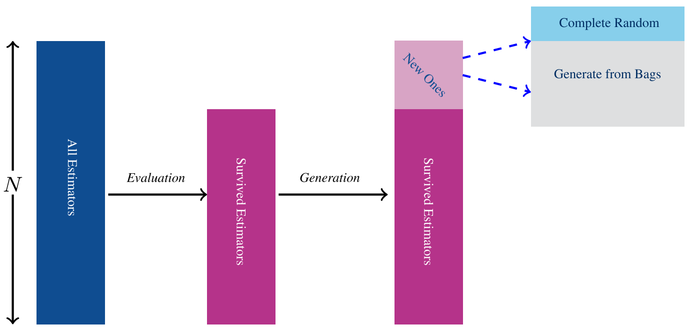


Fig. 4 Process of eliciting the preferences

Algorithm 2 Generate random routes

1: **procedure** GENARATEROUTES(a, b, Λ, ∇)
2: $\mathbf{R}_{ab} \leftarrow \emptyset$;
3: **while** $|\mathbf{R}_{ab}| = \Lambda$ **do**
4: $\alpha \leftarrow a$;
5: $r \leftarrow \emptyset$;
6: **repeat**
7: $r_s \leftarrow FindShortestRoute(\alpha, b)$;
8: $\mathbf{S} \leftarrow GetReachableStations(r_s, \nabla)$;
9: $s \leftarrow FindStation(\mathbf{S})$;
10: $\alpha \leftarrow GetLocation(s)$;
11: $r \leftarrow r \cup \{s\}$;
12: **until** $reachdestination$
13: $\mathbf{R}_{ab} \leftarrow \mathbf{R}_{ab} \cup \{r\}$;
14: **end while**
15: **return** $\mathbf{R}_{ab}$;
16: **end procedure**

In Algorithm 2, the first two inputs are the origin a and destination b of the driver. Moreover, Λ is the number of routes we will generate with this function. The other input of the function ∇ defines the maximum distance from the shortest path between a and b that is where we could look for stations. At Line 7, we first find the shortest path r_s between the moving point of the car α to the destination b. Initially, α is set to the origin a of the driver at Line 4 and updated later with the location of the station s in Line 10 where the car is supposed to be charged and move to the next destination. To find the shortest path, we used A^* algorithm [8], the fastest algorithm among all the algorithms such as Dijkstra [7] and the Manhattan distance is applied as the heuristic function. After getting the shortest path r_s between α and b, we will seek some stations around it. At Line 8, we find all stations $\mathbf{S}$ that are within ∇ distance of r_s. Based on the car's current state of charge, we calculate (Line 9) all stations that can be accessed from α among $\mathbf{S}$. We assume that there is at least one station the car can reach. We randomly select one station s from all reachable stations. After getting to that point, we will presume that the car is recharged to a certain value and consider that station s as the next moving point. This process will be repeated until it reaches the destination. All stations selected will be considered as one route. We repeat this process Λ times.

Later, using the *PredictWeights* function in Line 8 of Algorithm 1, we get the predicted weights $\mathbf{W}_p$ based on the current set of estimators $\mathbf{E}$, which will be explained in more detail further down. As we proceed to Line 9, the function *GenerateChoices* is used to find θ ($\theta < |\mathbf{R}_{ab}|$) possible routes among all possible routes by evaluating them based on the latest elicited preferences $\mathbf{W}_p$ of the driver. Therefore, the function gets randomly generated routes $\mathbf{R}_{ab}$ between a and b as well as $\mathbf{W}_p$. The function is explained in Algorithm 3, in which we calculate the utility u of each $r \in \mathbf{R}_{ab}$ as follows:

$$u = \sum_{i=1}^{n} w_{pi} x_{ri}$$

and then the route with maximum u will be considered as the estimated route r_e in Line 2.

Algorithm 3 Generate Route Choices

procedure GENERATECHOICES($\mathbf{R}_{ab}, \mathbf{W}_p, \theta$)
  $r_e \leftarrow \arg\max_{r \in \mathbf{R}_{ab}} (\sum_{i=1}^{n} w_{pi} x_{ri})$;
  $\mathbf{R}_r \leftarrow$ $GetRandomRoutes(\theta - 1, \mathbf{R}_{ab} \setminus \{r_e\})$;
  $\mathbf{R}^*_{ab} \leftarrow \mathbf{R}_r \cup \{r_e\}$;
  return $\mathbf{R}^*_{ab}$;
end procedure

After selecting the route with maximum utility, we will select another $\theta - 1$ routes from $\mathbf{R}_{ab} \setminus \{r_e\}$ randomly. Following that, the agent will interact with the driver and run a DRC with *GetSelectedRoute*. For this DRC, there are θ routes as choices with information about the routes' attributes and their related values. From these routes, the driver chooses his/her preferred route r^*. Later in Lines 11 and 12 of Algorithm 1, we update the list of route choices $\mathbf{C}$ with the offered routes to the driver $\mathbf{R}^*_{ab}$ and the preferred one r^* by the driver. After getting the route chosen by the driver, we will evaluate the estimators. Algorithm 4 presents the process for evaluating estimators. The key objective of this step is to find estimators that can estimate the chosen route correctly.

Algorithm 4 Evaluating Estimator

1: **procedure** EVALUATION($\mathbf{E}, \mathbf{C}$)
2: $\mathbf{F} \leftarrow \emptyset$; ▷ $\mathbf{F}$ is a set that contains failed estimators
3: **for each** $e \in \mathbf{E}$ **do**
4: $\zeta_e \leftarrow CalculateSuccess(\mathbf{C}, \mathbf{W}_e)$;
5: **if** $\zeta_e > 0$ **then**
6: **for** each $w_i \in \mathbf{W}_e$ **do**
7: $\mathbf{B}_i \leftarrow \mathbf{B}_i \cup \{w_i\}$;
8: **end for**
9: **else**
10: $f_e \leftarrow f_e + 1$;
11: **end if**
12: **if** $f_e > \xi$ **then**
13: $\mathbf{F} = \mathbf{F} \cup \{e\}$;
14: **end if**
15: **end for**
16: $\mathbf{E}_t = \mathbf{E} - \mathbf{F}$;
17: $\mathbf{B} \leftarrow \{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_n\}$;
18: **return** $\mathbf{E}_t, \mathbf{B}$;
19: **end procedure**

As the next step, we will need to evaluate the estimators of $\mathbf{E}$ and see which one is closest to the driver's preference. After each interaction with the driver and getting the selected route, we start evaluating our estimators and updating the success rate ζ_e and failure rate f_e of each estimator using Algorithm 4. To calculate ζ_e the increment is not simply done by $\zeta_e \leftarrow \zeta_e + 1$ and the *CalculateSuccess* function in Line 4 is applied for updating the ζ_e. The *CalculateSuccess* (Algorithm 5) function calculates by looking at the history of the interactions between the agent and the driver which are stored in $\mathbf{C}$. It checks each choice c in $\mathbf{C}$ and identifies what is the optimal route r^*_e among $\mathbf{R}_c$ by assuming $\mathbf{W}_e$ as the driver's true weight. If the preferred route by driver r^*_c is equal to the estimated route using $\mathbf{W}_e$ (r^*_e) then the ζ will be increased. This process will be repeated for all choices in $\mathbf{C}$ to get the total ζ. The value returned from the *CalculateSuccess* function will be stored in

ζ_e. In case there is at least one successful estimation for e, each w_i in the $\mathbf{W}_e$ vector will be stored in a respective bag $\mathbf{B}_i$, which is mentioned in Line 6 and 7. The union ($\cup$) which is applied in the equation means that new weights will be added to the bag with repetition. If a weight succeeds many times, it will appear in the bag with the same number of successes, so the chance of selecting it will be higher.

Algorithm 5 Evaluate Old Choices

```
1:  procedure CALCULATESUCCESS(C, W)
2:      ζ ← 0;
3:      for each c ∈ C do
4:          r*_e ← arg max_{r∈R_c} (Σ_{i=1}^{n} w_i x_{ri});
5:          if r*_c = r*_e then
6:              ζ ← ζ + 1;
7:          end if
8:      end for
9:      return ζ;
10: end procedure
```

If the estimator e could not predict the correct route among previous choices, then f_e is increased (Line 10). The first failure for the estimator e would not be the reason to remove it and it will be given more chances since it may still hold correct weights. Consequently, there is a *threshold* ξ for the removal, and if f_e is greater than ξ, the estimator e will be removed from $\mathbf{E}$.

After the *Evaluation* step, $\mathbf{E}_t$ will be the new surviving estimators as some of them were removed due to multiple failures of the estimators and now we start *Generation* step. In this step, the aim is to generate new estimators to increase the size of the set $\mathbf{E}$ back to N again. This means $N - |\mathbf{E}_t|$ new estimators need to be generated. Unlike the *Initialisation* step, new estimators are not only created with random values but a proportion of them are generated using previous successful weights from the *bags* $\mathbf{B}$. Accordingly, a new combination of weights that had at least one success in the previous steps can be utilised in generating new estimators.

Algorithm 6 Generating New Estimators

```
1:  procedure GENERATION(E_t, B, C, m)
2:      n ← 0;
3:      η ← (N − |E_t|) × m;
4:      while |E_t| < N do
5:          for all w'_i ∈ W' do
6:              if n < η  OR |B_i| = 0 then
7:                  w'_i ← random value from U(−1, 1);
8:              else
9:                  w'_i ← random value from B_i;
10:             end if
11:         end for
12:         hist_success ← CalculateSuccess(C, W');
13:         if hist_success > 0 then
14:             W_e ← W';
15:             ζ_e ← hist_success;
16:             f_e ← 0;
17:             E_t ← E_t ∪ {e};
18:             n ← n + 1;
19:         end if
20:     end while
21:     E ← E_t;
22:     return E;
23: end procedure
```

More detail of the process of generating new estimators is indicated in Algorithm 6. The main part of producing new estimators is creating a new weight vector $\mathbf{W}'$. The process of creating all new weights $\mathbf{W}'$ are shown in Lines 5–10 of the Algorithm 6. Weights for a proportion $(N - |\mathbf{E}_t|) \times m$ (where $m \in [0, 1]$) of the new estimators will be randomly sampled from a uniform distribution $\mathcal{U}(-1, 1)$. The other proportion $(N - |\mathbf{E}_t|) \times (1 - m)$ will be created as a new mixture from the respective bags, which contain previously winning weights. If all bags are empty, then all parameters will be random. Before creating a new estimator e', in Line 12 and 13 of the Algorithm 6, the *CalculateSuccess* function (Line 12) is employed here to check if the recently generated weights $\mathbf{W}'$ would have at least one success across the choices list so far. Checking the previous successes improves the algorithm since it decreases the likelihood of wasting an estimator with weights $\mathbf{W}'$ that would not be able to make any correct prediction in the previous steps. As a result, if the output of the function is zero, $\mathbf{W}'$ will be discarded. Otherwise, it will be considered as the weight $\mathbf{W}_{e'}$ of the new estimator e'. Consequently, $\zeta_{e'}$ will be assigned with the output of the *CalculateSuccess* function and $f_{e'}$ will be assigned to zero. In the end, the created e' will be added to $\mathbf{E}_t$, and the process repeats until $|\mathbf{E}_t| = N$.

Additionally, we need to have the estimated weights for u at each iteration to be able to guess the route that the driver would prefer. In this case, we would be able to recommend routes that are close to the driver's preferred route. To enable the personal agent to make better recommendations, it is necessary to estimate the weights of the driver's utility function u. For estimating the weights, we apply the *PredictWeights* function (Line 8). In this function, we get the weighted average of the predicted w_e in which weights are ζ_e of each estimator e. We used the weighted average method, as it is a valuable tool for estimating values in situations where multiple estimates with different reliability levels are available. In this way, we let more reliable estimates contribute more to the final result, while less reliable estimates give less influence. Therefore, it gives more weight to the more trustworthy estimates and less weight to those that are less reliable.

$$W_p = \frac{\sum_{e \in \mathbf{E}} \zeta_e w_e}{\sum_{e \in \mathbf{E}} \zeta_e}$$

The number of iterations will continue for σ times.

4 Evaluation

In this section, we assess the efficacy of our innovative approach, *Online Estimators for Preference Elicitation (OEPE)*. We undertake a comparative analysis between OEPE and two conventional classification methods: Decision Trees (DT) [16], Discrete Choice Model (DCM) [21] Deep Q-Networks (DQN) [26]. This evaluation aims to evaluate OEPE's performance in accurately estimating the route preferred by the driver, in comparison to established algorithms.

4.1 Benchmarks

We assume the driver's preferences will be elicited using either DCM/DT/OEPE for each scenario. Algorithms 7 and 8 show how we elicit preferences using DT and DCM. For both algorithms, all weights w_i of $\mathbf{W}_e$ are initialised with a random value from the uniform distribution $\mathcal{U}(-1, 1)$.

Algorithm 7 Eliciting Preferences With DT

1: **procedure** ELICITINGPREFERENCESDT(θ, σ)
2: $\quad\mathbf{W}_p \leftarrow RandomInitialisation()$
3: $\quad\mathbf{X}, \mathbf{Y} \leftarrow \emptyset$;
4: $\quad$**repeat**
5: $\qquad a, b \leftarrow GetOriginAndDestination()$;
6: $\qquad \mathbf{R}_{ab} \leftarrow GenarateRoutes(a, b, \Lambda)$;
7: $\qquad \mathbf{R}^*_{ab} \leftarrow GenerateChoices(\mathbf{R}_{ab}, \mathbf{W}_p, \theta)$;
8: $\qquad r^* \leftarrow GetSelectedRoute(\mathbf{R}^*_{ab})$;
9: $\qquad \mathbf{X} \leftarrow \mathbf{X} \cup \mathbf{R}^*_{ab}$;
10: $\qquad \mathbf{Y} \leftarrow \mathbf{Y} \cup \{r^*\}$;
11: $\qquad \mathbf{W}_p \leftarrow DT(\mathbf{X}, \mathbf{Y})$;
12: $\quad$**until** σ $iterations$
13: **end procedure**

In Algorithm 7 (similar to Algorithm 1), from Line 5 to Line 10, we generate $\mathbf{R}^*_{ab}$ route choices and then get the selected route by the driver r^*. The next goal of this algorithm is to use the data gathered so far to create a model that predicts the driver's preferences. Thus, the dataset is empty at the beginning of the process and increases gradually with each interaction with the driver. Here, we add $\mathbf{R}^*_{ab}$ and r^* to the $\mathbf{X}$ and $\mathbf{Y}$ sets of the dataset respectively. After updating the dataset, we applied *DecisionTreeClassifier* of *sklearn* and fitted the model based on $\mathbf{X}$ and $\mathbf{Y}$ and then predicted the $\mathbf{W}_p$. However, in Algorithm 8, after generating $\mathbf{R}_{ab}$, we run DCE for n times. After gathering n selected choices among the suggested routes by the driver we estimate new weights using Maximal Likelihood Estimation. For this, we used the *pylogit* library [14].

Algorithm 8 Eliciting Preferences With DCM

1: **procedure** ELICITINGPREFERENCESDCM(θ, σ, n)
2: $\quad\mathbf{W}_p \leftarrow RandomInitialisation()$
3: $\quad\mathbf{C} \leftarrow \emptyset$;
4: $\quad$**repeat**
5: $\qquad a, b \leftarrow GetOriginAndDestination()$;
6: $\qquad \mathbf{R}_{ab} \leftarrow GenarateRoutes(a, b, \Lambda)$;
7: $\qquad$**while** n choice set **do**
8: $\qquad\quad r^*, \mathbf{R}^*_{ab} \leftarrow RunDCE(\mathbf{R}^*_{ab}, \mathbf{W}_p)$;
9: $\qquad\quad \mathbf{C} \leftarrow \mathbf{C} \cup (\{r^*\}, \mathbf{R}^*_{ab})$;
10: $\qquad$**end while**
11: $\qquad \mathbf{W}_p \leftarrow DCM(\mathbf{C})$;
12: $\quad$**until** σ iterations
13: **end procedure**

To enable comparative analysis between our algorithm and RL techniques, we define the state space as including all potential charging stations, along with the current state of charge of the EV, as well as the origin and destination of the driver. Since we do not know the preferences of the driver and there can be an infinite combination of weights, our model has an infinite action space. Thus, we used the linear DQN function approximation to solve the problem by assuming that the true reward function is the driver's utility function which we defined as a linear function in our method.

4.2 Setup

To evaluate our algorithm, we used data from the survey described in Sect. 1 for the experiments. The survey was conducted both online and in person. To conduct the in-person survey, our team visited some of the UK's rapid charging stations, such as Cobham, Fleet, and Winchester, and went to the Fully Charged LIVE event in Farnborough. Our team interviewed electric vehicle drivers while they were charging their vehicles. As part of our interview process, we asked multiple questions, such as what challenges they faced and how they overcame them. Moreover, we asked about their priorities in selecting a charging station. We presented the following options to them:

1. High speed of charging
2. Overall low charging cost
3. Part of a charging network that I am a member of
4. Provisions for drivers with accessibility needs
5. Minimal deviation to my route
6. Easy payment methods (for example, access to contactless)
7. Pricing transparency
8. Safe location (for example, well-lit, likely to be busy)
9. Positive online reviews or charging reports
10. Typically not busy
11. Close to food/refreshment facilities
12. Close to shopping facilities
13. Location with baby change facilities
14. Sheltered charge point
15. Location with restrooms
16. Close to the playground
17. Real-time availability information.

Then, we asked them to choose up to 10 reasons for choosing a particular charging station on a long journey. As a next step, they were asked to rank the chosen options. As a result, we know how important each feature is to each user. For our evaluation, we assume that these participants are the EV drivers in our system. Thus, to derive the weights of each route feature preferred by the drivers, we converted their ranked preferences into weights using the following formula:

$$w_i = \frac{1}{p_i}, \tag{1}$$

where p_i denotes the rank order of features i for a particular participant. As an example, if a participant selected: 1, 2, 6, 9, 10, 11, 13, 15 as their main feature when choosing a charging stop and ranked them as follows: 10, 2, 1, 13, 11, 6, 15, 9. For this research, we only considered High speed of charging (time); Overall low charging cost (cost); Close to food/refreshment facilities (access to restaurants); Location with restrooms (access to restrooms). So, according to the above rankings, the rank for *cost* is 2nd, for *time* is 3rd, for *access to restaurants* is 5th, and for *access to restrooms* is 7th. Accordingly based on the above equation the weights for these features are $\frac{1}{5}$, $\frac{1}{3}$, $\frac{1}{5}$, and $\frac{1}{7}$. We employed these weights for route features and linked them as follows: the cost was mapped to the total charging cost of the route, speed was associated with the total driving time of the route (including driving time and total charging time), access to restaurants was correlated with the total number of food facilities on the route, and access to restrooms was tied to the number of restrooms available on the route.

In addition to considering the weight of each feature for the EV driver, we needed to simulate the road and charging stops. To evaluate our method and make comparisons, we initially defined a grid, assuming that various stops are located in different cells of the grid. Also, assuming that the source and destination are situated somewhere along the border of the grid. Here is how we set values to each station's features: Speed is a number between 10 and 300 kW; Cost is a number between 1 p/kWh and 100 p/kWh; 0 or 1 for having a restaurant or not; 0 or 1 for having a restroom or not. Therefore, the total charging cost for each route will be the aggregate of the charging cost at each station along that route, and the total driving time for that route will include the sum of the driving time plus the charging time. Regarding amenities such as food facilities and restrooms, we will count the number of available amenities at the stations on the selected route. Finally, to calculate the utility of the route, considering the variance in ranges among these values, we will normalise them before multiplying each with its respective weight. The charging cost and driving time are normalised based on the lowest and highest possible charge cost and charging speed (assuming constant driving time). To normalise the facilities of each route, we assume that all stations along the route have that specific facility and that there are no facilities along the route as the minimal value of them along the route. For this, we implemented this solution by applying the *keras-rl* library in Python. We constructed a grid environment where we positioned charging stations, as well as source and destination points. To thoroughly evaluate our method, we tested it across two grid sizes: 10×10 and 20×20, each with 10 distinct scenarios. We positioned 20 stations within the smaller grid (10×10) and 80 stations within the larger grid (20×20). These stations were located in cells that were not designated as drivers' starting points or destinations. In addition, the car's battery level when starting the trip is randomly chosen between 50 and 90 percent and the maximum range is set to a random value between 30 and 300 miles. For OEPE configuration, $N = 100$, $\theta = 3$, $\sigma = 20$, $\Lambda = 10$, $\xi = 2$ and $m = 0.2$. All these 200 scenarios were run 20 times for 1000 survey participants considering their preferences.

4.3 Results

For each run, we tracked the estimated weights of the drivers, the chosen route and the number of interactions with the driver. We aggregated the results and plotted the average and the confidence interval ($\rho = 0.05$). Figure 5 shows the number of interactions that each method needs for its preference error to converge to 0.01 for different sizes of scenarios. As we can see in both Figures, OEPE is significantly better than other methods and can learn the preferences in less number of interactions with the driver compared to other methods.

To plot the error in learning the preferences of drivers, we show the average error across all weights by evaluating the mean absolute error of the weights. Moreover, since we aggregate multiple results, we calculate and plot the average error. The average error across all weights for all drivers and all scenarios is shown in Fig. 6. As we see, OEPE the error of preferences gets closer to the true preferences of the driver after a couple of iterations. Figure 6 shows that our weight estimation error is consistently lower than the other algorithms starting with the second iteration, and it (almost) monotonically decreases with more iterations. On the other hand, DCM, DQN, and DT do not show any signs of convergence as the number of iterations increases.

However, we know the actual weights $\mathbf{W}_t$ of the driver for each scenario. Hence, using the driver's utility, we can find out which route would be the right choice r_t for the driver among all possible routes $\mathbf{R}_{ab}$ for a given origin and destination:

$$r_t \leftarrow \arg\max_{r \in \mathbf{R}_{ab}} \left(\sum_{i=1}^{n} w_{ti} x_{ri} \right)$$

Therefore, at each iteration, we compared the utility of the chosen route r^* and r_t:

$$d = |u(\mathbf{W}_t, \mathbf{X}_{r_t}) - u(\mathbf{W}_p, \mathbf{X}_{r^*})|$$

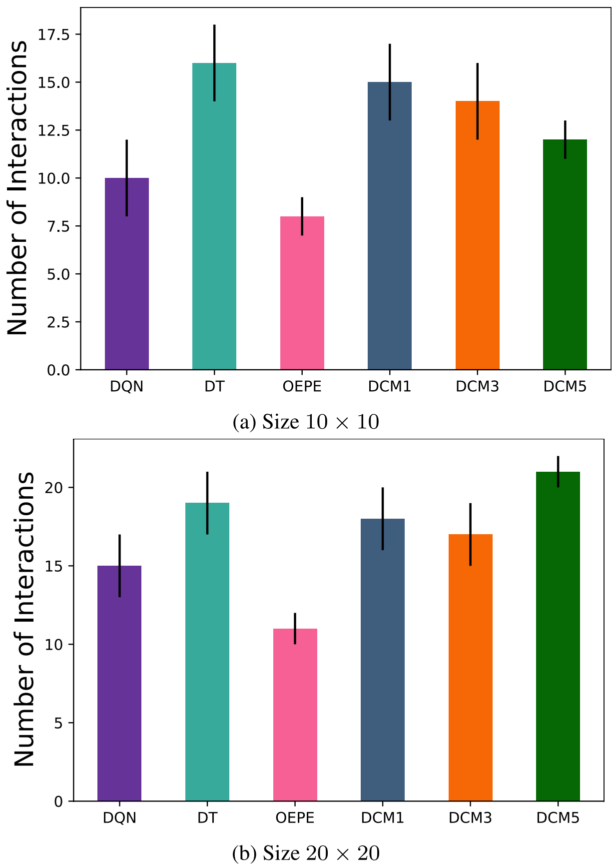


Fig. 5 Number of Interactions needed to converge preference error to 0.01 with error bars for 95% confidence interval for 10×10 and 20×20 scenario sizes

Figure 7 shows the percentage of the distance of the selected route r^* at each interaction compared to the best route r_t at each interaction. After more interactions with the driver, our distance quickly surpasses the other algorithms in the mean, becoming significantly better after a few iterations. This is because, in OEPE, we begin by assuming a hypothesis about what preferences might be. However, for others, there is no initial assumption for preferences and they need more data to predict the correct weights.

5 Limitations and discussion

There are some limitations to our work, which we discuss in the following. First, the experimental settings utilised a set of predefined preferences based on our initial survey data, which may not encompass the full diversity of preferences found in real-world scenarios. While the OEPE method is designed to adapt to a variety of preferences, the initial scope may limit its applicability to edge cases or uncommon preferences. Additionally, our experiments were conducted with a limited number of participants and routes. Although the results were promising, larger-scale studies are necessary to validate the findings across a more extensive and diverse dataset, reflecting a broader range of driving conditions and user preferences.

Fig. 6 Number of Interactions needed to converge preference error to 0.01 with area for 95% confidence interval for 10×10 and 20×20 scenario sizes

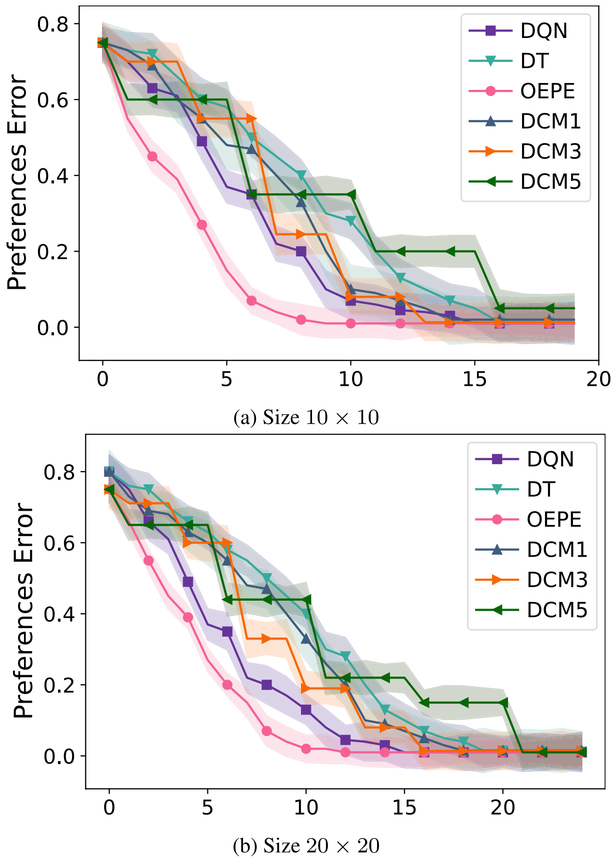
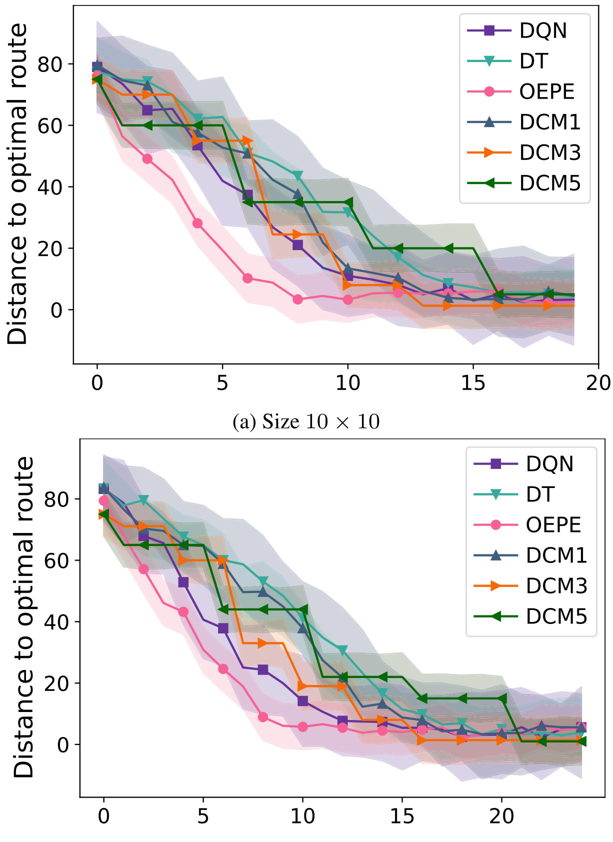


Fig. 7 Number of Interactions needed to converge to optimal route with 95% confidence interval for 10×10 and 20×20 scenario sizes



(a) Size 10×10

(b) Size 20×20

Regarding the environment, the experimental scenarios were conducted in a simulated environment that may not capture all the nuances of real-world driving, such as unexpected traffic conditions, roadworks, or the availability of charging stations. These factors can significantly influence route and charging stop decisions, and their impact should be evaluated in future real-world tests.

For driver interaction, while the discrete choice experiment method effectively minimised cognitive load, the interaction model in a controlled experiment might differ from real-life scenarios where drivers may face additional distractions or stress. Further studies should explore how these real-world factors affect the interaction and decision-making process.

6 Conclusions

Currently, the biggest barrier preventing people from making the switch to electric vehicles is the charging process. Moreover, there can be a wide range of charging stations in terms of type, price, and speed. These stations may also be near restaurants, restrooms, childcare facilities, or other amenities. Our survey of EV drivers revealed that drivers have different preferences regarding charging stops.

We aimed to facilitate the uptake of electric vehicles by recommending drivers with routes and charging stops that are aligned with their preferences. Online Estimators for Preference Elicitation (OEPE) allows us to elicit drivers' preferences without prior knowledge. To reduce interaction with the driver and get more information at the same time, we used DRC. We used real-world data collected through a survey conducted in 2022 as the basis for our analysis. We have obtained ethical approval for this study through the University of Southampton ethical review process under the ERGO number 70451.A1. We have demonstrated that we can learn the preferences of the driver using this data, and then suggest an optimal route using only a limited number of iterations. As a result of this solution, EV drivers will experience less range anxiety and be able to charge their electric cars seamlessly while on the road. In this way, they will be able to rest assured that using this technique will enable them to reach their destination in a timely manner. No matter what their car's battery level is. Our solution will help reduce charging issues, thereby motivating people to switch to EVs. This will contribute to the UN Sustainable Development Goals by creating a sustainable transportation system (SDG#11). Moreover, there would be a strong incentive for people to buy second-hand EVs, which would be cheaper but have a smaller range. This means we will increase the population who can afford an EV which will ensure universal access to affordable, reliable and modern energy services (SDG #7). Furthermore, this would allow people to buy cheaper EVs and not be left behind. As part of our future studies, we will learn the preferences of the driver under various circumstances such as who they travel with, whether they are in a rush to reach their destination, and many other possibilities; we will also focus on recommending dynamic routes to the driver as the features of the road and stations like driving time and waiting time might change during the journey.

In conclusion, while the OEPE method has shown effectiveness in personalising EV charging stop planning and quickly converging to optimal routes in experimental settings, further research is necessary to enhance the real-world applicability of our work. Future work will focus on conducting large-scale real-world trials to validate the experimental findings, exploring a broader range of preferences, and refining the interaction model to better reflect real-life driving conditions. By addressing these challenges, we aim to enhance the robustness and applicability of the OEPE method for widespread use in diverse real-world scenarios.

Acknowledgements The authors acknowledge the financial support received from the Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Acceleration Fellowship on Citizen-Centric AI Systems (EP/V022067/1), the Future Electric Vehicle Energy Networks supporting Renewables (FEVE4R) Grant (EP/W005883/1) and the Public Policy team of the University of Southampton via the New Things Fund.

References

1. Baarslag, T., & Gerding, E. H. (2015). Optimal incremental preference elicitation during negotiation.
2. Bourdache, N., Perny, P., & Spanjaard, O. (2019). Incremental elicitation of rank-dependent aggregation functions based on Bayesian linear regression. In *IJCAI-19-twenty-eighth international joint*

conference on artificial intelligence (pp. 2023–2029). International Joint Conferences on Artificial Intelligence Organization.
3. Brandt, F., Lederer, P., & Suksompong, W. (2022). Incentives in social decision schemes with pairwise comparison preferences. arXiv preprint arXiv:2204.12436
4. Crochepierre, L., Boudjeloud-Assala, L., & Barbesant, V. (2022). Interactive reinforcement learning for symbolic regression from multi-format human-preference feedbacks. In *31st International Joint Conference on Artificial Intelligence (IJCAI 2022)*.
5. Filipczuk, D., Baarslag, T., Gerding, E. H., & Schraefel, M. C. (2022). Automated privacy negotiations with preference uncertainty. *Autonomous Agents and Multi-Agent Systems, 36*(2), 49.
6. Hall, J., Viney, R., Haas, M., & Louviere, J. (2004). Using stated preference discrete choice modeling to evaluate health care programs. *Journal of Business research, 57*(9), 1026–1032.
7. Halpern, J. (1977). Shortest route with time dependent length of edges and limited delay possibilities in nodes. *Zeitschrift fuer Operations Research, 21*, 117–124.
8. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics, 4*(2), 100–107.
9. Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence*. MIT Press.
10. Krishna, G. (2021). Understanding and identifying barriers to electric vehicle adoption through thematic analysis. *Transportation Research Interdisciplinary Perspectives, 10*, 100364.
11. Kuhlman, C., VanValkenburg, M., Doherty, D., Nurbekova, M., Deva, G., Phyo, Z., Rundensteiner, E., & Harrison, L. (2018). Preference-driven interactive ranking system for personalized decision support. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 1931–1934).
12. Li, P., Wang, X., Gao, H., Xiaolong, X., Iqbal, M., & Dahal, K. (2021). A dynamic and scalable user-centric route planning algorithm based on polychromatic sets theory. *IEEE Transactions on Intelligent Transportation Systems, 23*(3), 2762–2772.
13. Lowd, D., & Meek, C. (2005). Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (pp. 641–647).
14. Martín-Baos, J. Á., García-Ródenas, R., & Rodriguez-Benitez, L. (2021). A python package for performing penalized maximum likelihood estimation of conditional logit models using kernel logistic regression. *Transportation Research Procedia, 58*, 61–68.
15. McPhie, T. EU deal to end sale of new CO2 emitting cars by 2035.
16. Myles, A. J., Feudale, R. N., Liu, Y., Woody, N. A., & Brown, S. D. (2004). An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society, 18*(6), 275–285.
17. Pippa Crerar, K. S., & Harvey, F. (2023). Rishi Sunak announces u-turn on key green targets.
18. Pradhan, R., Agarwal, A., & De, T. (2022). A multi-criteria route selection method for vehicles using genetic algorithms based on driverâ€™s preference. *Wireless Personal Communications, 10*(2), 1–20.
19. Shafipour Yourdshahi, E., & Stein, S. (2022). Electric vehicle charging on long journeys: Current challenges and future opportunities.
20. Shafipour Yourdshahi, E., Stein, S., & Ahipasaoglu, S. (2023). Personalised electric vehicle routing using online estimators. In *ECAI 2023 workshops, Kraków, Poland* (**in Press**).
21. Small, K. A. (1987). A discrete choice model for ordered alternatives. *Econometrica: Journal of the Econometric Society, 55*, 409–424.
22. Tiausas, F., Talusan, J. P., Ishimaki, Y., Yamana, H., Yamaguchi, H., Bhattacharjee, S., Dubey, A., Yasumoto, K., & Das, S. K. (2021). User-centric distributed route planning in smart cities based on multi-objective optimization. In *2021 IEEE international conference on smart computing (SMARTCOMP)* (pp. 77–82). IEEE.
23. Toffano, F., Viappiani, P., & Wilson, N. (2021). Efficient exact computation of setwise minimax regret for interactive preference elicitation.
24. UK Department for Transport. Electric vehicle charging research (2022).
25. Wiering, M. A., & Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, learning, and optimization, 12*(3), 729.
26. Xu, P., & Gu, Q. (2020). A finite-time analysis of q-learning with neural network function approximation. In *International conference on machine learning* (pp. 10555–10565). PMLR.