Paper number ITS-TP1942

Easy2Go's Intelligent trip recommendation system

Linglong Meng[1*] (linglong.meng@dfki.de), Stefan Schaffer[1] (stefan.schaffer@dfki.de)
[1] Cognitive Assistants, DFKI, Berlin

Abstract
Easy2Go [1] is a seamless multimodal mobility service through a pay-as-you-go solution supported by EIT Digital. In this paper, we present Easy2Go's intelligent trip recommendation system, an optimised commuter data-based routing service that can be used to learn the actual usage of the traffic system and to automatically create trip recommendations including or avoiding information about frequently used routes. Utilizing transport mode detection technology, we are able to collect commuter route data, from which our system leverages crowd knowledge by machine learning. Our system combines machine learning with a classical routing service to provide an improved trip recommendation. The underlying functional concepts of the overall system is presented as well as details about the used machine learning algorithm.

Keywords:
Intelligent trip recommendation, reinforcement learning, smart mobility, Mobility as a Service

Introduction

Routing services in multimodal public transport networks show a considerable variation between different solutions. Most routing service rely on single mode shortest distance or shortest travel time routes only, which lacks of user preference information. FAVOUR [4] has already approached to provide personalized, situation-aware route proposals based on user profiles, which is initialized at the first stage via prior knowledge on user preferences. However, this approach requires initiative of single user in order to initialize the user profile. Another approach TVPTA by Nuzzolo [5] gives pre-trip and en-route dynamic real-time information to support the user in the dynamic choice of the best path from the personal (dis)utility point of view on a multimodal network. The user can easily access organized information to compare different alternatives of travel recommendation. However, it also takes time and extra effort for the user to make the decision.

Alternatively, machine learning provides another option: leveraging information from the historical commuter trips to improve trip recommendation quality. A realistic assumption is that the historical data of commuter trips contains information regarding the public transportation context and the commuter preference (e.g., changes in public transportation across the time, the traffic density, the landscape along the bus line, etc.). Hence, the crowd-based information about the commuter travel preference can be transferred to the users. Using this information the trip recommendation can suggest automatically generated trip options including e.g. the intentional selection of reliable commuter routes or targeted avoidance of such routes in order to avoid overcrowded means of transport.

As is described in [1], the Easy2Go system uses *Automatic Transport Mode Detection* technology to automatically detect the transport mode by means of manual check-in and *automatic check-out* for pay-as-you-go ticketing. Enabling these functionalities requires to collect the travel data of the users of the system. Based on this point and our preliminary analysis, we propose an intelligent trip recommendation system – a data-based trip recommendation system which combines a machine learning approach with a classical routing service. Instead of creating an initial profile for a user, it requires no manual input about the preference that is being used in order to influence the routing algorithm. Unsupervised machine learning is utilized to adapt the recommendations of the routing system to the behaviour of its users. Our trip recommendation system is setup in two stages. In the **first stage**, which can be considered as a learning stage, a set of historical commuter trip data with a fixed period is fed into the machine learning module in order to learn the travel behaviour of the users in the past. In the **second stage**, the output of the machine learning module will be integrated into the routing engine, namely OpenTripPlanner[1], to produce commuter data-based trip recommendation itineraries. If enough data about the actual usage of the traffic system is collected, the system would also be able to adapt the routing according to temporal changes. Thus allowing for automated detection of regularly occurring load peaks and the automated adaptation of the routing algorithm to e.g. prioritize less loaded routes.

In the following sections, to build a better understand of the trip recommendation module as a service of the overall system, we first introduce the high-level architecture of Easy2Go. Thereafter an overview of the trip recommendation module is given. Next the details of the machine learning approach are explained focusing on two main questions: (1) How can we formulate the machine learning problem? (2) What are the inputs and outputs of the learning model in this domain? This is followed by a description of the system architecture of the trip recommendation module. The last section concludes and highlights future work.

Intelligent Trip Recommendation

In this section, we first give an introduction of the Easy2Go system. Further we explain the functional concept of the trip recommendation system.

---

1 http://www.opentripplanner.org/

The Easy2Go System

Easy2Go provides a seamless multimodal mobility experience to the commuter through a pay-as-you-go app, which was developed and tested during the year 2018 with the support of EIT Digital, the leading European open-innovation and go-to-market organisation. The system consists of five main components: the service platform, a central instance for managing the coordination between all other modules; the trip recommendation and digital travel assistance service component, providing machine learning based route recommendations and alerts; the ticketing engine for issuing tickets and handling the payment process; the check-out recommendation for automatic detection of the trip's end based on transport mode detection and lastly, the CEDUS smart city platform for visualization and analysis of the city's data, including but not limited to collected mobility data.

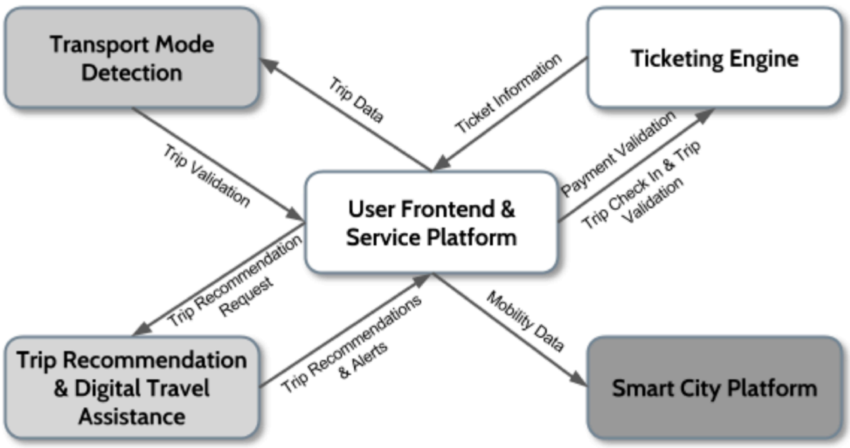

Figure 1 – High-level architecture of the Easy2Go system

The service platform orchestrates the communication between the individual components of the system and the user frontend, which is realized as a part of the existing Boogi app[2].

The high level functional architecture in Figure 1 shows how the single components are interconnected. The platform is connected to a (not shown) routing service, for obtaining available itineraries to a given destination. This information is supplemented by information from the trip recommendation module (Trip Recommendation Request) and presented to the user. In our ITS World Congress 2018 paper the interplay between the single components is depicted in more detail [1]. In this paper we focus on our works in the field of intelligent trip recommendation. In this context it has to be emphasized that the Easy2Go/Boogi application records the trips performed by the users. In order to establish the data-based services delivered by Easy2Go we had to implement the following usage-dependent data protection scheme:

[2] https://www.boogi.fr/easy2go/

- The *Service Platform* as well as the *Ticketing Engine* need personalized data in order to enable the payment process.
- The *Transport Mode Detection* and the *Trip Recommendation & Digital Travel Assistance* component need pseudonymized data as a relation to a specific user profile is not necessary.
- Within the *Smart City Platform* the relation to a specific user may not be desired. Therefore it typically uses anonymized data.

Further details about the data protection concept described in [1]. In the next subsection the working principle of the trip recommendation module is described.

Trip Recommendation Module

In this section we present some preliminary concepts and give an overview of the *Trip Recommendation Module*.

Preliminary Concepts

Definition 1: The public transport network graph: G=(V, E). We model the public transport network as a vertex-edge graph G=(V, E), where V and E are the vertex and edge sets. In the vertex-edge graph, the vertex and edge represent relatively the stop (or station) in the public transport network and the path between stops.

Definition 2: Route: A route R is a continuous travelling path. We use a sequence $[p_1, p_2, \cdot\cdot\cdot, p_n]$, which consists of an origin, a destination, and a sequence of intermediate stops in-between. A route represents a single data item of the trip data set collected by the Easy2Go system.

Definition 3: Default weight of an edge: w_d is a default weight that is used in the classical routing service, denoting the travel time.

Definition 4: Q-value weight of an edge: w_q. A Q-value weight of an edge is the learnt weight of an edge from the machine learning module which represents the "quality" of an edge within the public transport network graph. The term "quality" here means the preference quality from the commuters. In a simple manner, an edge with a higher "quality" means that the edge is more frequently travelled by the commuters.

Overview over the Trip Recommendation Module

The trip recommendation module comprises of two main components: the machine learning module and the routing service component. Figure 2 illustrates an overview of the trip recommendation module. To learn travel preference of the commuters from the historical trip data, the machine learning module calculates a Q-value weight matrix. The Q-value weight can be considered as a representation of the prior usage of the traffic system, which does not cover the basic information that is used as edge weights of classical routing engines, i.e. travel time, etc. In our approach we utilize a classical routing engine that can provide a routing service based on the basic edge attributes of a public transport network. For the routing service of our trip recommendation system, we use the open-source routing engine OpenTripPlanner (OTP). OTP is a family of open source software projects that providing passenger information and transportation network analysis services. The routing engine is a server-side Java component providing travel itineraries search combining transit, pedestrian, bicycle, and car segments through networks build from open standard OpenStreetMap and GTFS data. With the GTFS data, OTP builds an internal public transport network graph with default weight w_d as the edge weights. We integrate the Q-value weight w_q with the default weight w_d to generate a routing weight, which is further used by OTP to produce commuter data based itinerary recommendation.

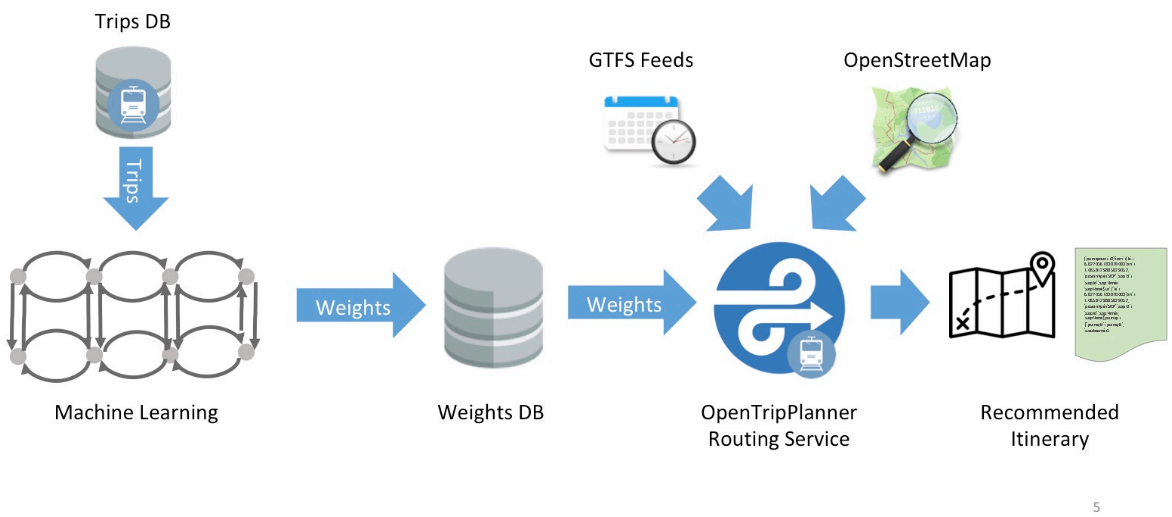

Figure 2 – Fundamental concept of the intelligent trip recommendation

Machine Learning Approach

In our investigation of the machine learning approach, we realize that the goal of the machine learning model is to learn a good mapping from prior commuter trip data to optimize the routing policy. For this reinforcement learning [2] is a promising approach, as reinforcement learning means learning a policy – a mapping of observations into actions based on the feedback from the environment.

An overview of the reinforcement learning model is shown in Figure 3. As described in the Section *Preliminary Concepts*, we model the public transport network as a vertex-edge graph, the stop is denoted as the vertex and the path from one stop to the next stop is denoted as the edge. In the reinforcement learning model, we define:

- *agent:* An agent takes actions; In the 2-D public transport network graph, an agent moves from a vertex to another vertex.
- *action:* An action represents an edge in the public transport network graph.
- *state:* A state represents a vertex in the public transport network graph.
- *reward* r_t*:* The frequency of the edge that derived from the commuter trip data
- *environment:* The environment represents the public transport network graph

In our approach, we use a type of reinforcement learning called Q-learning [3]. In Q-learning, each state-action pair is assigned a Q-value, which represents the sum of reinforcements that are calculated by a Q-value function,

$$Q_{st,at} = Q_{st,at} + \alpha * (r_t + \gamma * \max Q(st+1, a) - Q_{st,at})$$

where α is the learning rate, r_t represents the reward of regarding action, γ is discount factor, $Q_{st,at}$ is the Q-value of action *at* at the state *st*. The Q-value is updated while the agent explores the environment. As the agent explores the environment, it repeatedly interacts with the environment. At a given state, when the agent takes a particular action, it receives a reward signifying how good/bad the action is, and updates the Q-value table for the state-action pair to memorize the reinforcement that it received. As the agent moves, the Q-values for state-action pairs are continuously refined. The goal of the agent is to learn a good mapping from the state-action pairs with respect to maximizing the expected discounted reward. We used a greedy strategy for the reinforcement learning model. The output from the reinforcement learning model are represented as a Q-value matrix of state-action pairs, which is also the Q-value weight matrix that is further integrated with the classical routing service.

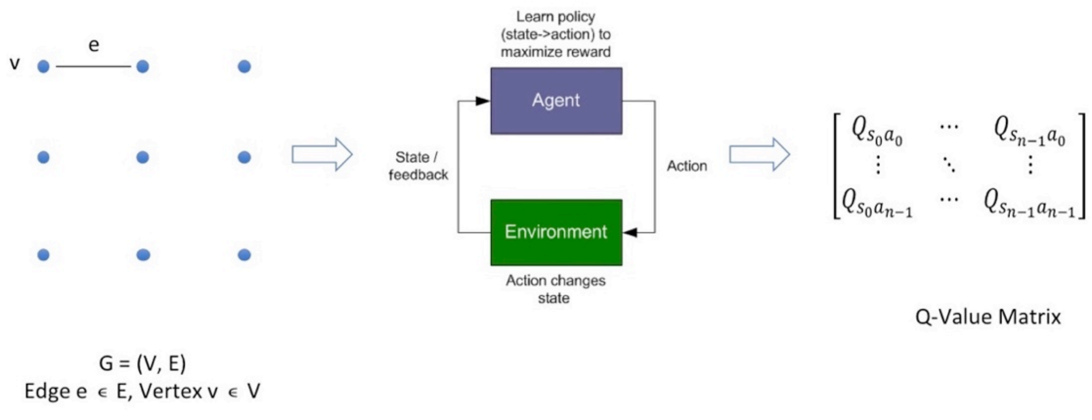

Figure 3 – Reinforcement learning Q-learning[3]

[3] https://www.ibm.com/developerworks/library/cc-reinforcement-learning-train-software-agent

System Architecture

This section describes the trip recommendation module with regard to the structure of the underlying the machine learning approach. The trip recommendation module imports and pre-processes the data from a trip data server by means of a data import service into trips DB. The trips DB is a central repository for pre-processed trip data. The machine learning module, applies the machine learning with given pre-processed trip data. The routing service handles the trip recommendation requests and provides trip proposals based on a customized OpenTripPlanner routing engine. The service component is implemented using the Spring framework and runs as a server-side Java application.

Figure 4 illustrates how the trip data are acquired from the trip data server via HTTP, and further persisted in the Trips DB. Before being fed into the machine learning module, the trip data are first pre-processed by calculating a matrix $R_{V,E}$ of the reward value r_t of each edge in the public transport network, which is defined as transit input data. With the transit input data, the machine learning module generates the Q-value weight matrix based on reinforcement learning as the output. The Routing service component is composed of an OpenTripPlanner routing engine integrated with the Q-value weight matrix and the web service unit that handles the routing request from Easy2Go/Boogi Server. Since we have different pilot cities with different GTFS data, the Routing Service runs as independent instance with respect to different pilot city.

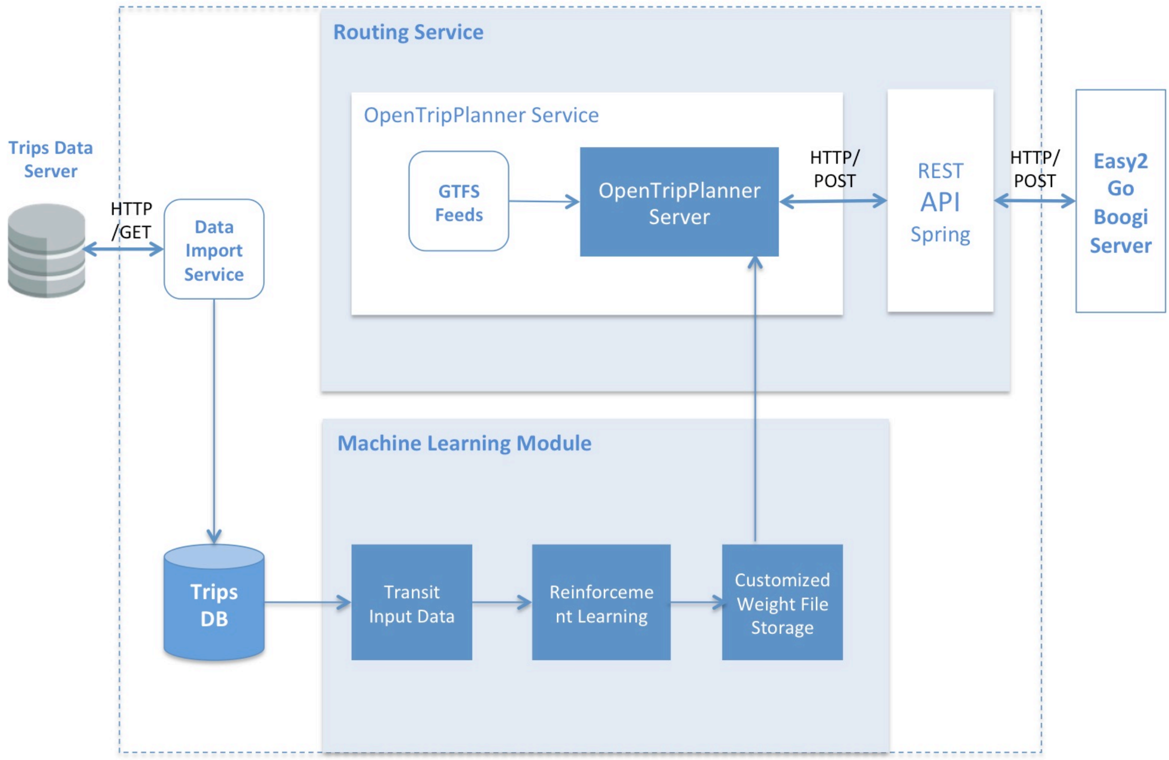

Figure 4 - Functional architecture of the trip recommendation system

Conclusion and Future Work

Our intelligent trip recommendation system is able to provide a commuter data-based routing service based on machine learning. A classical OTP routing service was adapted to use newly integrated weights modified by means of an reinforcement learning algorithm. We described the underlying technical concepts and the central aspects about the machine learning module. For further development, we are planning to collect more trip data to improve the reliability of the machine learning based weights and to optimise the reinforcement learning model in terms of the parameter estimation and optimisation. Future work also aims at utilizing historical trip data of a single commuters in order to facilitate personalized trip recommendations for individual users. This enables the trip recommendation system to suggest a more adequate route based on the personalized preference and crowd knowledge. The intelligent trip recommendation system will be further developed in the project ProSeMo (Proactive Seamless Mobility). In ProSeMo we plan to extend the approach to create data driven trip recommendations for intermodal mobility and to learn from the usage of the traffic system at different times of the day in order to be able to adapt the routing according the estimated utilization of the transport system at travel time.

Acknowledgements

This research was partially supported by the ProSeMo project of the research initiative "Entwicklung digitaler Technologien", which is funded by the Federal Ministry for Economic Affairs and Energy (BMWi) of the Federal Republic of Germany, and by EIT Digital, the leading European open-innovation and go-to-market organisation.

References

1. S. Péan, S. Schaffer, J. M. Alexandre, A. Filograna, F. Sauthier, L. Modena (2018). "*Towards seamless multimodal pay-as-you-go mobility*". 25th ITS World Congress, Copenhagen, Denmark
2. R. S. Sutton and A. G. Barto (1998). "*Reinforcement learning: An introduction*". MIT press Cambridge.
3. Watkins, C.J.C.H. (1989), "*Learning from Delayed Rewards*" (Ph.D. thesis), Cambridge University
4. P. Campigotto, C. Rudloff, M. Leodolter and D. Bauer (2017), "*Personalized and situation-aware multimodal route recommendations: the FAVOUR algorithm*", IEEE Transactions on Intelligent Transportation Systems, Volume 18 Issue 1, January 2017, Pages 92-102
5. A. Nuzzolo, A. Comi, U. Crisalli and L. Rosati (2013), "*An advanced pre-trip planner with personalized information on transit networks with ATIS*", Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, October 6-9