Bayesian classifier for Route prediction with Markov chains

Jonathan P. Epperlein[♭], Julien Monteil[♭], Mingming Liu[♯], Yingqi Gu[♯], Sergiy Zhuk[♭], Robert Shorten[♭♯]

***Abstract*— We present here a general framework and a specific algorithm for predicting the destination, route, or more generally a pattern, of an ongoing journey, building on the recent work of [1]. In the presented framework, known journey patterns are modelled as stochastic processes, emitting the road segments visited during the journey, and the ongoing journey is predicted by updating the posterior probability of each journey pattern given the road segments visited so far. In this contribution, we use Markov chains as models for the journey patterns, and consider the prediction as final, once one of the posterior probabilities crosses a predefined threshold. Despite the simplicity of both, examples run on a synthetic dataset demonstrate high accuracy of the made predictions.**

I. INTRODUCTION

Understanding driver intent is a prerequisite for a personalised driving experience with features such as personalised risk assessment and mitigation, alerts, advice, automated rerouting, etc. One of the most important driver intentions is the destination of the journey and the route to get there. Furthermore, in particular for hybrid vehicles, knowledge of the route ahead of time can be used to optimise the charge/discharge schedule, e.g. [2] find improvements in fuel economy of up to 7.8%.

In the data used in [3], 60% of trips are repeated and hence predictable from the driving history; [4] suggests that in the more general setting of user mobility, more than 90% of a user's trajectories are "potentially predictable." The combination of value and feasibility has sparked much research in this direction. While some algorithms rely on GPS trajectories only – e.g. [3] computes geometric similarity between curves obtained from cleaned and filtered raw GPS data – other approaches, such as [1], [5], [6], where Markov chains and hidden Markov models are built on the road network to estimate the most likely routes and destinations, next turns, or trip clusters in a broader sense, require map-matching to first map GPS traces to links in the road network.

The present paper falls into that latter category, as it builds on and extends the recent work of [1] and contributes to a novel and flexible approach to the important problem of driver intent prediction. It is structured as follows: after introducing some notation relating to probability, stochastic processes, and Markov chains in the next section, we then show in Section III how trips can be modelled as outputs of stochastic processes to obtain an estimate of the posterior probabilities of each known journey pattern. The algorithm resulting if Markov chains are used as the stochastic process models is described in detail in Section IV, and Section V provides some experimental validation. We close with some possible extensions and improvements, and the observation that [1] also fits into the presented framework, if the stochastic process model is chosen to be a naive Bayes model instead of Markov chains.

This work has been conducted within the ENABLE-S3 project that has received funding from the ECSEL Joint Undertaking under grant agreement no 692455. This joint undertaking receives support from the European Union Horizon 2020 research and innovation programme and Austria, Denmark, Germany, Finland, Czech Republic, Italy, Spain, Portugal, Poland, Ireland, Belgium, France, Netherlands, United Kingdom, Slovakia, Norway.

[♭]IBM Research – Ireland
{jpepperlein,julien.monteil,sergiy.zhuk}@ie.ibm.com
[♯]University College Dublin, Ireland yingqi.gu@ucdconnect.ie, robert.shorten@ucd.ie, mingming.liu@ibm.com

II. NOTATION

We *should* write $P(W = w)$ for the probability of the event that a realisation of the discrete random variable W equals w, and $P(W = w \mid V = v)$ the probability of that same event $W = w$ given that the event $V = v$ occurred. However, for convenience we will most of the time write $P(w \mid v)$ instead of $P(W = w \mid V = v)$, when it is clear from the context, what is meant. For a set of parameters μ parametrising a probability distribution, the notation $P(W = w \mid \mu)$ is taken to denote the probability of the event $W = w$ if the parameters are set to μ.

We let $\mathbb{N} = \{0, 1, 2, \dots\}$ denote the natural numbers, and for $N \in \mathbb{N} \setminus \{0\}$, $[N] := \{1, \dots, N\}$. Matrices will be denoted by capital letters, their elements by the same letter in lower case, and the set of $n \times n$ row-stochastic matrices, i.e. matrices with non-negative entries such that every row sums up to 1, by $\mathcal{M}^n$. We denote the cardinality of a set S, i.e. the number of its elements, by $\operatorname{card} S$. All cardinalities here will be finite.

A *stochastic process* is a sequence of random variables $\{X_t\}$ indexed by t, which often denotes time. For $t \in \mathbb{N}$, we call a sequence $(x_0, x_1, \dots, x_T)$ of realisations of the random variables X_t a *trajectory* of the process; $P(x_0, x_1, \dots, x_T \mid \mu)$ is the *probability* of the trajectory given that the stochastic process is parametrised by μ,[1] whereas interpreted as a function of μ, it is the *likelihood* of the parameters being equal to μ.

Specifically, we will use *Markov chains*, which are stochastic processes with $t \in \mathbb{N}$ and $X_t \in [N]$ that are completely defined by a *transition (probability) matrix* $A \in \mathcal{M}^N$ and a vector $\pi \in \mathbb{R}^N$ of initial probabilities. Then, $P(X_0 = i) = \pi_i$ and $P(X_t = j \mid X_{t-1} = i) = a_{ij}$ characterises the process. Note that the "next state" X_t depends *only* on the current realisation x_{t-1} and not on

[1]e.g. if X_t corresponds to an unfair coin flip, μ would correspond to the probability p of heads

the past; this is also know as the Markov property. This corresponds to a directed graph with N nodes and weights a_{ij} on the edge from node i to j. The stochastic process defined by the Markov chain then corresponds to an agent being initialised on some node i according to π and making every decision where to go next according to the weights on the edges leading away from it.

III. PROBLEM SETTING AND FRAMEWORK

From a driver's history H of trips taken in the past, we want to learn a predictive model, allowing us to identify properties (such as destination or specific route) of a currently ongoing trip as soon into the trip as possible. The history is

$$H = \Big\{ T_1 : \{(t_1^1, t_2^1, \dots, t_{L_1}^1), (r_1^1, r_2^1, \dots, r_{L_1}^1)\}, \dots \\ T_{N_H} : \{(t_1^{N_H}, t_2^{N_H}, \dots, t_{L_{N_H}}^{N_H}), (r_1^{N_H}, r_2^{N_H}, \dots, r_{L_{N_H}}^{N_H})\} \Big\},$$

where each *trip* T_i has length L_i and consists of sequences $(t_1^i, \dots)$ and $(r_1^i, \dots)$ of *time stamps* and *road segments*. The road segments are identified by their OpenStreetMap (OSM) way IDs [7], which implies that map matching has been performed on the raw GPS trajectories. We'll return to that point in Sec. V-A. Let $\mathcal{L} = \bigcup_{i=1}^{N_H} \bigcup_{t=1}^{L_i} \{r_t^i\}$ and $N = \operatorname{card} \mathcal{L}$ be the set and number of all road segments ever visited.

Each trip in H belongs to a *cluster* C_k, where the cluster encodes the journey pattern, destination, or more generally "properties" of the trip. From now on, we shall use the more generic term "cluster," as defined in [1]; see also there for further explanations. A cluster could be as coarse as a collection of all trips T_i with the same destination (defined as e.g. the last road segment of a trip), hence encoding only the property of destination, or more fine-grained, by defining a measure of similarity between trips and clustering according to those similarities; for instance trips along the "scenic route to work" and using the "fastest route to work" would then belong to different clusters despite sharing the same destination. More details are again postponed until the computational examples are described in Sec. V. Whichever way it is obtained, let us define this set of clusters by

$$C = \{C_1, \dots, C_{N_C}\}, \tag{1}$$

and assume that C_k is a partition of H, i.e. every trip belongs to *exactly one* cluster. We can then state the problem more precisely as:

> Given an ongoing trip $T : \{(t_1, \dots, t_L), (r_1, \dots, r_L)\}$, decide what cluster C_k trip T belongs to.

The proposed framework consists of a model for each cluster, providing a likelihood function $P(T \mid C_k)$, a prior probability $P(C_k)$, the Bayesian update to the posterior probabilities $P(C_k \mid T)$, and the criterion by which the prediction is made. The next sections elaborate on these parts.

A. Journeys as Stochastic Processes

Inspired by classical single-word speech recognition algorithms – see e.g. the classic survey [8] – where words are modelled as stochastic processes (for speech recognition frequently hidden Markov models), emitting sequences of vectors of spectral (and/or temporal) features derived from the acoustic speech signal, and the word corresponding to the stochastic process with the highest likelihood of having generated the present sequence is returned as the result, we propose to model journey patterns as stochastic processes emitting road links.

The choice of the type of stochastic process is free, there can even be different kinds for different clusters; once a type is chosen, the parameters for each cluster C_k have to be estimated from the trips in H belonging to C_k. In other words, a model for each cluster is "trained" on the available history. The outcome of this training process is a mapping $(T, C_k) \mapsto P(T \mid C_k)$, i.e. a function that allows us to evaluate the likelihood for each cluster of it having produced the currently available sequence of road segments (and time stamps). Our choice of stochastic process here will be Markov chains, which are particularly easy to train and evaluate; the details are given in Sec. IV-A.

B. Prior Probabilities

Before a trip even started, there might already be a high probability of it belonging to a certain cluster: if you have Aikido class on Wednesdays at 19:00 o'clock, and the current trip started on a Wednesday at 18:30 o'clock, there's a very high probability that the trip's destination will be your dojo. More generally, all additional information available about the trip, such as the current day of the week, the weather, current public events etc. form the *context* of the trip, and from the context, we can estimate the *prior probabilities* $P(C_k)$ of each C_k without any trip information yet available. In the absence of context, we can set $P(C_k) = \operatorname{card}\{i \mid T_i \in C_k\}/N_C$, i.e. make the prior proportional to how many trips in H belonged to C_k, or even simply set $P(C_k) = 1/N_C$ for all $j \in [N_C]$.

C. Bayesian Updates

Bayes' law relates the quantity we are ultimately interested in – the probability of the current trip belonging to C_k given what we already know about the trip, i.e. $P(C_k \mid T)$ – to the quantities estimated from H – the likelihood $P(T \mid C_k)$ and the prior $P(C_k)$ – by

$$P(C_k \mid T) = \frac{P(T \mid C_k) P(C_k)}{P(T)}. \tag{2}$$

A subtle point concerns the normalisation $P(T)$, the probability of the trip observed so far with no further assumptions on its nature: by computing the numerators $P(T \mid C_k)P(C_k)$ and then normalising them to sum up to one, i.e. by imposing

$$\sum\nolimits_{C_k \in C} P(C_k \mid T) = 1,$$

we implicitly assume that every trip is indeed in one of the already known clusters. As a simple fix, we could introduce the probability of any trip not belonging to any known cluster as a constant $P(\neg C)$, in which case we'd have $\sum_{C_k \in C} P(C_k \mid T) = 1 - P(\neg C)$, simply a normalisation

to a different constant. We proceed without accounting for unknown clusters, but keep this implicit assumption in mind.

D. Stopping Criterion

If, when, and how the updates should be stopped and the final prediction announced depends on the application. If the goal is route planning, we need to be reasonably certain of the unique destination before planning a route. On the other hand, if the goal is identifying risks along the remaining journey, it is sufficient to narrow down the set of possible routes and check for risks along all of them; predictions can be made continuously in this scenario. Other applications might call for other measures.

Here, we apply the simple criterion that as soon as one of the clusters' posterior probabilities exceeds a threshold $P(C \mid T) > 1-\alpha$, prediction stops and returns cluster C as the result, which should work well for the first application.

IV. CLUSTER PREDICTION ALGORITHM

In order to derive a concrete algorithm for cluster prediction, a choice of statistical process model has to be made, and the models have to be trained. In this section, we describe this for the case of Markov chains.

A. Modelling Clusters as Markov Chains

Modelling clusters by Markov chains is, of course, a simplification of reality, but as we shall see it leads to a computationally very tractable algorithm and performs very well in our computational experiments in the next section. The simplifying assumption is the "Markov assumption": if the current trip belongs to a cluster C_k, the probability distribution of the next road segment[2] R_{t+1} depends *only* on the current road segment r_t.

More formally, the state space of the Markov chain is $[N]$, where N is the total number of road segments in the road network under consideration. Every trip T, or rather its sequence $(r_1, \dots, r_L)$ of road segments, then corresponds to a trajectory of the Markov chain. If a trip belongs to a certain cluster C_k, e.g. "scenic route from home to work," and if the trip so far has been $r_0, \dots, r_t$, then, in full generality, the probability distribution of the next road segment, given all that is known at t, is $P(R_{t+1} = r_{t+1} \mid r_0, \dots, r_t, C_k)$. In modelling this as a Markov chain, we are imposing that

$$P(R_{t+1} = j \mid r_0, \dots, r_{t-1}, R_t = i, C_k) = P(R_{t+1} = j \mid R_t = i, C_k) = a_{ij}^k. \quad (3)$$

Then, $A^k \in \mathcal{M}^N$ is the transition probability matrix of cluster C_k, and a_{ij}^k is the probability to turn into road j from road i, if the current trip is in cluster C_k.

The transition probabilities are estimated by

$$a_{ij}^k = \frac{\sum_{T_m \in C_k} \operatorname{card}\{t \mid r_t^m = i, r_{t+1}^m = j\}}{\sum_{T_m \in C_k} \operatorname{card}\{t \mid r_t^m = i, t = 1, \dots, L_m - 1\}} = \frac{\text{number of transitions from } i \text{ to } j \text{ in } C_k}{\text{number of times } i \text{ is transitioned from in } C_k}. \quad (4)$$

[2]The capital "R" is used because it denotes a random variable.

This very intuitive estimate is in fact the maximum-likelihood (ML) estimate, see e.g. [9]. If a road segment i never appears (or more precisely, is never transitioned from) in any trip in C_k, then (4) cannot be applied; for now, we can just set these a_{ij}^k to 0, and return to this issue in Sec. IV-B.

For the initial probabilities $\pi_i^k = P(R_0 = i \mid C_k)$, the (ML) estimate is

$$\pi_i^k = \frac{\operatorname{card}\{m \mid r_0^m = i \text{ and } T_m \in C_k\}}{\operatorname{card}\{m \mid T_m \in C_k\}}, \quad (5)$$

but this assumes that the prediction task always starts at the beginning of a trip; if for whatever reason the first few links of a trip are missing in the data, this might lead to a failure of the prediction. E.g. by choosing $\pi_i^k = 1/\operatorname{card}\{j \mid r_t^m = j \text{ for any } t \text{ and any } T_m \in C_k\}$ if $r_t^m = i$ for any t and any $T_m \in C_k$, and 0 else, i.e. making the initial probability uniform over all road segments that appear in the cluster, or even setting $\pi_i^k = 1/N$ for all $i \in \mathcal{L}$, we avoid this problem and allow for prediction to start during a trip; we thus treat π^k somewhat heuristically as a design parameter. The likelihood function is then given by

$$P(r_1, r_2, \dots, r_L \mid C_k) = \pi_{r_1}^k \prod_{t=2}^{L} a_{r_{t-1}, r_t}^k, \quad (6)$$

or recursively by

$$P(r_1, r_2, \dots, r_L \mid C_k) = P(r_1, r_2, \dots, r_{L-1} \mid C_k) a_{r_{L-1}, r_L}^k. \quad (7)$$

B. Unseen Transitions and Unseen Road Segments

When a transition that has never occurred in the training data occurs in the current trip to be predicted, the algorithm described so far will break down, because the likelihood $P(r_1, r_2, \dots, r_L \mid C_k)$ will drop to 0 for all clusters C_k, and hence the posteriors $P(C_k \mid T)$ will be undefined as $0/0$. This will be a rather common situation: the data is not perfect, hence segments visited in reality could be missing in the data, GPS data could be mapped to the wrong way ID, small detours could take the driver along never before visited roads, etc. Similar to the PageRank algorithm [10] and as in [1], we address this problem by introducing a small $\varepsilon > 0$ and adding it to *each* transition probability (except self-loops, i.e. transitions $i \to i$), even the ones never observed. The matrices A^k then have to be re-normalized to obtain stochastic matrices again, however now every probability $a_{ij}^k \geq \varepsilon/(1 + (N-1)\varepsilon) > 0$. Formally, for all $i \neq j$:

$$\widetilde{a}_{ij}^k = a_{ij}^k + \varepsilon \qquad a_{ij}^k = \widetilde{a}_{ij}^k / \sum_m \widetilde{a}_{im}^k. \quad (8)$$

If the prediction algorithm receives a road segment that has never been seen, i.e. if $r_L \notin \mathcal{L}$, this is treated in much the same way by extending the likelihood function, or rather the transition probability matrices A^k, to assign a minimum probability $\varepsilon/(1+(N+1)\varepsilon)$ to transitions to or from unseen links. This can be addressed very easily in the code by inserting an **if** statement before updating the likelihood, but

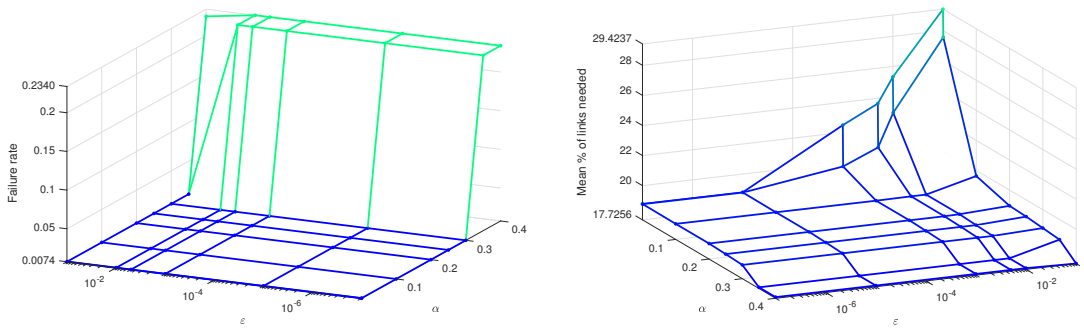

Fig. 1. Failure rates and mean percentage of links needed to make a prediction after 8 rounds of 50-50 cross validation. Note the different view angles of the two axes.

Algorithm 1 Cluster prediction using Markov chains

procedure TRAINING(H, C, $\varepsilon > 0$)
  for all $C_k \in C$ **do**
    $(A^k, \pi^k) \leftarrow$ by (4), (5), (8)
procedure PREDICTION($A^k, \pi^k, P(C_k)\,\forall C_k \in C, \alpha, \bar{\varepsilon}$)
Initialization
  $r_{\text{to}} \leftarrow r_0$, $r_{\text{from}} \leftarrow$ None
  for all $k = 1, \ldots, N_C$ **do**
    $\ell_k \leftarrow \pi^k_{r_0}$, $P_k \leftarrow \ell_k P(C_k) / \sum_{j=1}^{N_C} P_j P(C_j)$
Prediction loop
  while $P_k \leq 1 - \alpha\ \forall k$ **do**
    if trip is finished **then**
      return None
    else
      wait until new segment r_{new} is received
      $r_{\text{from}} \leftarrow r_{\text{to}}$, $r_{\text{to}} \leftarrow r_{\text{new}}$
      for all $k = 1, \ldots, N_C$ **do**
        if $r_{\text{from}} \notin \mathcal{L} \vee r_{\text{to}} \notin \mathcal{L}$ **then**
          $\ell_k \leftarrow \ell_k \cdot \bar{\varepsilon}$
        else
          $\ell_k \leftarrow \ell_k \cdot a^k_{r_{\text{from}}\, r_{\text{to}}}$ ▷ see Eq. (7)
        $\bar{P}_k \leftarrow \ell_k \cdot P(C_k)$ ▷ Bayesian update
      for all $k = 1, \ldots, N_C$ **do**
        $P_k \leftarrow \bar{P}_k / \sum_{j=1}^{N_C} \bar{P}_j$ ▷ Normalization
  return $\arg\max_k P_k$

also formally: we add a state u for "unseen segment" to $\mathcal{L}$ to obtain $\mathcal{L}^u = \mathcal{L} \cup \{u\}$, and add a column and row equal to $\bar{\varepsilon} := \varepsilon/(1+(N+1)\varepsilon)$ to every A^k (note that a transition from u to u is now allowed, as it corresponds to two previously unseen segments in sequence, and not necessarily a repetition of the current segment). Every link with an ID not in $\mathcal{L}$ is then mapped to u before the likelihood is computed.

Pseudocode of the full algorithm, including the modifications described here, is shown as Algorithm 1.

V. COMPUTATIONAL EXPERIMENTS

We now use the dataset in [1] to test the proposed algorithm; a short description is given below, for more details, see [1].

A. Data

Seven origins and destinations across Dublin were selected, representing typical points of interest such as "home," "work," "childcare," etc. This yields a total of 21 possible origin/destination pairs, for 17 of which up to 3 distinct routes were generated. These routes were then fed into the microscopic traffic simulator SUMO [11] to generate a total of $N_H = 781$ trips in the form of timestamps and longitude/latitude coordinates. To simulate real GPS data, uniformly distributed noise (on a disk of radius 10m) was added to each point.

To prepare the data in the form of H for our algorithm, the sequence of GPS points needs to be converted into a sequence of way IDs, which was done using the Map Matching operator of IBM Streams Geospatial Toolkit. Subsequently, duplicates were removed (i.e. if more than one consecutive GPS point was mapped to the same road segment, only the first instance was kept).

B. Clustering

As in [1], we consider two types of clusters:

- *by Origin/Destination:* Two trips belong to the same cluster, iff they have the same origin and destination (as defined by proximity on their final GPS coordinates). This results in $N_C = 17$ clusters.
- *by Route:* Similarity between two trips is measured by the ratio of shared road segments between the two trips and the total number of road segments in both trips. Hierarchical clustering is then performed, and the dissimilarity threshold is chosen to be 0.3, yielding $N_C = 30$ clusters.

C. Prediction Results

In all cases described below, we chose uniform probabilities for the prior $P(C_k)$ and initial probabilities π^k, i.e. $P(C_k) = 1/N_C$ and $\pi^k_i = 1/N\ \forall k, i$. More careful choices could certainly improve prediction results, but as we shall see below, this simplest of choices is sufficient to demonstrate the efficacy of the algorithm.

We collected two quantities of interest: the *failure rate* as the number of false predictions (which includes cases where the end of a trip is reached without a prediction being made) divided by the number of all trips predicted,

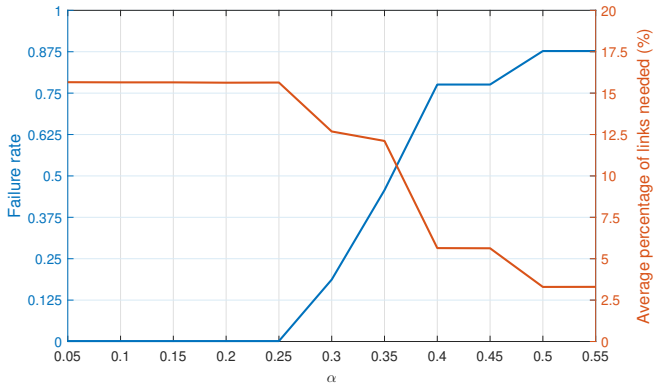

Fig. 2. Failure rate and fraction of links needed for prediction vs confidence level α for fixed $\varepsilon = 10^{-6}$ if clustering by Origin/Destination.

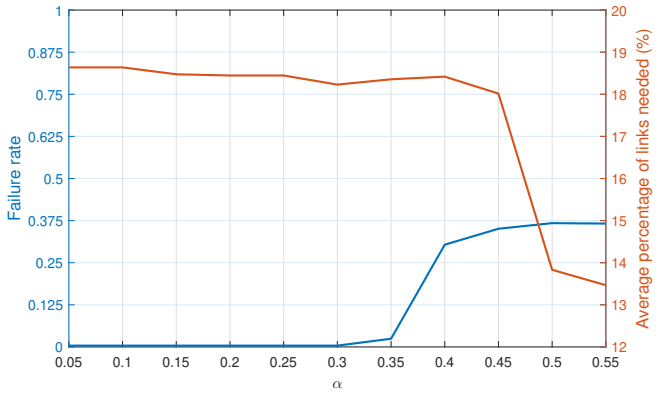

Fig. 3. Failure rate and fraction of links needed for prediction vs confidence level α for fixed $\varepsilon = 10^{-6}$ if clustering by Route.

and the *fraction of trip completed* at the time of prediction, i.e. #(road segments visited)/#(total road segments in trip). If a wrong prediction was made, we recorded the amount of links needed as *NaN*; these points are then excluded from the computation of averages.

In order to get an idea of good ranges for the parameters ε and α, we performed a small initial **cross-validation** experiment by training on only 50% of the data and then predicting the route of the remaining 50%. This was done for 8 random choices of training and testing set, on a grid of values $(\alpha, \varepsilon) \in \{10^{-4}, 0.001, 0.1, 0.2, 0.25, 0.3, 0.35, 0.4\} \times \{10^{-7}, 10^{-5}, 0.001, 0.005, 0.01, 0.1\}$. The results are shown in Fig. 1 and indicate that the approach is robustly effective over a wide range of values for α and ε: with parameters in a reasonable range, e.g. $\alpha \in [0.01, 0.1]$ and $\varepsilon \in [10^{-7}, 10^{-5}]$, the routes are predicted accurately in more than 99% of the test cases within the first 20% of the trip. The results also show "breakdown points:" if the confidence level α is increased to the point where a posterior probability of 0.6 is sufficient for prediction already, the amount of false predictions increases rapidly; on the other hand, if the small probability parameter ε is chosen so large that it dominates the probabilities estimated from data, the amount of links necessary to distinguish routes increases rapidly.

To investigate further, we then performed **leave-one-out**

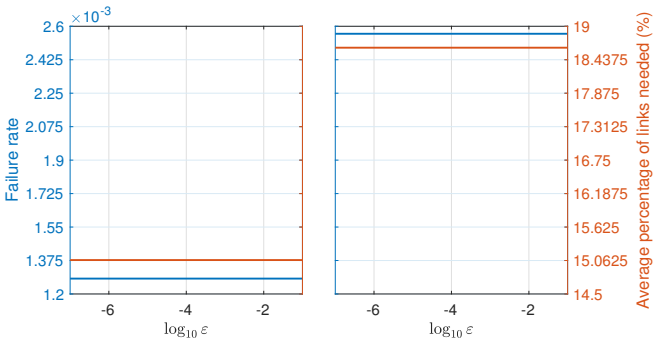

Fig. 4. Failure rate and fraction of links needed for prediction vs ε for fixed $\alpha = 0.1$ if clustering by Origin/Destination (left) and Route (right).

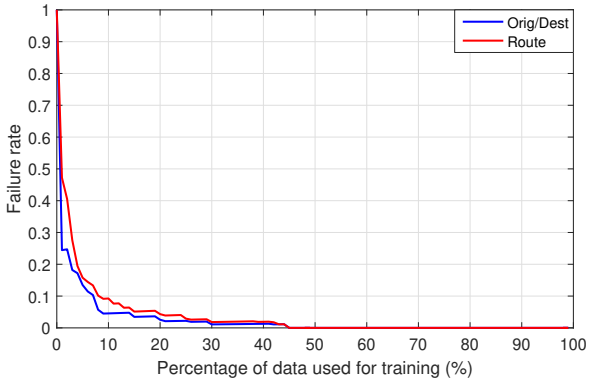

Fig. 5. Decreasing failure rates as more trips are added to the training data for clustering by Origin/Destination and Route; $\alpha = 0.1$ and $\varepsilon = 10^{-6}$.

cross-validation along two "slices" on a finer grid: for each of the $N_H = 781$ trips, we trained the Markov chains on the remaining $N_H - 1 = 780$ trips; then, the trip that was left out was predicted. This was done for $\varepsilon = 10^{-6}$ and a finer grid of values for α, and for $\alpha = 0.1$ and a finer grid of values for ε. The entire procedure was repeated once for clustering by Origin/Destination, and once for clustering by Route. The results are shown in Figs. 2-4, and we observe:

- The prediction accuracy is very robust with respect to α, the failure rate is below 1% for a wide range of α, and declines rapidly once a breakdown point of $\alpha \approx 0.25$ is reached.
- The same goes for the percentage of links needed to make a correct prediction. Additionally we note that, as should be expected, lower confidence in prediction (i.e. larger values of α) tends to lead to fewer links needed for prediction; however this effect only kicks in once the failure rate increases quickly.
- Predicting the origin and destination appears to be slightly easier, since it consistently requires fewer links to do so. This is not surprising as there are only 17 clusters to choose from, whereas in the case of predicting the route there are 30.
- The robustness with respect to ε is even more pronounced: Fig. 4 indicates that, once a good value for α is selected, neither the failure rate nor the amount of links needed for prediction depend on ε.

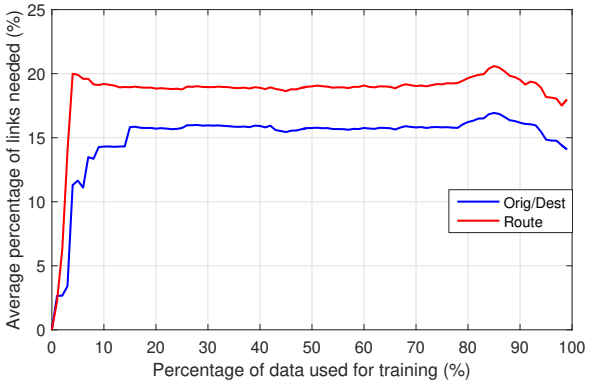

Fig. 6. Fraction of links needed as more trips are added to the training data for clustering by Origin/Destination and Route; $\alpha = 0.1$ and $\varepsilon = 10^{-6}$.

As a last experiment, we attempted to simulate the realistic situation of an in-car system improving its prediction model with each taken trip by **incrementally moving trips from the testing set to the training set**. Specifically, for the first data point, we trained on the first trip only and predicted the remaining $N_H - 1 = 780$ ones. This obviously lead to an immediate wrong prediction of almost all trips. We then added the second trip, retrained and predicted the remaining 779 trips, and so on. The results are shown in Figs. 5 and 6, and we see that once roughly 10% of the trips have been taken (so around 75 trips), the algorithm predicts at least 90% of the remaining trips correctly while needing on average between 15 and 20% of the trip to be completed to make its prediction.

VI. CONCLUSIONS

The contribution was twofold: on the one hand, the flexible framework of modelling route patterns as stochastic processes and using the associated likelihoods to update a posterior probability is introduced, and on the other, a concrete algorithm is presented, obtained by modelling the stochastic processes as Markov chains.

The flexibility of the approach is only touched upon, there are many possible extensions which should be explored once a richer dataset is available – even though we worked on the generation of a realistic dataset (see Sec. V-A), it is still a synthetic one and the excellent performance of the presented algorithm is hard to improve upon. Improvements to be explored using a more challenging, real dataset, include:

- Other stochastic process models can be used. Indeed, the approach taken in [1] fits into the outlined framework, if the choice of stochastic process is a naive Bayes model, i.e. if assuming that $P(r_1, \ldots, r_L \mid C_k) = \prod_{t=1}^{L} P(r_t \mid C_k)$. We intend to test other stochastic process models, such as the recently developed closed-loop Markov modulated Markov chains [12] in the near future.
- So far, the available context is not used at all. For future practical applications however, the prior probabilities should be made dependent on such contextual variables as the day of the week or the time of the day, for instance by setting $P(C_k, \text{weekday}) \propto \frac{\#(\text{trips in } C_k \text{ that occurred on a weekday})}{\#(\text{trips that occurred on a weekday})}$, the prior probability if the current trip occurs on a weekday would be made proportional to the relative frequency of trips in C_k among previous weekday trips; [1] has further details on context and its inclusion, only there, the contextual variables influence the stochastic process model directly instead of entering via the prior.
- The initial probabilities π^k and small probabilities ε can be shaped to be larger for roads that are not on, but close to, the roads in cluster C_k, and smaller for roads that are far away. This can be expected to improve convergence of the posterior probabilities.

Overall, the success the approach has without tapping into such extensions is encouraging further research.

REFERENCES

[1] Y. Lassoued, J. Monteil, Y. Gu, G. Russo, R. Shorten, and M. Mevissen, "Hidden Markov model for route and destination prediction," in *IEEE International Conference on Intelligent Transportation Systems*, 2017.

[2] Y. Deguchi, K. Kuroda, M. Shouji, and T. Kawabe, "HEV charge/discharge control system based on navigation information," in *Convergence International Congress & Exposition On Transportation Electronics*. Convergence Transportation Electronics Association, oct 2004.

[3] J. Froehlich and J. Krumm, "Route prediction from trip observations," in *SAE Technical Paper*. SAE International, 04 2008. [Online]. Available: http://dx.doi.org/10.4271/2008-01-0201

[4] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010. [Online]. Available: http://science.sciencemag.org/content/327/5968/1018

[5] Simmons, B. Browning, Y. Zhang, and V. Sadekar, "Learning to predict driver route and destination intent," in *2006 IEEE Intelligent Transportation Systems Conference*, Sept 2006, pp. 127–132.

[6] J. Krumm, "A Markov model for driver turn prediction," SAE Technical Paper, Tech. Rep., 2008.

[7] OpenStreetMap Wiki, "Way — OpenStreetMap Wiki," 2017, accessed Nov 24. [Online]. Available: http://wiki.openstreetmap.org/w/index.php?title=Way&oldid=1520779

[8] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb 1989.

[9] T. W. Anderson and L. A. Goodman, "Statistical inference about Markov chains," *The Annals of Mathematical Statistics*, vol. 28, no. 1, pp. 89–110, 03 1957. [Online]. Available: http://dx.doi.org/10.1214/aoms/1177707039

[10] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep. 1999-66, November 1999, previous number = SIDL-WP-1999-0120. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[11] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012. [Online]. Available: http://sumo-sim.org/pdf/sysmea_v5_n34_2012_4.pdf

[12] J. Epperlein, R. Shorten, and S. Zhuk, "Recovering Markov Models from Closed-Loop Data," *ArXiv e-prints*, June 2017.