Characterizing Driving Context from Driver Behavior

Sobhan Moosavi[†], Behrooz Omidvar-Tehrani[†], R. Bruce Craig[§], Arnab Nandi[†], Rajiv Ramnath[†]

[†]Department of Computer Science and Engineering, Ohio State University
[§]Nationwide Insurance
{moosavi.3,omidvar-tehrani.1,nandi.9,ramnath.6}@osu.edu
craigr2@nationwide.com

ABSTRACT

Spatio-temporal data is increasingly available due to the ubiquity of sensors of various types and the almost unlimited capacity of data storage resources. Consequently, a variety of data-analytic applications have been developed to gain useful insights from the data. Discovery of the characteristics of *driving context*, where a context is a combination of *location* and *time*, is a new and challenging problem in this area. An example of such a characteristic is the pattern of *correlation* between driving behavior and traffic condition. This contextual information enables us to validate hypotheses about driving behavior of an individual. In this paper, we present DriveContext, a novel framework to find the characteristics of a context. DriveContext consists of two major components: dSegment which extracts driving patterns within a trajectory (e.g., a speeding-up), and dDescribe which finds the set of potential causes to justify a pattern (e.g., traffic congestion). We build and evaluate DriveContext components using several spatio-temporal data sources, including a large-scale trajectory dataset, traffic data, and data on the features of roads. Our analysis and results show the feasibility of the framework in identifying meaningful driving patterns, with improvements in comparison with the state-of-the-art. We also demonstrate how the framework derives interesting characteristics for different contexts, through real-world examples.

CCS CONCEPTS

• **Information systems → Data mining**; *Spatial-temporal systems*;

KEYWORDS

Driver Behavior, Segmentation, Driving Pattern, Driving Context

ACM Reference format:
Sobhan Moosavi[†], Behrooz Omidvar-Tehrani[†], R. Bruce Craig[§], Arnab Nandi[†], Rajiv Ramnath[†]. 2017. Characterizing Driving Context from Driver Behavior. In *Proceedings of SIGSPATIAL'17, Los Angeles Area, CA, USA, November 7–10, 2017,* 10 pages.
https://doi.org/10.1145/3139958.3139992

1 INTRODUCTION

The amount and availability of spatio-temporal data has drastically increased thanks to the ubiquity of sensors in various applications and high-capacity data centers that can store and serve up this data. Transportation data is an example of spatio-temporal data, where the New York taxi cab [7], Porto cab [19], and GeoLife [29] are some instances of that. Given the availability of these large transportation data sources, various analysis applications have been developed to gain insights from this data. Discovery of characteristics of *driving context* is a new application area which we introduce in this paper. A *context* can be described as combination of *location* (e.g., Interstate-90[1]) and *time* (e.g., weekdays between 3pm to 7pm). A *characteristic* for a context can be identified as correlation between driving behavior and an environmental effect (e.g., traffic congestion). By having information about different driving contexts, one can validate hypotheses about behavior of an individual within a context, and also provide feedback to drivers in order to help them to improve their skills. The former one is related to *usage-based insurance* (see Example 1.1) and the latter one is known as *driver coaching* [25]. Note that neither of mentioned applications can be appropriately handled without having the characteristics of driving contexts as a prior.

In this paper, we address the problem of *discovering driving context* by exploring characteristics for a given *context*, based on behavior of drivers and using complementary sources of spatio-temporal data to analyze behavior. We define behavior of a driver in terms of meaningful *driving patterns*. Moreover, we try to explore *causes* which underlie a specific pattern within a context by conducting analysis across several spatio-temporal data sources (e.g., traffic data, data on the features of roads, etc.). This process will shape our *framework* to identify the characteristics for a context as we demonstrate that later in this paper. Figure 1 shows an example of a trajectory, where red dots show the location of the car for every second of the trip. The trajectory begins at the bottom center and continues to the left after a clock-wise turn. Different parts of the trip can be seen to exhibit different driving patterns, as marked out by the ovals. For instance, the oval *B* shows *slow-down*, oval *C* shows a *loop (ramp)*, oval *D* is a *speeding-up*, etc.

In this way, driving patterns are portions of a trajectory where there is homogeneity of driving behavior. In other words, a driving pattern is the consistent behavior of a driver within a sub-trajectory. The cause behind a transition between patterns, hence introducing a new pattern, can be *extrinsic* (e.g., an accident, a traffic signal, a traffic congestion, etc.) or *intrinsic* (e.g., driver-generated distraction, personality of the driver, etc.). For example, in Figure 1, a transition occurred in the middle of highway, where a *slow-down* is happened (the oval *E*). The cause behind this change may be a traffic congestion. The focus of this study is on extrinsic causes.

The problem of discovery of driving context, as we formulate that in this paper, is a complex problem, because we have to deal with following challenges. First, unlike studies [12, 13, 23] which collected data using a fully monitored environment (for example,

SIGSPATIAL'17, November 7–10, 2017, Los Angeles Area, CA, USA

ACM ISBN 978-1-4503-5490-5/17/11.
https://doi.org/10.1145/3139958.3139992

[1] Interstates form a network of controlled-access highways which are part of the National Highway System of the United States (see https://en.wikipedia.org/wiki/Interstate_Highway_System).

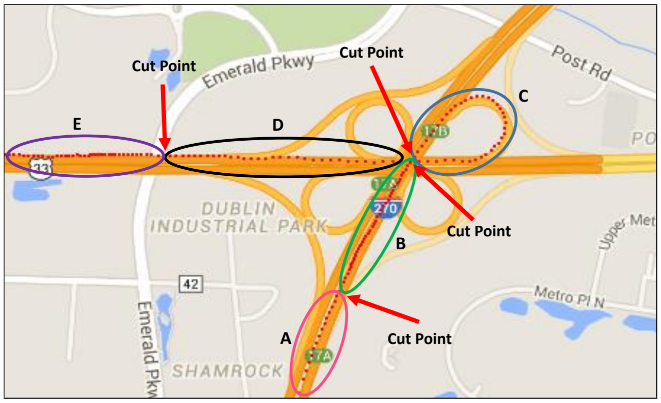


Figure 1: A sample trajectory with several driving patterns specified by ovals. Red arrows show the points of transition between patterns. Each pattern illustrate a homogeneous part of the trajectory and existence of each pattern is correlated to some causes (e.g., traffic congestion).

with cameras placed inside the car monitoring driver's every move and expression) and a small set of drivers and routes, we work on a large-scale dataset which is the result of collecting data by observing only externally visible phenomena (e.g., vehicle's speed) with no additional intrusive monitoring. In addition, because intrinsic reasons could also be causes of specific pattern transitions [8], and even if we could comprehensively monitor drivers, routes and vehicles, relating a pattern to *extrinsic* causes still remains challenging. Thus, finding a valid set of patterns and also the exact set of extrinsic causes that underlie each pattern, which are the bases of deriving characteristics for a context, are two challenging tasks, worthy of our study. Example 1.1 illustrates a potential use of our results.

Example 1.1. Regarding usage-based insurance (UBI), an insurance company provides a personalized insurance policy for a customer, based on his/her driving history[2]. For this purpose, the insurance company needs to compare the behavior of a driver to a reference population of drivers in order to find out how "risky" or "safe" the driver is. Assume Mark is a new customer and his driving history reveals that his driving behavior, within a context C, shows 20% more *hard-braking* and *hard-acceleration* patterns in compare to a reference population of drivers in the same context. Hence, Mark appears to show an abnormal behavior, and his driving may be characterized as risky.

In this paper, we introduce DriveContext, a framework to efficiently discover characteristics of driving contexts. This framework consists of two major components, dSegment and dDescribe. The first component, dSegment, applies a behavior-based trajectory segmentation algorithm to find meaningful driving patterns within a trajectory. Then, dDescribe, the second component of our system, reveals the extrinsic causes for each driving pattern. We apply DriveContext on a real-world dataset of car trajectories to illustrate how to derive interesting characteristics for different contexts. The main contributions of this paper may be summarized as follows:

- We propose a novel trajectory segmentation approach, dSegment, to find driving patterns based on behavior of drivers.
- We propose a novel usage of spatio-temporal data sources, in terms of dDescribe component, to explore causes which make a driving pattern to happen.
- We leverage the causality analysis results, conducted for a set of driving patterns within a context, to explore the characteristics of that context.

The remainder of this paper is organized as follows: Section 2 introduces the formal problem statement. Section 3 provides the DriveContext framework and its major components. Next, the experimental protocol and results are presented in Section 4. We provide a summary of related work in Section 5. Lastly, we conclude in Section 6 by summarizing and describing future work.

2 PRELIMINARIES AND PROBLEM STATEMENT

Assume we are given a transportation database $\mathcal{D}$ of the form $\langle \Upsilon, \Gamma \rangle$ where Υ and Γ are the set of vehicles and trajectories, respectively. Each trajectory $\gamma \in \Gamma$ is a sequence of $|\gamma|$ data points $\langle \rho_1, \rho_2, \ldots, \rho_{|\gamma|} \rangle$. Each data point ρ is a tuple of the form $\{t, lat, lng, s, a, h\}$ which captures a vehicle's status at time t as its latitude and longitude are $\langle lat, lng \rangle$, with speed s (km/h), acceleration a (m/s^2), and heading h (degrees). Time is considered to be measured in seconds. Also, the heading is the direction of the moving vehicle, described by a degree-value between 0 and 359, where 0 means the north.

We study the "discovery of driving context" in terms of two subproblems: *Segmentation* and *Causality Analysis*. A segmentation of a trajectory γ into n segments, denoted as seg_γ, is a set of cutting indexes $seg_\gamma = \langle I_1, I_2 \ldots, I_n \rangle$ that mark the end points of the segments within a trajectory. Thus, we can define a set of cutting data points for the segmented trajectory γ as $\langle p_{I_1}, p_{I_2} \ldots, p_{I_n} \rangle$. Note that $p_{I_n} = \rho_n$ (since segments are specified by their last data point, the last cutting point is the last data point of γ). All data points between indexes I_{i-1} and I_i, excluding point $\rho_{I_{i-1}}$ and including point ρ_{I_i}, belong to the i^{th} segment. Note that segments are non-overlapping. Each segment represents a *driving pattern* and each cutting point $p_{I_i}, I_i \in seg_\gamma$, represents a *transition between patterns.* Figure 1 shows several segments (by ovals) and cutting points (by arrows). Considering the segmentation task as an optimization problem, we define the optimization goals as (i). maximizing homogeneity within segments, and (ii). minimizing the number of extracted segments.

The existence of a segment is potentially relevant to extrinsic or intrinsic causes. In this work, the focus is on extrinsic causes that we refer to as *events*. We keep track of events in an event database $\mathcal{E}$ of the form $e = \langle t, lat, lng, type \rangle$, where each event $e \in \mathcal{E}$ occurs in time t, in a geographical area whose center is $\langle lat, lng \rangle$ of type $type$. An event can be of any of types including *Physical Fact* (e.g., traffic light in a road), *Physical-Temporal Event* (e.g., traffic congestion), or *Temporal Event* (e.g., tornado). Given the set of cutting points $\langle p_{I_1}, p_{I_2} \ldots, p_{I_n} \rangle$, identified as result of segmenting trajectory γ, and the database $\mathcal{E}$ of events, the second sub-problem (i.e., causality analysis) is one of finding if, and to what extent, each cutting point p_{I_i}, $1 \leq i \leq n$, is related to (or caused by) an event $e \in \mathcal{E}$.

3 THE DRIVECONTEXT FRAMEWORK

In this section, we present the novel framework DriveContext to discover characteristics of different driving contexts. Figure 2 depicts the overall process of DriveContext where it consists of two major components, which are dSegment and dDescribe. Details on these components are provided in the following sub-sections.

[2] https://en.wikipedia.org/wiki/Usage-based_insurance

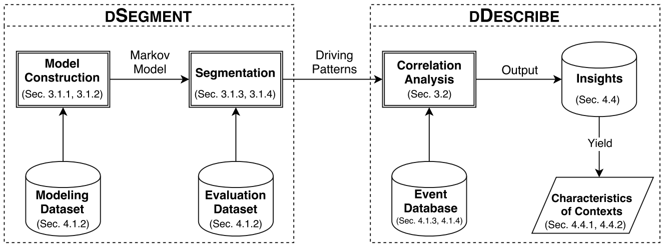


Figure 2: The overall process of DriveContext framework which consists of two components, the dSegment which includes the *model construction* and the *segmentation approach*, and the dDescribe which includes the *correlation analysis* based on extracted patterns that yields insights in terms of *characteristics of contexts.*

3.1 dSegment Component

dSegment is a novel approach to wisely partition a trajectory based on behavior of driver, such that each resulting segment corresponds to a meaningful driving pattern (e.g., turn, speed-up, etc.). Based on Figure 2, dSegment consists of two parts, *Model Construction* and *Segmentation.* The first part, "Model Construction", includes *Dataset Preprocessing* and *Markov Model Creation* . The second part, "Segmentation", comprises *Trajectory Transformation* and *Trajectory Segmentation.* We describe each of aforementioned sub-parts as follows.

3.1.1 *Dataset Preprocessing.* Regarding the description of the data model in Section 2, the dataset is a collection of trajectories, where each trajectory is a sequence of data points. The preprocessing of dataset consists of the following steps: 1) Removing data points with missing or noisy (out of range) GPS records, 2) Rounding the values of *Acceleration* to be divisible by 0.25, and 3) Using *Change of Heading* instead of absolute heading values. Steps 2 and 3 help to simplify the Markov Model by reducing the possible number of states, where there will be no significant effect on generalization of the model. Moreover, the step 3, which is an empirical decision, helps to reflect the change of heading more clearly. For example, the difference between 0 and 359 is 1, while we cannot directly derive such value from absolute heading values.

3.1.2 *Markov Model Creation.* The goal is to model behavior of drivers in terms of a finite state machine that provides probability of transition from one *driving state* to another one. In this way, we build a memory-less Markov model $M = \{\Phi, \Delta, \Pi\}$, where Φ is the set of states, Δ is the set of transition between states (along with the frequency of each transition), and Π is the set of probabilities of transition between states. We use the following principles to create M:

- State: We define a state $\phi \in \Phi$ as $\phi = \langle s, a, h \rangle$, where s, a, and h are speed, acceleration, and change of heading, respectively.
- Transition: Given a trajectory $\gamma = \langle \rho_1, \rho_2, \ldots, \rho_n \rangle$, for each pair of consecutive data points ρ_i and ρ_{i+1} of γ, $1 \leq i < n$, we create two states $\phi_i = \langle s_i, a_i, h_i \rangle$ and $\phi_{i+1} = \langle s_{i+1}, a_{i+1}, h_{i+1} \rangle$ for ρ_i and ρ_{i+1}, respectively. We denote a transition from state ϕ_i to ϕ_{i+1} as $\phi_i \longrightarrow \phi_{i+1}$. If Δ doesn't contain transition $\phi_i \longrightarrow \phi_{i+1}$, then we add $\langle \phi_i \longrightarrow \phi_{i+1}, 1 \rangle$ to Δ. Otherwise, we increase the frequency of transition $\phi_i \longrightarrow \phi_{i+1}$ by 1.
- Probability of Transition: For a state ϕ, let us assume there is a $\delta \subseteq \Delta$, where $\delta = \{\langle \phi \longrightarrow \phi_1, n_1 \rangle, \ldots, \langle \phi \longrightarrow \phi_k, n_k \rangle\}$ and n_i is the number of observed transitions from ϕ to ϕ_i in the training (modeling) dataset, then we update Π by inserting the probability of each transition $\phi \longrightarrow \phi_i$, $1 \leq i \leq k$, using Equation 1:

$$prob_{\phi \rightarrow \phi_i} = \frac{n_i}{\sum_{j=1}^{k} n_j} \tag{1}$$

By using above principles, we may end up with a sparse Markov model. For example, we may not observe a transition from $\phi_1 = \langle 25, -2, 170 \rangle$ to $\phi_2 = \langle 24, -2, 170 \rangle$, although such a transition is quite likely to happen. In order to deal with this shortcoming of a basic Markov model and also to avoid the overfitting problem that the model construction is purely based on the training (modeling) trajectories, we need a further processing step known as *Regularization.* To do this, we adapt an existing, intuitive regularization approach known as *Wedding Cake* technique [11]. Assume we have a state $\phi = \langle s, a, h \rangle$ which has transition to a set of states $\bar{\Phi} = \{\langle s_1, a_1, h_1 \rangle, \langle s_2, a_2, h_2 \rangle, \ldots, \langle s_n, a_n, h_n \rangle\}$. Also, consider values S_{th}, A_{th}, and H_{th} as thresholds on speed, acceleration, and heading, respectively, to define regularization intervals. We use Algorithm 1 to regularize the Markov Model.

Algorithm 1: Wedding Cake Regularization [11]

```
   Input: $\phi, \bar{\Phi}, S_{th}, A_{th}, H_{th}$
1  for $sp = (\phi.s - S_{th})$ to $(\phi.s + S_{th})$ do
2      for $ac = (\phi.a - A_{th})$ to $(\phi.a + A_{th})$ do
3          for $hd = (\phi.h - H_{th})$ to $(\phi.h + H_{th})$ do
4                  ▹ Expanding state $\phi$
5              for $\phi' \in \bar{\Phi}$ do
6                  $\phi'' \leftarrow \langle (\phi.s + sp), (\phi.a + ac), (\phi.h + hd) \rangle$
7                  $prob_{\phi'' \rightarrow \phi'} += \frac{prob_{\phi \rightarrow \phi'}}{Euclidean(\phi, \phi'')}$
8              end
9                  ▹ Expanding states in $\bar{\Phi}$
10             for $\phi' \in \bar{\Phi}$ do
11                 $\phi'' \leftarrow \langle (\phi'.s + sp), (\phi'.a + ac), (\phi'.h + hd) \rangle$
12                 $prob_{\phi \rightarrow \phi''} += \frac{prob_{\phi \rightarrow \phi'}}{Euclidean(\phi', \phi'')}$
13             end
14         end
15     end
16 end
   Output: Regularized Markov Model
```

In Algorithm 1, a *Euclidean* function calculates the Euclidean distance between two states. Prior to using this distance measure, we normalize the value of all features to lie between 0 and 1 (i.e., min-max normalization). The idea of regularization is intuitive, where we first try to expand the set of states of the basic Markov model (lines 6 and 11), and then update the probability values of the transitions for the updated states (lines 7 and 12). Expansion of states simply means creating new states, if they do not exist already, by updating the feature values of initial state (e.g., ϕ in line 6), using different thresholds. Moreover, the update of probabilities is about assigning the probability of transition to newly created states (or update the probability of existing ones), as a fraction of transition probability between initial states (e.g., $prob_{\phi \rightarrow \phi'}$ in line 7). Also note that we set aforementioned thresholds during the experiments.

Our goal with the regularization is to reduce the gap between the state transition probabilities of the training (modeling) trajectories, and those of the test (evaluation) trajectories. The resulting regularized Markov Model is like a finite state machine which models behavior of drivers in terms of probability of transitions between different driving states. For example, by having such a model, we may infer the speed change from 20 km/h to 50 km/h is not likely to happen. On the contrary, the change in speed by less than 5 km/h is quite likely. Figure 3 shows a sample Markov Model which contains transitions from S_1 to four other states in the model.

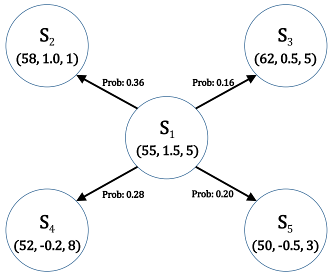


Figure 3: A cut of a sample Markov Model. Each state consists of a triple of Speed (km/h), Acceleration (m^2/s), and Change of Heading. Probability of transition is shown on each edge.

3.1.3 Trajectory Transformation. Recall that the aim of dSegment is to provide a segmentation of trajectories based on "behavior of drivers". To accomplish this goal and prior to segmentation, we apply a *transformation* on input trajectory to a *signal* in a new space which we call that Probabilistic Movement Dissimilarity (PMD) space. Given a trajectory $\gamma = \langle \rho_1, \rho_2, \ldots, \rho_n \rangle$ and a regularized Markov Model $M = \{\Phi, \Delta, \Pi\}$, we propose Algorithm 2 to map γ to a signal S_γ in PMD space. Given consecutive data points $\rho_i, \rho_{i+1} \in \gamma$, Algorithm 2 first maps them to states ϕ and ϕ', respectively. Then, the algorithm calculates how *unlikely* the transition $\phi \longrightarrow \phi'$ is, given the model M. In this algorithm, *ReturnState* returns a state corresponding to input data point ρ_i, and *ReturnProb* returns transition probability from ϕ to ϕ'. *TransitionFrom* returns a set of states R for an input state ϕ, such that $\{\phi \longrightarrow r\} \in \Delta$, for $r \in R$. Also note that if ϕ and ϕ' represent the same state with zero acceleration, then the transition is quite likely. In other words, the unlikelihood of this specific kind of self transition is zero.

Algorithm 2: Trajectory Transformation

```
   Input: $\gamma$, $M$
1  $S_\gamma \leftarrow \langle\rangle$
2  for $i = 1$ to $n$-1 do
3      $\phi \leftarrow ReturnState(M, \rho_i)$
4      $\phi' \leftarrow ReturnState(M, \rho_{i+1})$
5      $v = 0$
6      if $\phi \neq \phi'$ or $\phi.a \neq 0$ then
7          $prob_{\phi\rightarrow\phi'} = ReturnProb(M, \phi, \phi')$
8          $R \leftarrow TransitionFrom(M, \phi)$
9                                   ▷ $R = \{r \mid (\phi \rightarrow r) \in \Delta\}$
10         for $r \in R$ do
11             $prob_{\phi\rightarrow r} = ReturnProb(M, \phi, r)$
12             $v$ += $Euclidean(\phi', r) \times prob_{\phi\rightarrow r}$
13         end
14         $v = \frac{v}{|R|}$
15     end
16     $S_\gamma \leftarrow Append(S_\gamma, v)$          ▷ Append $v$ at the end of $S_\gamma$
17 end
   Output: $S_\gamma$                          ▷ $S_\gamma$ is transformed version (signal) of $\gamma$
```

Based on Algorithm 2, we map a trajectory into a signal in PMD space. The signal of a trajectory demonstrates the unlikelihood of driving behavior for each moment of the trajectory. An unlikelihood score is calculated based on the transition probabilities in the Markov Model M, which are demonstration of behavior of drivers in a population. Lines 7 to 14 in Algorithm 2 measure how far the observed transition $\phi \longrightarrow \phi'$ is from the expectation, regarding the M. Figure 4a depicts a sample trajectory, where its corresponding signal in PMD space is represented in Figure 4b. The numbers in rectangular call-outs in Figure 4a show time stamps which can be mapped to *Time* axis in Figure 4b. The larger the PMD values, the more unlikely the behavior of driver is. For instance, a large PMD value is observable for time stamp 990 in Figure 4b, where the actual trajectory in Figure 4a shows an unexpected reduction in speed and probably a lane change. The main takeaway from this sub-section is that we use a signal in PMD space as representation of behavior of a driver for a given trajectory.

3.1.4 Trajectory Segmentation. As we intend to identify the homogeneous parts of a trajectory as segments which are also representatives for driving patterns, we leverage an existing approach for segmentation of *electrical* signals, which is proposed by Han et al. [10], to find optimal segments of the signal of a trajectory. This approach is a dynamic programming algorithm that uses the Maximum Likelihood principle for segmenting one dimensional signals. Given an input signal $S = \langle x_1, x_2, \ldots, x_N \rangle$, the Maximum Likelihood (ML) of S can be defined by Equation 2.

$$ML(\theta; x_1, x_2, \ldots, x_N) = f(x_1, x_2, \ldots, x_N | \theta) = \prod_{i=1}^{N} f(x_i | \theta) \qquad (2)$$

In Equation 2, θ is the set of parameters for a probability density function (PDF) f, which can be estimated based on data points of signal S. Similar to [10], we leverage the *Gaussian distribution* to find the parameters of the PDF f. Thus, $\theta = \langle \mu, \sigma \rangle$, where μ and σ are the sample mean and the standard deviation, respectively.

Recall that the goal of segmenting a trajectory γ, so its corresponding signal $S_\gamma = \langle x_1, x_2, \ldots, x_N \rangle$, is to find a set of cutting indexes $seg_\gamma = \langle I_1, I_2 \ldots, I_n \rangle$ that maximize the likelihood within segments and minimize $n \leq N$, the best number of existing segments (see section 2). The recurrence relation to segment signal S_γ is defined by Euation 3.

$$SSC(S_\gamma, i, v) = \underset{i+5 \leq j \leq N}{\operatorname{argmax}} \left(ML(\theta; x_i, \ldots, x_j) + SSC(S_\gamma, j+1, v-1) \right) \quad (3)$$

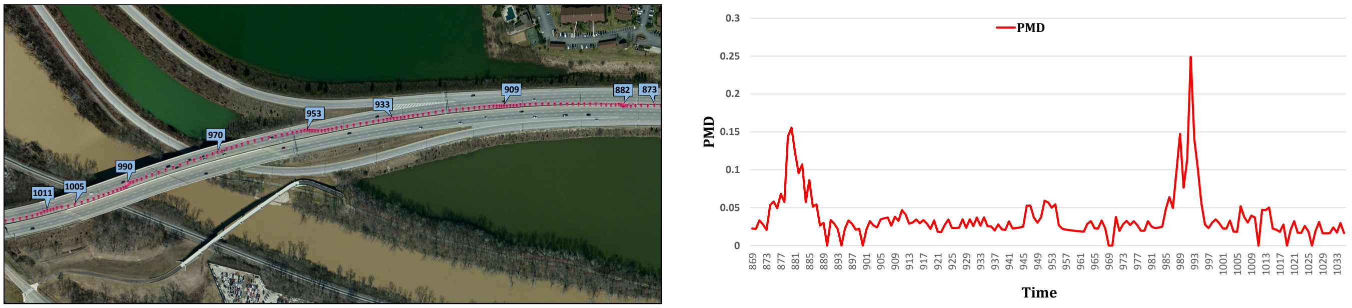
In Equation 3, $SSC(S_\gamma, i, v)$ gives the best Segmentation Score (SSC) for a sub-sequence of signal S_γ which starts at index i, with the goal being to find v segments. Also, $ML(\theta; x_i, \ldots, x_j)$ gives the maximum likelihood score for sub-sequence $\langle x_i, x_{i+1}, \ldots, x_j \rangle$ of S_γ. Note that we assume the minimum length of a segment to be five[3], this is why j starts from $(i + 5)$. The initial call for Equation 3 is $SSC(S_\gamma, 1, n)$. The interested reader may refer to [10] for more details.

The last question in this sub-section is: *how do we find the best number of existing segments within a signal?* We use the Minimum Description Length (MDL) [21] for this purpose, which has been applied in [10] as well. MDL minimizes the Equation 4 for $n = 1, 2, \ldots, K$, where n is the number of segments and K is the upper bound on the number of segments (chosen by the user):

$$MDL(n) = -ln \prod_{i=1}^{n} f(x_{I_{i-1}+1}, x_{I_{i-1}+2}, \ldots, x_{I_i}, |\theta_i) + \frac{r_n}{2} lnN \qquad (4)$$

In Equation 4, θ_i is the parameter set of the corresponding PDF, r_n is the number of estimated parameters (where n is the number of segments), and N is the length of the signal. We also set $I_0 = 0$. Figure 5 shows a part of a segmented signal which is related to the sample trajectory in Figure 4a. The blue lines in Figure 5 show the end points of segments (i.e., the cutting points). The best number of segments which has been found by MDL is 5. An interesting observation in Figure 5 is homogeneity of behavior of driver *within* segments (patterns) and dissimilarity of patterns of behavior *between* neighboring segments, which is compatible with our optimization goals. As an example of behavior-based driving pattern which is captured by dSegment, we point to the segment which starts at time

[3] This values is found empirically and it will be described later in Section 4.



(a) Sample Trajectory

(b) Signal in PMD space

Figure 4: The sample trajectory (a) and corresponding signal (b). Red dots on map show trip points, where the vehicle goes from right to left. Numbers in rectangular call-outs in (a) show time stamps which can be matched with time axis in (b). The PMD value shows the unlikelihood of driver's behavior.

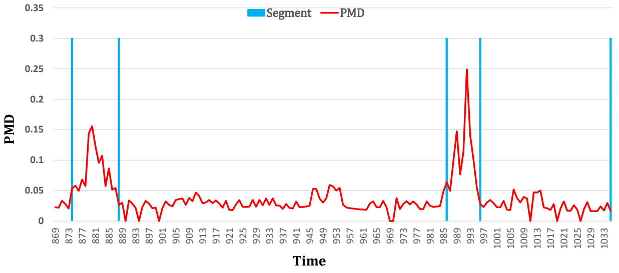


Figure 5: Segmentation of a sample trajectory by dSegment, where the best number of segments is found as 5 by MDL.

stamp 986 in Figure 5. Matching it with the actual trip in Figure 4a, we observe that this segment is related to a driving pattern where the driver reduces speed and changes the lane. It is likely that these actions are due to traffic congestion. We describe how to discover underlying causes behind driving patterns in the next section. Also, more analysis on correctness of dSegment to extract valid segments in compare to the state-of-the-art is provided in Section 4.2.

3.2 dDescribe Component

Regarding Figure 2, the dDescribe is the second important component of DriveContext which analyzes extracted driving patterns (segments) to explore the underlying causes that make each pattern happen. This step helps to identify the characteristics for a given context as we describe later in Section 4.4. As we discussed in Section 2, the existence of a driving pattern is potentially related to extrinsic or intrinsic causes. In this paper, our focus is to find the extrinsic causes, so-called *events*.

Recall that for a given trajectory γ, dSegment returns a set of cutting points $\langle p_{I_1}, p_{I_2} \dots, p_{I_n} \rangle$. Having a database of events $\mathcal{E}$ (Section 2) and a cutting point p_{I_i}, $1 \leq i \leq n$, the goal is to find whether p_{I_i} is *related* to an event $e \in \mathcal{E}$ or not. If we found that p_{I_i} is related to e, then this means the segment which starts at cutting index $(I_i + 1)$ is potentially caused by event e. Note that we accommodate the fact that a segment (pattern) can be caused by more than one event in some cases. We define the relevancy relationship between a cutting point p and an event e based on the type of the event. In this study, we consider three types of events: *physical fact*, *temporal-physical event*, and *temporal event*[4]. Following is how we measure relevancy for each type of the event:

- **Physical Fact**: An example is the presence of a traffic signal. In such a case, the relevancy can be measured as the distance between the locations of cutting point p and event e. We then say p and e are correlated if their locations are within a specified distance threshold.
- **Temporal-Physical Event**: An example is the existence of a traffic congestion in a specific place during a time interval. In this case, we say p and e are correlated if the two following conditions are satisfied: i. the time of the trajectory γ, where $p \in \gamma$, overlaps with the time interval of the event e, and ii. the distance between locations of p and e are lower than a threshold.
- **Temporal Event**: An example is having a severe storm within a specific time interval. In this case, we say p is correlated with e if the time of trajectory γ, where $p \in \gamma$, overlaps with the time interval of event e. The implicit assumption for this case is that we assume the event e is happened in the same region (city, state, etc.) as trajectory γ is happened.

In this study, we leverage "physical facts" and "temporal-physical events" to build the event database $\mathcal{E}$ to be used as input of dDescribe component. More detail about creating the event database is provided in the next section.

4 EVALUATION

In this section, we first describe the datasets which we used in this study. Then, we evaluate dSegment with respect to a ground-truth dataset. Next, we apply dSegment on a real-word dataset of car trajectories and conduct causality analysis using dDescribe. Finally, we provide examples of the applicability and usefulness of the DriveContext framework to demonstrate the implication of practice.

4.1 Dataset

We used four different sets of spatio-temporal data sources to build and evaluate components of DriveContext. These four datasets consists of (1) *Dataset of Annotated Car Trajectories (DACT)*, (2) *Nationwide Trajectories*, (3) *Physical Facts*, and (4) *Temporal-Physical Events*. We describe each dataset in the following subsections.

4.1.1 Dataset of Annotated Car Trajectories.

A dataset of annotated car trajectories (documented in Moosavi et al. [18]), is used to evaluate dSegment and compare it to other state-of-the-art segmentation approaches. In this dataset, an annotation is a *cutting point (segment border)* as described in Section 2. DACT consists of two sets of annotations for each trajectory, one that assumes *flexible* constraints to identify segment borders, and the other that uses *Strict* constraints. The former is called *Easy-Aggregation* and the latter *Strict-Aggregation*. The DACT includes 50 trajectories which cover

[4] There may be additional types of event, but we only consider the ones listed.

about 13.3 hours of driving data. The Easy-Aggregation set contains 1, 372 annotations for 50 trajectories. The Strict-Aggregation set contains 2, 465 annotations for the same set of trajectories. The reader may refer to [18] for more details.

4.1.2 Nationwide Trajectories. In order to build and to show the applicability of DriveContext framework, we use a rich, real-world dataset of trajectories we term the "Nationwide Trajectories", provided by a Fortune 100 insurance and financial services company based in Columbus, Ohio. To our knowledge, Nationwide Trajectories is one of the few large scale datasets with driving data for personal vehicles (as opposed to taxi cabs or other kinds of commercial transportation vehicles). Further, this data is precise, having been collected by highly accurate devices connected to the On Board Diagnostic (OBD-II) port of the vehicles, as well as rich, consisting of a variety of useful data items such as speed, acceleration, GPS coordinates, heading, etc. Finally, this new dataset is highly granular, with data being collected at a consistent sampling rate as 1 second for all trajectories. The Nationwide Trajectories data was collected between July 2011 and January 2014 from 103 drivers, and contains 83,406 trajectories and covering about 20,689 hours of driving data.

We divided the Nationwide Trajectories into two sets: *modeling* and *evaluation.* We use the former to build the dSegment model, and the latter to evaluate the dDescribe and also to show the application of DriveContext framework as an end-to-end solution. In order to build the evaluation set, we first sampled all the trajectories for 5 popular routes in the city (i.e., Columbus Ohio). Then, for each driver in the sampled data, we randomly chose 40% of their trajectories to be used in the evaluation set. Thus, the modeling set contains 81,895 trajectories (20,073 hours of driving), produced by 103 drivers, and the evaluation set contains 1,421 trajectories (616 hours of driving), produced by 48 drivers. Also, it is worth mentioning that for each of five common routes in the evaluation set, we have the same *start* and *end* points (on map) for trajectories in evaluation set. More details about the evaluation set is summarized in Table 2.

4.1.3 Physical Facts. Physical facts were drawn from two different sources of data, as follows:

i. Open Street Map (OSM)[5]: We used OSM as a publicly available source of annotations for different places all around the world. We only used a subset of the available annotations, specifically those related to physical facts, such as *exit/merge*, *ramp*, *bridge*, etc.

ii. Hand-Curated Annotations (HCA): Since OSM cannot be considered as a comprehensive source of annotations, we manually annotated routes in the evaluation set by using *Google Street View* and created a set of hand-curated annotations. Examples of annotations in this set include *sharp-turn* , *smooth-turn*, *exit/merge*, *intersection*, etc.

The set of physical facts contains 1,825 annotations from OSM and 95 HCA.

4.1.4 Temporal-Physical Events. One of the best examples of a temporal-physical event is *traffic congestion* which may be found in traffic congestion reports. However, since there is no publicly available historical sources of traffic congestion report which could be matched to our dataset, we used two different APIs, specifically, *Bing Traffic API* [6] and *Map Quest Traffic API* [7], to collect real-time traffic reports. A summary of current dataset of congestion reports which covers the data for a period of one year, from February 2016 to February 2017, for the routes in the evaluation set is provided in Table 1. We present more details about how we use this traffic congestion data for causality analysis in Section 4.3.2.

Table 1: Summary of Congestion Report Dataset, collected using Bing and MapQuest APIs from Feb 2016 to Feb 2017.

Route	#Bing congestion	#MapQuest congestion
Interstate-70	477	112
Interstate-71	401	1,369
Interstate-270	290	800
Interstate-670	365	1,189
315 Freeway	155	1,025

4.2 dSegment Evaluation

As shown in Figure 2, we first use the modeling set of Nationwide Trajectories to create the Markov model, setting S_{th}, A_{th}, and H_{th} in Algorithm 1 to 3, 0.5, and 6, respectively. The regularized Markov model consists of 47,495 states and about 5.8 million transitions between states. In order to evaluate our segmentation approach, we use the DACT annotation sets [18]. For comparison purposes, we use following four baselines:

- **Stable Criteria**: This approach is an alternative to dSegment, where a set of spatio-temporal heuristics are used for segmentation. A example heuristic is the *maximum* amount of the change of an input feature (e.g., speed) that can be allowed within a segment [1, 4].
- **Point of Change Detection**: As we transform a trajectory to a time series (PMD signal), instead of using the dynamic programming approach, one can use a *point of change detection* solution. Here we use a state-of-the-art approach proposed by Liu et al. [14] to first obtain the change score for each point of time series. Then, as authors described in their paper, we empirically find a threshold on the change score to find peak points, which then define our segment borders (or cutting points).
- **Equal Length**: In this approach, we first assume all trajectories have the same number of segments, say η, and then we divide a trajectory to η equal size segments. Later in this section we describe how to find η.
- **Random**: This approach is similar to the "Equal Length" approach, except we find segment borders at random, where the goal is to end up with η segments.

In order to find the upper bound on the number of existing segments, i.e., K (see section 3.1), we set $K = \frac{N}{5}$, where N is the length of trajectory. The minimum length of a segment is assumed to be 5, which is also compatible with our finding in [18]. Sine we have two sets of annotations for trajectories in DACT, we evaluate and compare our approach based on both sets. Also, we use *Precision* and *Recall* as evaluation metrics. Given a trajectory t with annotations $Ant_t = \langle a_1, a_2, \ldots, a_n \rangle$, if algorithm Alg finds cutting points $CP_{Alg} = \langle p_1, p_2, \ldots, p_m \rangle$ for t, then we use Equations 5 and 6 to define precision and recall, respectively.

$$Precision = \frac{Ant_t \cap CP_{Alg}}{m} \tag{5}$$

$$Recall = \frac{Ant_t \cap CP_{Alg}}{n} \tag{6}$$

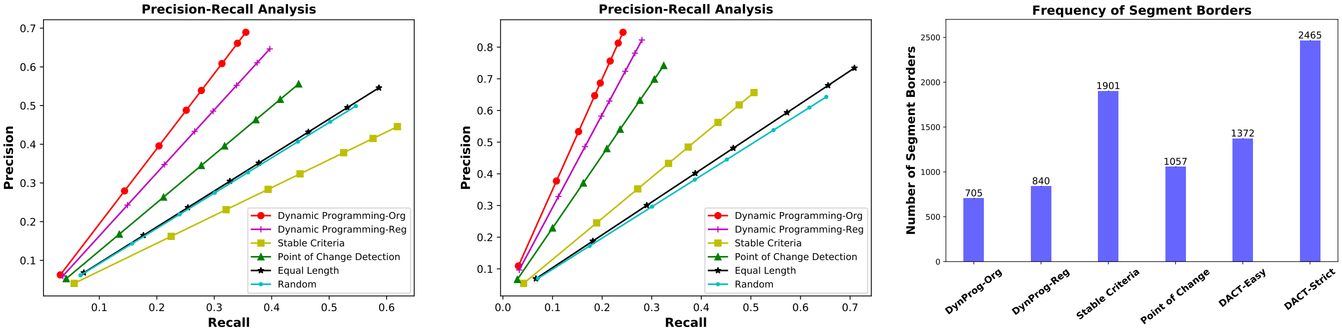
We obtain the intersection between Ant_t and CP_{Alg} as follows: to find a match for $p_i \in CP_{Alg}$, we calculate its *Haversine*[8] distance to all *available* annotations in Ant_t. If we find a pair (p_i, a_j), $a_j \in Ant_t$

[5] www.openstreetmap.org

[6] https://msdn.microsoft.com/en-us/library/hh441725.aspx

[7] https://developer.mapquest.com/products/traffic

[8] https://en.wikipedia.org/wiki/Haversine_formula



(a) Comparison based on Easy-Aggregation set

(b) Comparison based on Strict-Aggregation set

(c) Frequency of extracted Segment Borders

Figure 6: Comparing different segmentation approaches based on DACT annotations (a) and (b), and frequency of extracted segments (c).

Table 2: Summary of evaluation set and segmentation outcome.

Route	Route Length	Number of Trajectories	Avg. Trajectory Length (secs)	Avg. Number of Segments
Interstate-70	6.7 km	535	333	4
Interstate-71	13.8 km	438	546	5
Interstate-270	12.3 km	195	444	3.3
Interstate-670	7.0 km	131	359	4.2
315 Freeway	14.6 km	120	703	8.9

for $1 \leq j \leq n$, such that their Haversine distance is lower than a predefined threshold, then we say there is a match for p_i. Once that we found such a_j, we no longer use that to match other cutting points in CP_{Alg}. We use values in set $\{0, 25, 50, 75, 100, 150, 200, 250\}$ as distance thresholds, where the measure is *meters*. Taking the aforementioned into account, Figures 6a and 6b show the comparison between different segmentation approaches based on two different sets of annotations in DACT, when varying the distance threshold. Note that we report the results of dSegment by *Dynamic Programming-Org* and *Dynamic Programming-Reg*. The former refers to the case where we use the original Markov Model for segmentation, without regularization, while the latter refers to the regularized Markov Model. For Equal Length and Random approaches, we use $\eta = 30$ based on Easy-Aggregation and $\eta = 50$ based on Strict-Aggregation annotation sets. This numbers are set based on average number of segments in a trajectory, as reported in [18]. Figures 6a and 6b show that dSegment outperforms the other baselines by reasonable margins, based on both ground truth datasets. Moreover, using the original Markov Model leads to higher precision for prediction of segment borders, however, the regularized graph can improve the recall by loosing some precision points. This shows the regularization helps to generalize the model. Note that the choice of maximum distance threshold (i.e., 250 m) is based on the observation that precision and recall are not significantly changed by using larger thresholds. Also, given the average length of routes in evaluation set which is about 10 km (see Table 2), and the average number of segments for a trajectory based on DACT datasets which is about 40 [18][9], we have $\frac{10km}{40} = 250m$. Thus, using larger thresholds may invalidate the evaluation results,

Figure 6c depicts the frequency of extracted segments found by the different approaches. As one can see, dSegment and the point-of-change-detection baseline tend to extract less segments, while stable-criteria extracts many more. This can justify the difference between the recall values of different approaches in Figures 6a and 6b. Note that a solution which maximizes the precision is preferred, because we need *valid* segments to conduct a precise causality analysis to confidently derive the characteristics for a context. In Figure 6c, DACT-Easy and DACT-Strict show the number of annotations (segments) in ground truth sets. We omitted the values for Equal-Length and Random baselines as the number of extracted segments by these two approaches depends on the choice of η.

An important observation from Figures 6a and 6b is the linear relationship between precision and recall, which can be justified by the formulation of these two metrics, in that they have the same numerator but different denominators (Equations 5 and 6). The denominator for precision is the number of extracted segments (m), and for recall is the number of specified segment borders in the ground-truth set (n). As shown in Figure 6c, one can compare these numbers for different cases to obtain the slope of precision-recall lines. For instance, for the case of *DynamicProgramming-Org* and *DACT-Easy* as ground-truth set, we have $n/m = 1.95$, which is very close to the slope of corresponding line in Figure 6a, which is obtained as 1.94 by *Linear Regression* analysis.

4.3 dDescribe Evaluation

4.3.1 Context. As it is described earlier in the paper, we define the context as combination of *location* (e.g., Interstate-270, from intersection Interstate-70 to intersection US-33) and *time* (e.g., weekdays between 3pm to 7pm). We show the list of routes in Table 2. For time, we use two granularity levels: i. *Type of Day*, and ii. *Time of the Day*. The first level contains *Weekday (WD)* and *Weekend (WE)*, and the second level contains five time intervals including: *P1*: from 6am to 9:59am, *P2*: from 10am to 2:59pm, *P3*: from 3pm to 6:59pm, *P4*: from 7pm to 9:59pm, and *P5*: from 10pm to 5:59am. To find the appropriate time intervals, we used the one-year traffic congestion reports by Map Quest for the city of Columbus (Section 4.1.4). As shown in Figure 7, one can see how traffic condition (i.e., congestion frequency) is changing during the different parts of the day. Regarding such change in traffic condition, we derived aforementioned intervals.

4.3.2 Causality Analysis. After showing the applicability of dSegment (see Section 4.2), we apply it on our evaluation set. Table 2 provides some statistics about the evaluation set and also reports the average number of extracted segments for each route in the evaluation set. In summary, dSegment extracted 6,674 segments from trajectories in the evaluation set. Next, we present the result of applying dDescribe on the extracted segments to discover properties for each context. As it is described previously, we use physical facts, from OSM and HCA, and temporal-physical events, from traffic congestion reports, to create the event database $\mathcal{E}$. We conduct

[9] We have the average number of trajectories for *easy* and *strict* sets as 30 and 50, respectively.

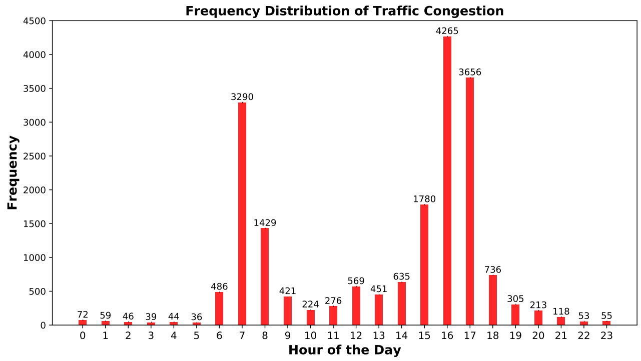


Figure 7: Frequency distribution of congestion for City of Columbus Ohio for one year (from Feb 2016 to Feb 2017), based on the Map Quest Traffic reports. One may observe how traffic condition is changing during different parts of a day. Regarding such changes, we can divide a day to different intervals.

the causality analysis by introducing a new measure *Correlation* which we define it as follows: suppose that for a set of trajectories $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_M\}$, which happened in context C, we found a sequence of cutting points $CP_i = \langle p_{i_1}, p_{i_2}, \ldots \rangle$ for each $\gamma_i \in \Gamma$, as result of segmentation process (Section 3.1). Then, given an event database $\mathcal{E}$, we use Equation 7 to obtain the correlation for context C. In this equation, the *CheckRelevancy* returns 1 or 0. Later in this section we provide more details about this function with respect to the type of event.

$$Correlation(C, \mathcal{E}) = \frac{\sum_{i=1}^{i=M} \sum_{p \in CP_i} CheckRelevancy(p, \mathcal{E})}{\sum_{i=1}^{i=M} |CP_i|} \quad (7)$$

For causality analysis, we define three different tasks based on the source of event data, as we present them next.

Event Data as Physical Facts. First, we use physical facts to build the event database $\mathcal{E}$ and then conduct the causality analysis. For this case, we define the *CheckRelevancy* function in Equation 7 as calculating the Haversine distance between a cutting point p and a physical event $e \in \mathcal{E}$, and then checking if their distance is lower than a pre-specified threshold th. We empirically set $th = 200m$, and it also takes the length of the route of each context into account. In this way, Figure 8a shows the correlation between cutting points (i.e., the extracted segments) of different contexts with physical facts. Note that correalation analysis is only done for those contexts for which we have enough data. Our observation is that on average, about 76.5% of the driving patterns (segments) are correlated with the physical properties of routes. That said, different contexts may show different patterns of correlation, depending on the properties of each context.

Event Data as Temporal-Physical Events. The second analysis is to use the temporal-physical events to build the database $\mathcal{E}$ and conduct causality analysis. The challenging part here is to define the *CheckRelevancy* function in Equation 7. To do this, we propose a solution as follows which consists of two steps.

- Step 1: Heuristic to find potential congestion evidences. Given a trajectory T, we try to find sub-trajectories of minimum length 5, where the speed of all points in such sub-trajectories is less than 55 kmh. We consider such sub-trajectories as showing potential evidence of congestion. The minimum length 5 is given from [18], and the speed 55 kmh is the average congestion speed in the traffic congestion dataset (Section 4.1.4).
- Step 2: Finalize the decision about potential evidences. After finding potential evidence C of congestion, we scan through our traffic congestion dataset (Section 4.1.4) to see if there are at least 12 evidences which happened in the *neighborhood* of location of C, within the same day of the week and hour of the day (e.g., Tuesday 4pm). The minimum number 12 is chosen to reflect the observation of one report per month for a potential congestion spot. Also, in order to define the neighborhood, we use a 200 *meters* threshold and calculate distance using the Haversine metric.

Following above guidelines, we identified 465 traffic congestion sub-trajectories within 1,421 trajectories of evaluation set. Now, the rest is straightforward: we need to find the correlation between identified congestion evidences with cutting points. For a cutting point p of trajectory T, if p is in the neighborhood (i.e., $th = 200m$) of at least one of traffic congestion evidences of T (if there is any), then *CheckRelevancy* returns 1, otherwise, it returns 0. Taking aforementioned, Figure 8b illustrates the correlation analysis results between driving patterns and temporal-physical events. On average, about 10.5% of driving patterns were correlated with traffic congestion.

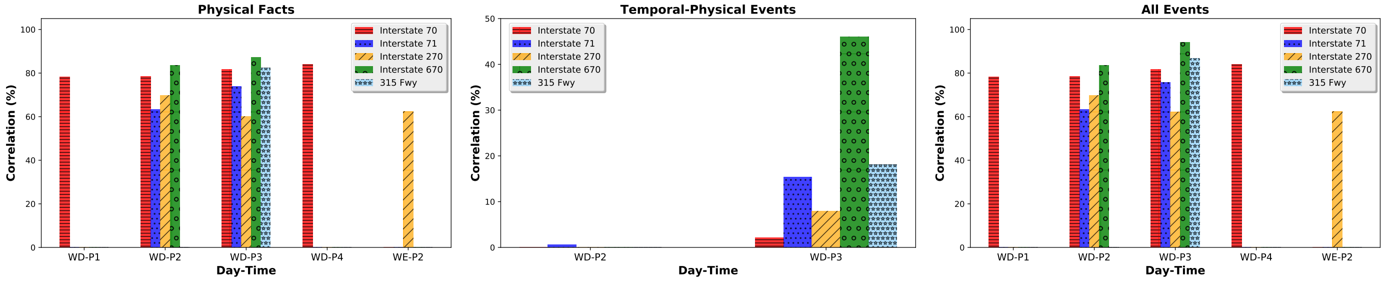
Event Data as Union of Fact and Events. Finally, we consider both physical facts and temporal-physical events to build the event database $\mathcal{E}$. Given a cutting point p of trajectory T, in order to find if it is correlated with an event $e \in \mathcal{E}$ by using function *CheckRelevancy*, we use either of approaches described previously, with respect to the type of the event. Figure 8c demonstrates the correlation of driving patterns with the set of all existing events, where, in summary 78.1% of segments are correlated with at least one of the event types. Moreover, by comparing Figures 8b and 8c, one can see that both analyses lead to almost the same patterns of correlation. This shows a significant number of segments are correlated with both sources of events.

4.4 Deriving Characteristics of Context

To this point, we have shown the applicability of dSegment, and also how to conduct the causality analysis in terms of dDescribe. Now, we show how one can use these two components to derive the characteristics for a context. We also describe other applications of the framework.

4.4.1 First-Order Insights. The results of causality analysis provides the strongest signal from which to derive the characteristics for a context. As an example, Figure 8a shows that one can expect to see about 82% correlation between driving patterns and physical properties of routes during weekdays, between 3pm to 7pm for Interstate-70. Another example is the effect of traffic on driving patterns on the context of 315-Freeway, during weekdays between 3pm to 7pm (Figure 8b).

4.4.2 Second-Order Insights. Besides the direct usage of dDescribe results, one can further analyze the results to identify *second-order insights* about a context. As an example, based on Figure 8c, one can see the correlation ratio for Interstate-270 during weekdays between 3pm to 7pm (WD-P3) is significantly lower than other contexts. Also, about 83% of evaluation data set fall into the category of WD-P3. Such lower correlation for a specific route, may be interpretable with some other events (facts), rather than what we have used in dDescribe. To understand what these events may be, we examined the available Annual Average Daily Traffic (AADT) reports which are provided by Department Of Transportation (DOT)



(a) Physical

(b) Temporal-Physical

(c) All

Figure 8: Correlation of extracted driving patterns (segments) with (a) Physical Facts, provided by Open Street Map (OSM) and Hand-Curated Annotations (HCA), (b) Temporal-Physical Events, provided by Bing and MapQuest, and (c) All Facts and Events. WD and WE are stand for weekday and weekend, respectively. P1, P2, P3, P4, and P5 are different time intervals. We show correlation for those contexts which we have enough data for them.

for Columbus Ohio[10]. Based on these reports for 2010 and 2014 (the most recent ones), as demonstrated by Table 3, we observed that the proportion of trucks that use Interstate-270 for transportation is significantly larger than other routes in the evaluation set [11]. Thus, the presence of trucks may be a potential reason for the difference in the correlation of segments in this context. Such a finding can be considered as a second order insight that is not directly derivable from the causality analysis results.

4.4.3 Other Applications of the Framework. DriveContext may also be used as an analysis tool for usage-based insurance purposes. As an example, for a given context *c*, the DriveContext may find driving patterns are on average 55% correlated with physical properties of the route. An insurance company may use such contextual information to study the behavior of an individual driver in order to evaluate how risky or safe he/she is, regarding the characteristics of context *c* (see Example 1.1). The resulting insights from our framework may also be used for driver coaching, which recommends further training to those drivers whose driving behavior in a context is not compatible with the properties of that context. Finally, by learning the characteristics for different contexts, our framework may be able to discover valuable insights about similarities and differences in driving habits for different types of the roads, different cities, regions, etc.

Table 3: Annual Average Daily Traffic (AADT) volume estimation for 2010 and 2014 for Franklin county of Columbus Ohio.

Route	I-70	I-71	I-270	I-670	315 Fwy
Truck Load (2010)	11.5%	9.7%	**13.5%**	5.3%	4.3%
Truck Load (2014)	9.4%	8.9%	**11.4%**	4.9%	3.7%
Route Length (miles)	7.5	11.1	10.5	7.7	11.2
#Vehicles (millions)	1.1	1.7	1.0	1.2	1.2

5 RELATED WORK

To the best of our knowledge, no previous research has proposed a similar framework for discovery of driving context. However, our work does relate to the research of a number of others, specifically research in *trajectory segmentation* (as used in dSegment), and *making sense of trajectories* (as discussed in dDescribe). Also, our work is related to *driving pattern discovery*. A review of related work is presented next.

5.1 Trajectory Segmentation

The task of segmentation has been addressed in the literature in several studies such as [1, 3–5]. In [4], a greedy segmentation algorithm exploits a set of monotonic spatio-temporal criteria (e.g., defining relative thresholds for some feature values) on features like speed, heading, etc. Alewijnse et al. extended this previous work to both monotonic and non-monotonic criteria [1]. However, criteria-based methods need human input for tuning parameters. Moreover, they are *context-agnostic* in the sense that they only consider the input trajectory and not the whole dataset. Therefore, the optimization process is a local one, where we propose a global optimization for segmentation. Similar to dSegment, where we propose a *context-aware* approach by building a Markov Model, Alewijnse et al. [2] presents a solution which builds a Brownian Bridge model and uses a dynamic programming approach to capture the best set of segments of animal movements. While dSegment bears some similarities with [2], it exploits a normal distribution model instead, which we find that more suitable for car transportation data. Transforming trajectory prior to segmentation is also previously discussed by [20], however, their transformation is a local approach, based on comparing line segments of input trajectory. Instead, we perform a global, likelihood-based transformation to provide a segmentation where the extracted segments represent meaningful driving patterns. Essentially, dSegment is a global optimization-based segmentation approach that builds up a model on the entire dataset. Note also that there is no need for human intervention in dSegment as in [1, 4].

5.2 Making Sense of Trajectories

Similar to dDescribe, there are some other approaches which try to make sense of driving data and to explore insights encapsulated in trajectories. Among these approaches, we can point to discovery of transportation mode [26], map matching [9, 16], points of interest (POI) discovery [15, 17], and providing descriptive summary for trajectories [27]. Besides, Wu et al. [28] recently proposed to predict traffic based on some external data sources including POI data, collision data, weather data, and geo-tagged tweet data. This is similar to our analysis in terms of dDescribe, where we try to find correlation between driving patterns and traffic congestions and physical properties of routes. The latter one is, in some sense, similar to POI. However, we pursue a different goal which is the identification of characteristics of a context.

5.3 Driving Pattern Discovery

Discovery of driving patterns (e.g., make turn, change/keep lane, etc.) has been prominently studied in the literature [6, 22, 23]. In [12],

[10] http://www.dot.state.oh.us/Divisions/Planning/TechServ/traffic/Pages/Traffic-Count-Reports-and-Maps.aspx

[11] Note that here we report the DOT data for the exact routes in the evaluation dataset, not the entire Interstate-270 for example.

a fully monitored test environment is elaborated where a small set of drivers are provided with instructions in order to measure several feature values. Then, a Hidden Markov Model (HMM) is applied to predict specific driving patterns. Driver eye movement is analyzed in [13] as an additional feature to predict driving patterns. However, all these works exploit a fully monitored context, which is costly and nearly infeasible on large-scale or to be used for Usage-Based Insurance. On the other hand, some computational based approaches proposed in the literature, like [24], which proposes a time-series matching solution to discover recurring driving patterns. While the application of this approach on large-scale data is straight-forward, there is no guarantee to find *meaningful* patterns. In the contrary, DriveContext is developed based on externally observable features which are rather easy to collect. Hence, our framework is applicable on large-scale. Also, in comparison to related work, more chances of finding meaningful patterns are available.

6 CONCLUSION AND FUTURE WORK

In this paper, we present the DriveContext framework as a solution for deriving new characteristics of a context by extracting meaningful driving patterns (dSegment), and then analyzing the extracted patterns (dDescribe) to derive insights. Our proposed segmentation approach, to our knowledge, is the first behavior-based solution which takes the behavior of a driver into consideration. Moreover, the dDescribe is also a novel solution to conduct analysis across a set of spatio-temporal data sources to find underlying causes for driving patterns. Our analysis show how the dSegment is comparable with the state-of-the-art to find meaningful driving patterns. In addition, the results of dDescribe show the ability of framework for interpretation of driving patterns which lead to new insights. The current framework can be utilized for applications like usage-based insurance, driver coaching, urban planning, etc. There are multiple lines of research to extend the current study. Regarding the dSegment, we plan to improve this component by augmenting its recall to extract more meaningful driving patterns. Besides, leveraging further external sources of spatio-temporal data to be used in dDescribe (e.g., *twitter event data*, *weather data*, and points of interest), seems to be a straightforward extension for this component. Moreover, exploring *sequences* of meaningful driving patterns by sequence mining approaches is another promising extension toward discovery of more useful characteristics for a given context.

ACKNOWLEDGMENTS

We would like to thank the Nationwide Mutual Insurance Company for their support for current research and also providing valuable sources of data and insights to conduct current research.

REFERENCES

[1] Sander Alewijnse, Kevin Buchin, Maike Buchin, Andrea Kölzsch, Helmut Kruckenberg, and Michel A Westenberg. 2014. A framework for trajectory segmentation by stable criteria. In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, USA, 351–360.

[2] Sander PA Alewijnse, Kevin Buchin, Maike Buchin, Stef Sijben, and Michel A Westenberg. 2014. Model-based segmentation and classification of trajectories. In *Dead Sea, Israel: Proceedings of the 30th European Workshop on Computational Geometry March.* ", Israel, 3–5.

[3] Aris Anagnostopoulos, Michail Vlachos, Marios Hadjieleftheriou, Eamonn Keogh, and Philip S Yu. 2006. Global distance-based segmentation of trajectories. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, USA, 34–43.

[4] Maike Buchin, Anne Driemel, Marc van Kreveld, and Vera Sacristán. 2010. An algorithmic framework for segmenting trajectories based on spatio-temporal criteria. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, USA, 202–211.

[5] Chen Chen, Hao Su, Qixing Huang, Lin Zhang, and Leonidas Guibas. 2013. Pathlet learning for compressing and planning trajectories. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, USA, 392–395.

[6] Shinko Yuanhsien Cheng and Mohan M Trivedi. 2006. Turn-intent analysis using body pose for intelligent driver assistance. *IEEE Pervasive Computing* 5, 4 (2006), 28–37.

[7] New York Taxi Dataset. 2009-2016. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml. (2009-2016). Accessed: 2017-05-08.

[8] James Elander, Robert West, and Davina French. 1993. Behavioral correlates of individual differences in road-traffic crash risk: An examination of methods and findings. *Psychological bulletin* 113, 2 (1993), 279.

[9] CY Goh, J Dauwels, N Mitrovic, MT Asif, A Oran, and P Jaillet. 2012. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, USA, 776–781.

[10] Tony X Han, Steven Kay, and Thomas S Huang. 2004. Optimal segmentation of signals and its application to image denoising and boundary feature extraction. In *Image Processing, 2004. ICIP'04. 2004 International Conference on*, Vol. 4. IEEE, USA, 2693–2696.

[11] John Krumm and Eric Horvitz. 2006. Predestination: Inferring destinations from partial trajectories. In *International Conference on Ubiquitous Computing*. Springer, USA, 243–260.

[12] Nobuyuki Kuge, Tomohiro Yamamura, Osamu Shimoyama, and Andrew Liu. 2000. A driver behavior recognition method based on a driver model framework. *SAE transactions* 109, 6 (2000), 469–476.

[13] Andrew Liu and Dario Salvucci. 2001. Modeling and prediction of human driver behavior. In *Intl. Conference on HCI.* ", USA, 1479–1483.

[14] Song Liu, Makoto Yamada, Nigel Collier, and Masashi Sugiyama. 2013. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks* 43 (2013), 72–83.

[15] Yanchi Liu, Chuanren Liu, Bin Liu, Meng Qu, and Hui Xiong. 2016. Unified Point-of-Interest Recommendation with Temporal Interval Assessment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, USA, 1015–1024.

[16] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. 2009. Map-matching for low-sampling-rate GPS trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, USA, 352–361.

[17] Raul Montoliu, Jan Blom, and Daniel Gatica-Perez. 2013. Discovering places of interest in everyday life from smartphone data. *Multimedia Tools and Applications* 62, 1 (2013), 179–207.

[18] Sobhan Moosavi, Behrooz Omidvar-Tehrani, R. Bruce Craig, and Rajiv Ramnath. 2017. Annotation of Car Trajectories based on Driving Patterns. *CoRR* abs/1705.05219 (2017), 1–10.

[19] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[20] Costas Panagiotakis, Nikos Pelekis, Ioannis Kopanakis, Emmanuel Ramasso, and Yannis Theodoridis. 2012. Segmentation and sampling of moving object trajectories based on representativeness. *IEEE Transactions on Knowledge and Data Engineering* 24, 7 (2012), 1328–1343.

[21] Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica* 14, 5 (1978), 465–471.

[22] Dario D Salvucci. 2004. Inferring driver intent: A case study in lane-change detection. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 48. SAGE Publications, SAGE, USA, 2228–2231.

[23] Amardeep Sathyanarayana, Pinar Boyraz, and John HL Hansen. 2008. Driver behavior analysis and route recognition by hidden Markov models. In *Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on*. IEEE, USA, 276–281.

[24] Stephan Spiegel. 2015. Discovery of driving behavior patterns. In *Smart Information Systems*. Springer, USA, 315–343.

[25] Neville A Stanton, Guy H Walker, Mark S Young, Tara Kazi, and Paul M Salmon. 2007. Changing drivers' minds: the evaluation of an advanced driver coaching system. *Ergonomics* 50, 8 (2007), 1209–1234.

[26] Leon Stenneth, Ouri Wolfson, Philip S Yu, and Bo Xu. 2011. Transportation mode detection using mobile phones and GIS information. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, USA, 54–63.

[27] Han Su, Kai Zheng, Kai Zeng, Jiamin Huang, Shazia Sadiq, Nicholas Jing Yuan, and Xiaofang Zhou. 2015. Making sense of trajectory data: A partition-and-summarization approach. In *2015 IEEE 31st International Conference on Data Engineering*. IEEE, USA, 963–974.

[28] Fei Wu, Hongjian Wang, and Zhenhui Li. 2016. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, Burlingame, CA, 69.

[29] Hao Fu Yu Zheng. 2011. *Geolife GPS trajectory dataset - User Guide*. MS Microsoft. https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/