The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

Spatially Aligned Clustering of Driving Simulator Data*

David Grethlein, Santiago Ontañón
Drexel University
djg329@drexel.edu, so367@drexel.edu

Abstract

We set out to compare the utility of different representations of driving simulator time series data in the context of both supervised and unsupervised learning algorithms. Given the task of identifying similar time series; it is important to understand how a dataset of time series samples might be distributed and how effectively different methods capture the groupings of distinct behaviors. First we engineer three representations of the driving simulator data: converting them to feature vectors, using the raw time series, and rendering them as images. At which point, we introduce a novel method for comparing time series using temporal and spatial alignments. Then, we employ a battery of clustering algorithms to isolate groups of samples with similar traits and evaluate the quality of clusters produced. We also explore the performance of *k-NN* classifiers using the different dissimilarity measures resulting from these representations. While our methods demonstrated some sensitivity to the data's class labels (0.43 cluster purity and classification accuracy vs. 0.33 base-line values) we assert that the classes are not easily separable; at least when time series are compared globally (suffering from being "weakly-labelled and strongly aligned" data).

***Keywords*—** Time Series Clustering, DTW, Driving Simulator

Introduction

In order to best protect everyone on the road, there must be a consistent way to identify drivers with skill deficits or cognitive impairments so to evaluate the risk they present to themselves and others. Driver safety is an established field of research where *subject matter experts* (SMEs) typically construct domain-specific *features*, variables as input for machine learning classifiers to train and test on, for the purpose of identifying and characterizing different driving behaviors. Employing SMEs can be cost-ineffective for some research efforts, and the specialized features developed may not be applicable to other studies. In recent years driving simulators have become an increasingly popular method of evaluating driving performance in a variety of settings without the motor vehicle safety concerns of on-road testing (Walshe, Oppenheimer, and Winston 2019). Being able to distinguish appropriate from dangerous driving behaviors could be invaluable for preventing motor vehicle crashes (Yasar, Berbers, and Preuveneers 2008) and improving evaluator safety within state licensing procedures by identifying unskilled drivers beforehand (McDonald et al. 2015).

With so many aspects of driver safety and driving behavior under investigation, there is a need to quickly and reliably determine which driving records are most alike and to articulate which characteristics unify them. Conversely, detecting outliers and anomalous behaviors is of equal importance when examining a cache of driving simulator data (Wong et al. 2018). Thus, this paper focuses on numerical methods for processing and identifying time series with homogeneous characteristics, distinct from other groups of time series considered to be similar, and to evaluate these methods on driving simulator data. Specifically, in this paper we compare three different representations of driving simulator time series data: (1) converting them to manually engineered feature vectors, (2) using the raw time series data, and (3) rendering them as images with the driving paths. We then compare a collection of distance and dissimilarity functions in the context of input for partitioning algorithms. In addition to comparing existing functions, we also propose a new one, which we call *Dynamic Coordinate Locally Aligned Warping* (DCLAW). DCLAW is an an extension of FastDTW (Salvador and Chan 2007), that leverages the *spatiotemporal* (measurements through space over a period of time) aspect of the driving simulator data to align time series. Finally, we compare and contrast all these functions in terms of classification accuracy in automatically detecting drivers diagnosed with ADHD using a *k-NN* classifier. Particular concern is paid attention to the scalability of all considered methods, as large troves of driving simulator data may be computationally infeasible to process by traditional methods.

The rest of this paper is organized as follows: first we review several recently developed approaches to clustering time series data. Next, we discuss the driving simulator data collected and analyzed for this study. Then, we describe the dissimilarity measurements used to construct matrices as the basis for clustering. Finally, we review our experimental results and posit several extensions to our algorithms as future work for manipulating larger datasets and distilling clusters more indicative of the data's class labels.

Background

This section presents a brief introduction to *Dynamic Time Warping* (DTW), one of the most common techniques to assess the dissimilarity between time series (since much of our work builds upon DTW), as well as some brief description of the various clustering algorithms used in this paper. Moreover, throughout this paper, we will use the term *samples* to refer to individual time series within a dataset, i.e. to the time series that represent a single driving simulator session in our application domain.

*This work is supported by NSF Grant No. 1521943; in a collaboration on part of the Children's Hospital of Philadelphia (CHOP), Drexel University, George Mason University, the University of Central Florida, and Diagnostic Driving, Inc.

Dynamic Time Warping

Dynamic Time Warping (DTW) has long been the preferred technique for assessing the dissimilarity between time series in countless studies (Ten Holt, Reinders, and Hendriks 2007). Intuitively DTW aligns two time series, stretching one of them along the time dimension, trying to find the time warping that minimizes their difference, i.e., the aggregate residual distance between aligned time steps. The major drawback of DTW is its $\mathcal{O}(n^2)$ complexity for time series of length n, prohibiting its effective use for larger and longer time series datasets. As a brute force method, DTW will always generate the minimum warping distance between time series. There have been numerous well-documented extensions to the original algorithm that can vastly increase its scalability and predictive power (Al-Naymat, Chawla, and Taheri 2009; Hartmann and Link 2010; Bettens and Todoroff 2009).

For example, using *Piecewise Aggregate Approximations* (PAAs), which has been successful in non-trivially reducing the number of pairwise comparisons for computing time series similarity by condensing multiple time-steps into fewer frames through averaging. Applying *abstraction*, the moving average yields a trade: sacrificing the approximation's fidelity to the original series for a substantially quicker computation time (Keogh and Pazzani 2000). Compressing the resolution of comparison across time axes too much may wash out the discernible features that typify a certain class of time series. More recent efforts utilize a derivative algorithm called FastDTW that scales linearly to compute near-optimal warping dissimilarity between time series samples.

During our review of the literature we found no existing variants of DTW that leverage the spatial alignment of time steps when positional data is available, which is the basis of our proposed dissimilarity function, DCLAW.

Partitioning Algorithms

In the experimental evaluation presented in this paper, we employed the following algorithms:

- k-Means divides a dataset into k clusters by identifying k centroids (cluster centers), and assigning each sample to the closest centroid. A problem of *k-Means* is that it cannot distinguish clusters from noisy samples, a problem exacerbated by the temporal element of time series data (Ernst, Nau, and Bar-Joseph 2005; Ding et al. 2008). *k-Medoids* is a variant of k-Means, which does not require calculating the mean of collections of points in the dataset, since it constraints the set of possible centroids to the actual points in the dataset. Thus, it is useful for domains where calculating the mean of two points is ill-defined (such as time series data).
- DBSCAN (Ester et al. 1996): *Density Based Spatial Clustering of Applications with Noise* is a clustering algorithm based on grouping together points that are closely packed together. Points in areas of low density left ungrouped are marked as outliers.
- *Shared Nearest Neighbor clustering* (SNN) (Ertoz, Steinbach, and Kumar 2002) is an alternative to DBSCAN that uses the number of nearest neighbors two points in a dataset share as an affinity measure. Noise is separated from clustered points by forming a graph from these affinities and imposing a threshold on the minimum number of shared points to reduce the number of edges to preserve.
- Spectral clustering (Shi and Malik 2000): uses the eigenvalues of the dissimilarity matrix to perform dimensionality reduction before performing clustering. It has been shown that using the FastDTW dissimilarity measure for comparing velocity profiles as a basis for spectral clustering can successfully predict vehicle states (Lohrer and Lienkamp 2015).
- k-NN: A *k-Nearest Neighbor Classifier* (Cover and Hart 1967) is a supervised learning methods that assigns a label to an unseen sample by looking for the k most similar samples in the training set, and then predicting the majority label amongst these k most-similar neighbors. Using DTW and its derivatives as the basis for constructing a *k-NN* classifier has yielded promising results across many domains (Bagnall and Lines 2014).

Methods

In this section we give a detailed description of the origins of the data used in all experiments reported on herewithin, and the representations with which samples were compared. We review dissimilarity measures such as the FastDTW algorithm, and present our extension to it, DCLAW; both used to compare the time series directly. At which point, we describe the configurations used in the partitioning algorithms listed above and their respective parameters' implications on partitions produced. Lastly, we describe the evaluation metrics used to measure the success of each partitioning as it pertains to both unsupervised and supervised learning tasks.

Simulator Data

Our data was recorded from 30 participants in a behavioral study of adolescent drivers conducted at CHOP (NSF Grant No. 1521943), 15 of whom had confirmed diagnoses of ADHD and had been prescribed medications by their physicians prior to being considered as candidates. Participants with confirmed diagnoses of ADHD were considered the *variable* group as they were instructed to drive all 4 simulated tracks (Drive 1, Drive 2, Drive 3, Drive 4) twice: once while taking their medication (*regulated*), and once while taking a placebo (*delayed*). The other 15 individuals, the *control* group, had no such diagnoses and were instructed to drive the simulated tracks only once. One participant opted to stop recording part-way through their session due to motion sickness, so we removed the incomplete drive from the dataset ($N = 179$ total sessions).

For all our experiments we recorded time series data composed of multiple channels sampled at 60Hz; however, we down-sampled to 10Hz by selecting every 6th frame to reduce computation times for all similarity measures. Acknowledging that this method is susceptible to picking up noise from higher frequencies, we opted not to use PAA to avoid smoothing of already homogeneous characteristics within a small dataset. Recordings varied in length from 3,829 to 9,746 frames, with an average of 5,728 frames; this is equivalent to 6-17 minutes with an average of roughly 9 and a half minutes. Several dozen channels compose each recording; we limit our experiments to using a few to minimize reliance on domain knowledge:

- pos_x : The x coordinate on the simulation's Cartesian map.
- pos_z : The z coordinate on the simulation's Cartesian map.
- *throttle* : The percent depression of the accelerator pedal where 0 corresponds to foot off the gas, and 1 is hard acceleration.
- *brake* : The percent depression of the brake pedal, where 0 corresponds to foot off the brakes, and 1 is hard braking.
- *steering* : Signed percentage of rotation of the steering wheel from its resting position at center normalized to the interval $[-0.5, 0.5]$ where the extremes correspond to one and a half complete rotations from center in either direction (negative being left, and positive being right).
- *mph speed diff* : The miles per hour difference between the simulated player vehicle and the posted speed limit on the stretch of road they are driving in each frame.

Representations of Simulator Data

Going forward we refer to the time series channels described in the previous section as the *raw representation*. We explored two strategies for directly manipulating this representation: comparing the *inputs* (throttle, brake, and steering channels) from both samples; and comparing the *speed diff* (mph speed diff channels).

Intuitively, the *inputs* should help establish the partitioning power of only considering each participants' interactions with the simulator's physical hardware (use of steering wheel, brake and accelerator pedals). Alternatively, the *speed diff* should establish the partitioning power of only considering each participants' vehicle speed within the context of speed-limited environments (requires domain knowledge of posted speed limit and vehicle speed).

We pursue both strategies with and without the *map coordinates* (pos_x and pos_z channels) of the simulated vehicle in each frame; to determine if temporal or spatial alignments produce dissimilarities with more partitioning power over the dataset.

Each driving simulator session, which we call a *sample*, is recorded and stored as a *replay file* representing a multivariate T composed of one univariate time series T_i (a.k.a., a *channel*) for each variable i we recorded: *time series* $T = \{T_1, T_2, ..., T_d\}$. Each *channel* $T_c = [T_{(1,c)}, T_{(2,c)}, ..., T_{(n,c)}]$ tracks a particular value frame by frame over time-steps $[t_1, t_2, ..., t_n]$, referred to as the *time axis*. The vector of values from all channels at time-step t_j is denoted $T_{(j,:)}$.

The *features representation* embodies the traditional approach for building classifiers that are heavily reliant on domain knowledge and manual feature construction. Each simulator session is reduced to a feature vector of 43 global values intended to quantify known aspects of driving behaviors. Features computed to assess vehicle speed relative to the posted speed limit, braking, acceleration, steering, the number of times the vehicle came to a complete stop, lateral deviation from lane center, mismatch between the road-following direction and vehicle heading, and the vehicle's lateral distance from the center of the road.

Our third approach, the *image representation*, also has fixed dimensions for evaluating each sample; attempting to embed each session by plotting the vehicle trajectory as a 3-channel RGB image (plotted at size: 1470 by 850 pixels). Each plot is a birds-eye view framed by the coordinates of the planned route through the *variant* (simulated track). Every frame of the session is illustrated as a plot-point against a black background, where coloration is determined by the *inputs* in the given frame (green representing throttle, red representing brake, and blue representing steer) and positioned by the *map coordinates*. Plot-points congregate densely when the participant vehicle is moving slowly through the environment and more sparsely when the vehicle is moving faster. Plots are resized to 350 by 200 pixels with averaging to reduce dimensionality of samples as well as accounting for variance in vehicle positions along the route. Figure 1 shows one such image.

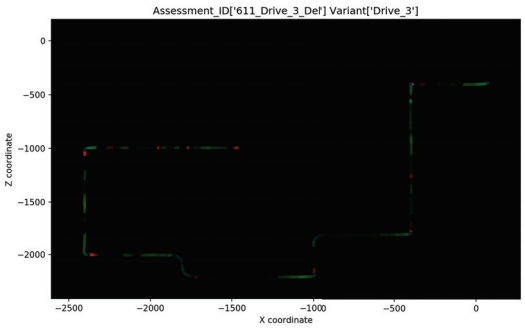

Figure 1: Plot-points are circles with radii of 5 pixels and alpha transparency set to %20 (we don't want to ignore many plot-points overlapping when the vehicle moves slowly). RGB channels of every plot-point expect values in $[0, 1]$ so we manipulate the *inputs* to colorize each one: $(throttle\ ,\ brake\ ,\ 2 * abs(steering))$.

Dissimilarity Matrices

In order to apply clustering, we precomputed pair-wise dissimilarity matrices between all N samples using a collection of functions, resulting in non-negative *dissimilarity matrix* $X \in \mathbb{R}_{N \times N}$. This section enumerates the list of dissimilarity/distance functions used in our experiments for each of the three representations used.

The *features* and *images* representations of the data are compared using *distance metrics*; i.e. dissimilarity measures that satisfy the conditions of: obeying the triangle inequality; non-negativity; symmetry; and the identity of indiscernibles. These mathematical properties provide assurances against *false dismissals*, confusing dissimilar samples with similar samples; though such metrics are often too rigid to work well directly manipulating temporal data of constant dimension (Ratanamahatana and Keogh 2004).

Features Representation For the *features representation*, each feature vector $\vec{v}$ is stored as a row in a 2-D matrix $V \in \mathbb{R}_{N \times 43}$. *Manhattan* ($\mathcal{D}_1$) and *Euclidean* ($\mathcal{D}_2$) distances where used to assess the dissimilarity between row vectors of V to construct dissimilarity matrices X_{Man} and X_{Euc}. Accounting for the potential numerical dominance of any particular feature, we separately normalize the columns of V to have 0 means and standard deviations of 1 each. These normalized feature vectors and the same distance metrics construct dissimilarity matrices $\widehat{X_{Man}}$ and $\widehat{X_{Euc}}$.

Images Representation Comparing samples with the *image representation* requires manipulating tensors; each of which has been reduced to constant dimensions ($\mathbb{R}_{350 \times 200 \times 3}$). Acknowledging that there are vastly more intensive means of computing image dissimilarity (Chen and Chu 2005), we opt to simply flatten each tensor into a 1-D vector. The distances between flattened vectors are computed with Euclidean and Manhattan distances to construct dissimilarity matrices X_{IMan} and X_{IEuc}, respectively.

Raw Representation Disimilarity between samples A and B using the *raw representation* is computed numerically with FastDTW by finding the approximate minimum distance warp path $\rho_{FastDTW} =< (i_1, j_1), (i_2, j_2), ... >$ aligning each time step $A_{(i,:)}$ to a $B_{(j,:)}$. Residual distance between time steps is calculated $\mathcal{D}(A_{(i,:)}, B_{(j,:)})$. The approximate minimum warping distance $\mathcal{D}_{FastDTW}$ is calculated by recursively reducing A and B to coarser resolutions using PAA's, and projecting minimum warping paths computed to search areas in finer resolutions. The parameter *radius* defines the width of the search area around each projected warping path at all resolutions; where larger values increase the chances of finding the true minimum distance warping path at the cost of increased computation times.

For the *input* time series, FastDTW dissimilarity matrices $X^{(r)}_{FIn_{p(\mathcal{D})}}$ are constructed using both $\mathcal{D} = \mathcal{D}_1$ and $\mathcal{D} = \mathcal{D}_2$ with radii $r = \{1, 2, 4, 8, 16\}$. Using the same permutations of parameters and the *speed diff* time series, FastDTW dissimilarity matrices $X^{(r)}_{FSD_{(\mathcal{D})}}$ are constructed. This method attempts to meaningfully compare samples by their performances aligned temporally.

Algorithm 1: DCLAW for time series A , B with alignment distance $\mathcal{D}_{align}$, *FastDTW* radius r, and residual distance $\mathcal{D}_{resid}$.

Result: $\mathcal{D}_{DCLAW}$
$A = \{A_{align}$ channels, A_{perf} channels$\}$;
$B = \{B_{align}$ channels, B_{perf} channels$\}$;
ρ_{align} =*FastDTW*$(A_{align}, B_{align}$; $\mathcal{D}_{align}, r)$;
$\mathcal{D}_{DCLAW} = 0$;
for (i, j) **in** ρ_{align} **do**
  $\mathcal{D}_{DCLAW} += \mathcal{D}_{resid}(A_{perf_{(i,:)}}, B_{perf_{(j,:)}})$;
end
return $\mathcal{D}_{DCLAW}$;

The central purpose of using simulated driving environments is to suss out how different drivers react when commonly exposed to the same road conditions. For that reason, it's imperative to understand variance in behaviors over time aligned spatially (i.e. how do different drivers behave when navigating the same stretch of road).

We propose a new alignment method for DTW we call *Dynamic Coordinate Locally-Aligned Warping* (DCLAW) to compute a warping path ρ_{align} for *alignment channels* (i.e. *map coordinates* for all experiments) with FastDTW, and then compute the aggregate residual distance $\mathcal{D}_{DCLAW}$ between time-steps of the *performance channels* defined by ρ_{align}. Thus, intuitively, DCLAW first finds the best warping path to align the spatial trajectories of two driving simulator traces, and then uses that alignment to compare the other variables (e.g., speed, throttle and brake usage).

Since our *map coordinates* exist in a Euclidean plane, we always use $\mathcal{D}_{align} = \mathcal{D}_2$ to compute ρ_{align}. Using the *inputs* time series as the performance channels, *DCLAW* dissimilarity matrices $X^{(r)}_{DInp_{(\mathcal{D})}}$ are constructed with the same parameterizations as the FastDTW methods ($\mathcal{D} = \mathcal{D}_{resid}$). The same is true for *DCLAW* dissimilarity matrices $X^{(r)}_{DSD_{(\mathcal{D})}}$ constructed with the *speed diff* time series as the performance channels. Algorithm 1 shows the pseudo code of DCLAW.

Given time series A, B of lengths n, m respectively, *DCLAW* scales $\mathcal{O}(n+m)$; experimentally, the majority of the computational overhead is incurred obtaining the warping path from the alignment channels. That is unless there are significantly more performance channels than alignment ones. In which case, walking the approximate minimum distance warping path ρ_{align} to compute aggregate residual distance $\mathcal{D}_{DCLAW}$ still requires $\mathcal{O}(n + m)$ steps.

Partitioning Algorithms

All dissimilarity matrices X constructed were used as the input for several popular clustering and classification algorithms:

- *DBSCAN*: DBSCAN has a collection of parameters. The first is the size of the neighborhoods ϵ. We tested $\epsilon = \{0.1, 0.2, 0.3, ..., 0.7, 0.8, 0.9\}$. The second is the minimum number of connections to other points a graph point must have to be considered a *core point*. We tested $min_{edges} = \{1, 2, 3, 4, 5, 10, 15, 20, 25\}$.
- *k-Medoids*: For all dissimilarity matrices, we tested $k = \{2, 3, 4, 5, ..., 18, 19, 20\}$. Since *k-Medoids* relies on random initialization and the final clusters may therefore not be consistently obtained; we ran 100 trials for each k, reporting results from the trial with maximum median *silhouette value*.
- *k-Nearest Neighbors Classification*: We tested $k = \{1, 2, 3, 4, 5, 10, 15, 20, 25\}$.
- *Shared Nearest Neighbor Clustering*: We tested $k = \{1, 2, 3, 4, 5, 10, 15, 20, 25\}$ for the number of neighbors to consider. The second parameter of SNN is the minimum affinity ϵ two points must have for their connection to be preserved. We tested $\epsilon = \{1, 2, ..., k-1\}$. After graph reduction, samples with at least min_{edges} connections are nominated as *core points*. We tested $min_{edges} = \{1, 2, 3, 4, 5, 10, 15, 20, 25\}$.
- *Spectral Clustering*: To construct adjacency and degree matrices A and D, respectively; spectral clustering uses each sample's *k-nearest neighbors*. We tested $k = \{1, 2, 3, 4, 5, 10, 15, 20, 25\}$. Next, samples are projected onto e eigenvectors of the Graph Laplacian matrix $L = D - A$, corresponding to the e smallest eigenvalues; we tested $e = \{2, 3, 4, 5, ..., 18, 19, 20\}$. Finally, we perform 20 trials of *e-Means* clustering on the projected data (accounting for random initialization of cluster centers).

Evaluation Metrics

The quality of labellings produced are assessed quantitatively to determine the fitness of suggested partitions derived from X with respect to the ADHD class labels. For all experiments, only clustered samples (i.e. not noisy points) were included in the computation of evaluation metrics. When using *SNN* and *Spectral Clustering* we record ξ, the fraction of the dataset considered to be noise.

- *Accuracy* : The percentage of correct classifications made for true class labels (*k-NN classification* only).
- *Silhouette Score* : Each clustered sample in a dataset has a relative "fitness" to its cluster; that being the relative distance from its assigned cluster center with respect to the distance for the next nearest cluster's center (Rousseeuw 1987). Samples perfectly fit to their cluster have silhouette values of 1, samples definitely incorrectly grouped have silhouette values of -1. Near 0 means a sample is on the decision boundary and not strongly associated with either potential cluster assignment. This intuition holds especially true when clusters are Gaussian in shape, though it may not be indicative of success with all cluster shapes. We track the *Median Silhouette Value* (MSV) to indicate that at least $\%50$ of clustered samples had the reported fitness or better to their assignments.
- *Purity* : The degree to which the dominant class label in each cluster composes the entirety of the cluster membership, taken as an average over all clustered samples. A proxy for accuracy.

$$Purity(\text{Clusters } C, \text{Classes } D) = \frac{1}{N}\sum_{c \in C} \max_{d \in D} |c \cap d|$$

- *V Measure* : The overall performance of separating class labels when partitioning a dataset may be judged as a proxy for accuracy in unsupervised learning tasks when such labels are available. To do so, we compute *homogeneity*, the amount which clusters contain uniform true class labels; and *completeness*, the amount which samples of the same class labels are grouped into single clusters (Rosenberg and Hirschberg 2007).

$$VM = \frac{2 * homogeneity * completeness}{homogeneity + completeness}$$

Results

Table 1 shows our empirical results, organized such that the best results for all partitioning algorithms are reported in each row. We used the following procedure to determine the "best" results for each partitioning algorithm: for the DBSCAN, SNN, and Spectral Clustering algorithms the configurations that maximize MSV are considered the best. For the *k Medoids* algorithm, the trial identified

by *Elbow Method*, examination of the largest value of *k* whereby $MSV_{k-1} - MSV_k$ is maximized, is reported. For the k-NN algorithm, the trial with *k* that maximizes accuracy is reported. Moreover, while a parameter for Spectral Clustering, the number of resultant clusters (# C) is not an explicit parameter for the DBSCAN and SNN algorithms, we report it for completeness.

Overall, we found that the ADHD class labels were highly comingled as Table 1 shows, none of the algorithms cleanly partitioned the dataset along those lines (max VM < 0.1). DBSCAN struggled to capture clusters of different densities for most similarity matrices (# C = 2 and Purity ≈ 0.34), considering large portions of the dataset as noise ($\xi > 0.5$). The notable exception for DBSCAN are from dissimilarity matrices constructed using the *inputs* time series channels via DCLAW; producing uniform density clusters that capture the entire dataset ($\xi = 0.0$) with no indication of separating class labels (VM ≈ 0.01 and Purity ≈ 0.34). As a result of this uniformity, DBSCAN likely struggled to distinguish samples driven along the different simulated tracks (# C < 4).

The elbow method for selecting the most appropriate *k* in *k Medoids* tended to partition the data into more groups than the other algorithms (k typically > 5). Dividing the data into more groups distills clusters with elevated purity (> 0.39) and provides some evidence that class labels and the partitioning are correlated (VM ≈ 0.04). The *speed diff* FastDTW dissimilarity matrices generated prototypical behaviors that were indicative of common driving patterns (e.g. smooth vs. jerky braking and accelerating; speeders vs. timid drivers) showed some signs of corresponding to ADHD class labels (VM $= 0.05$ and purity > 0.4). The features representation with dissimilarity computed via $\mathcal{D}_2$ performs even better at isolating such prototypical behaviors, partially along the lines of ADHD class labels (VM $= 0.04$ and purity $= 0.47$).

Computing distance as dissimilarity between the engineered features of samples still produces the best classification accuracy (0.43 for $\mathcal{D}_2$, 0.4 for $\mathcal{D}_1$) with a larger neighborhood size ($k = 22$ for $\mathcal{D}_2$, $k = 23$ for $\mathcal{D}_1$). Rendering samples as images to compute dissimilarity via $\mathcal{D}_1$ relied on fewer neighbors ($k = 9$) and distilled groups with the strongest evidence of aligning with ADHD labels (purity $= 0.49$ and VM $= 0.09$) suggesting *k-NN* is susceptible to conflating the class labels with one another (Acc $= 0.37$). Since the class prior probabilities are roughly 0.33, any accuracy higher than that is an improvement in ADHD detection in drivers. Marginal gains in accuracy reinforce the extreme difficulty inherent to the task of partitioning the time series data along ADHD class labels.

Using the *speed diff* channels for computing dissimilarity via DCLAW, SNN consistently captured clusters (# C = 4 and Purity ≈ 0.34) along the lines of the simulated tracks driven (Drive 1,...,Drive 4). Results where the majority of the dataset was treated as noise ($\xi > 0.5$) suggest that samples aren't similar enough for SNN to detect any overarching patterns in the data. When DCLAW (parameterized with $\mathcal{D}_1$) computes dissimilarity between the *inputs* channels, SNN considers a large portion of the dataset to be noise ($\xi \approx 0.63$); isolating prototypical behaviors that showed some alignment with ADHD class labels (VM $= 0.07$ and purity ≈ 0.4).

Spectral clustering showed little signs of producing clusters that aligned with ADHD class labels (VM ≈ 0.00). Instead it captured more obvious divisions in the data; e.g. dissimilarity derived from the *speed diff* channels separated samples by the simulated track that was driven (# C $= 4$ and purity $= 0.34$). Dissimilarities computed using the *features* and *images* representations did not distinguish samples from different tracks as clearly (# C < 4). FastDTW dissimilarity between the *inputs* channels isolated prototypical behaviors (# C = 6) that are loosely associated with class labeling (VM $= 0.01$ and purity ≈ 0.39). Given the small dataset ($N = 179$), it is difficult to surmise one way or the other if such prototypes are generalizable to larger populations of drivers.

Conclusions

In this paper we presented an empirical evaluation of methods for computing dissimilarity between time series samples, leveraging the spatiotemporal nature of the simulator data intended to uncover common behavioral sequences. We employed a battery of partitioning algorithms to break that data apart into groups to isolate these prototypical behaviors.

All approaches relied on global dissimilarity measurements between drives, where highly predictive traits may only occur in critical interviews along the planned routes and are therefore washed out. Moreover, examining segments of drives in isolation may strengthen both the connection from class labels to distinct events and the alignment of time series considered. While the distilled prototypical driving behaviors didn't cleanly partition class labels, spatial comparisons such as image embedding and DCLAW showed some sensitivity to the presence of ADHD. We plan to assess the utility of DCLAW in other time series domains, including those with known separable classes, to further develop an understanding of its ability to isolate similar time series relative to FastDTW.

We plan to explore several other methods for developing prototypical behavior profiles not heavily reliant on domain knowledge. For example, we are interested in discretizing the time series into sequences of state-action pairs and attempt to learn patterns of participant interactions with the physical inputs (Ontañón et al. 2017).

References

Al-Naymat, G.; Chawla, S.; and Taheri, J. 2009. Sparsedtw: A novel approach to speed up dynamic time warping. In *Proceedings of the Eighth Australasian Data Mining Conference-Volume 101*, 117–127. Australian Computer Society, Inc.

Bagnall, A., and Lines, J. 2014. An experimental evaluation of nearest neighbour time series classification. *arXiv preprint arXiv:1406.4757*.

Bettens, F., and Todoroff, T. 2009. Real-time dtw-based gesture recognition external object for max/msp and puredata. *Proc. SMC* 9:30–35.

Chen, C.-C., and Chu, H.-T. 2005. Similarity measurement between images. In *29th Annual International Computer Software and Applications Conference (COMPSAC'05)*, volume 2, 41–42. IEEE.

Cover, T., and Hart, P. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13(1):21–27.

Ding, H.; Trajcevski, G.; Scheuermann, P.; Wang, X.; and Keogh, E. 2008. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment* 1(2):1542–1552.

Ernst, J.; Nau, G. J.; and Bar-Joseph, Z. 2005. Clustering short time series gene expression data. *Bioinformatics* 21(suppl_1):i159–i168.

Ertoz, L.; Steinbach, M.; and Kumar, V. 2002. A new shared nearest neighbor clustering algorithm and its applications. In *Workshop on clustering high dimensional data and its applications at 2nd SIAM international conference on data mining*, 105–115.

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, 226–231.

Hartmann, B., and Link, N. 2010. Gesture recognition with inertial sensors and optimized dtw prototypes. In *2010 IEEE International Conference on Systems, Man and Cybernetics*, 2102–2109. IEEE.

Keogh, E. J., and Pazzani, M. J. 2000. Scaling up dynamic time warping for datamining applications. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 285–289. ACM.

	DBCSAN							K Medoids				KNN Classifier					SNN							Spectral				
Dissimilarity Matrix	ϵ	ME	# C	ξ	MSV	Pur	VM	K	MSV	Pur	VM	K	Acc	MSV	Pur	VM	ϵ	K	# C	ξ	MSV	Pur	VM	K	# C	MSV	Pur	VM
FEATS EUC	**0.3**	**1**	**2**	**0.00**	**0.74**	**0.34**	**0.01**	**12**	**0.40**	**0.47**	**0.04**	**22**	**0.43**	**0.17**	**0.43**	**0.02**	**5**	**5**	**3**	**0.97**	**0.47**	**0.67**	**0.58**	**15**	**3**	**0.44**	**0.37**	**0.01**
FEATS MAN	0.1	4	6	0.37	0.51	0.41	0.02	3	0.39	0.35	0.01	23	0.40	0.10	0.40	0.01	19	20	8	0.82	0.57	0.61	0.25	10	2	0.46	0.34	0.00
NORM FEATS EUC	0.5	1	2	0.00	0.64	0.34	0.01	3	0.41	0.34	0.01	21	0.38	0.06	0.38	0.01	9	10	4	0.95	0.58	0.78	0.51	5	2	0.43	0.34	0.00
NORM FEATS MAN	0.1	5	2	0.68	0.74	0.37	0.00	17	0.16	0.46	0.05	23	0.38	-0.01	0.38	0.01	19	20	3	0.97	0.77	0.50	0.30	10	2	0.50	0.34	0.00
IMAGE EUC	0.5	3	4	0.71	0.28	0.38	0.06	8	0.19	0.38	0.02	12	0.35	-0.01	0.36	0.00	9	15	3	0.01	0.21	0.34	0.00	15	3	0.20	0.34	0.00
IMAGE MAN	**0.4**	**5**	**5**	**0.00**	**0.05**	**0.38**	**0.03**	**3**	**0.24**	**0.34**	**0.02**	**9**	**0.37**	**-0.43**	**0.49**	**0.09**	**20**	**20**	**2**	**0.53**	**0.21**	**0.39**	**0.01**	**4**	**2**	**0.43**	**0.36**	**0.01**
Inp DCLAW EUC Rad 1	0.5	1	2	0.00	0.27	0.34	0.01	5	0.33	0.36	0.02	24	0.38	0.00	0.38	0.01	14	15	2	0.92	0.56	0.47	0.19	20	2	0.38	0.34	0.00
Inp DCLAW EUC Rad 2	0.5	1	2	0.00	0.27	0.34	0.01	9	0.14	0.37	0.02	22	0.35	-0.04	0.36	0.01	10	20	3	0.00	0.21	0.35	0.00	20	2	0.38	0.34	0.00
Inp DCLAW EUC Rad 4	0.5	1	2	0.00	0.27	0.34	0.01	4	0.36	0.35	0.01	24	0.36	-0.01	0.37	0.01	10	20	3	0.00	0.21	0.35	0.00	20	2	0.38	0.34	0.00
Inp DCLAW EUC Rad 8	0.5	1	2	0.00	0.27	0.34	0.01	5	0.34	0.37	0.02	24	0.36	-0.01	0.37	0.01	10	20	3	0.00	0.21	0.35	0.00	10	2	0.38	0.34	0.00
Inp DCLAW EUC Rad 16	0.5	1	2	0.00	0.27	0.34	0.01	6	0.35	0.38	0.04	24	0.36	-0.01	0.37	0.01	14	15	2	0.93	0.56	0.58	0.38	10	2	0.38	0.34	0.00
Inp DCLAW MAN Rad 1	0.5	1	3	0.00	0.26	0.35	0.02	5	0.36	0.35	0.02	19	0.35	0.03	0.36	0.00	23	25	6	0.64	0.27	0.41	0.06	25	2	0.38	0.34	0.00
Inp DCLAW MAN Rad 2	0.5	1	3	0.00	0.26	0.35	0.02	6	0.22	0.37	0.01	18	0.36	0.00	0.37	0.01	17	20	6	0.27	0.22	0.38	0.01	5	2	0.38	0.34	0.00
Inp DCLAW MAN Rad 4	0.5	1	3	0.00	0.26	0.35	0.02	7	0.22	0.36	0.02	22	0.36	0.00	0.38	0.01	23	25	7	0.64	0.23	0.40	0.07	15	2	0.38	0.34	0.00
Inp DCLAW MAN Rad 8	0.5	1	3	0.00	0.26	0.35	0.02	7	0.19	0.35	0.02	22	0.37	-0.01	0.39	0.02	23	25	7	0.64	0.23	0.40	0.07	20	2	0.38	0.34	0.00
Inp DCLAW MAN Rad 16	0.5	1	3	0.00	0.26	0.35	0.02	5	0.36	0.35	0.02	22	0.37	-0.01	0.39	0.02	23	25	7	0.63	0.23	0.41	0.07	25	2	0.38	0.34	0.00
Inp FDTW EUC Rad 1	0.2	25	2	0.65	0.42	0.40	0.00	5	0.37	0.39	0.07	2	0.38	0.03	0.39	0.02	13	15	7	0.85	0.40	0.59	0.23	3	6	0.31	0.40	0.02
Inp FDTW EUC Rad 2	0.6	1	2	0.00	0.46	0.34	0.01	5	0.36	0.35	0.01	25	0.39	0.05	0.39	0.02	13	15	6	0.83	0.54	0.48	0.11	25	6	0.31	0.39	0.01
Inp FDTW EUC Rad 4	0.2	25	2	0.67	0.42	0.39	0.00	7	0.24	0.36	0.01	25	0.39	0.05	0.41	0.04	23	25	8	0.69	0.33	0.45	0.10	25	6	0.29	0.39	0.01
Inp FDTW EUC Rad 8	**0.6**	**1**	**2**	**0.00**	**0.45**	**0.34**	**0.01**	**6**	**0.23**	**0.37**	**0.02**	**25**	**0.41**	**0.05**	**0.41**	**0.02**	**4**	**4**	**2**	**0.98**	**0.41**	**0.50**	**0.00**	**25**	**6**	**0.28**	**0.39**	**0.01**
Inp FDTW EUC Rad 16	0.6	3	2	0.01	0.42	0.34	0.01	3	0.37	0.36	0.01	25	0.38	0.06	0.39	0.01	14	15	2	0.92	0.64	0.36	0.07	25	6	0.27	0.39	0.01
Inp FDTW MAN Rad 1	0.2	25	2	0.65	0.42	0.40	0.00	15	0.13	0.39	0.03	23	0.37	0.04	0.37	0.01	19	20	2	0.97	0.40	0.60	0.46	25	6	0.31	0.39	0.01
Inp FDTW MAN Rad 2	0.2	25	2	0.65	0.42	0.37	0.00	5	0.33	0.36	0.02	25	0.37	0.08	0.41	0.03	14	15	2	0.93	0.60	0.46	0.18	5	6	0.30	0.37	0.01
Inp FDTW MAN Rad 4	0.2	25	2	0.65	0.40	0.37	0.00	15	0.16	0.40	0.04	1	0.35	0.00	0.37	0.01	19	20	2	0.98	0.48	0.75	0.80	25	6	0.28	0.39	0.01
Inp FDTW MAN Rad 8	0.5	4	2	0.03	0.36	0.35	0.01	7	0.22	0.37	0.01	25	0.39	0.06	0.39	0.01	19	20	2	0.98	0.48	0.75	0.34	25	6	0.27	0.39	0.01
Inp FDTW MAN Rad 16	0.5	3	2	0.03	0.36	0.35	0.01	4	0.33	0.37	0.04	25	0.38	0.02	0.39	0.01	19	20	2	0.97	0.48	0.50	0.17	25	6	0.26	0.40	0.01
SD DCLAW EUC Rad 1	0.1	25	2	0.61	0.75	0.39	0.00	9	0.64	0.37	0.02	18	0.39	-0.08	0.39	0.01	4	4	3	0.97	0.76	0.83	0.74	10	4	0.64	0.34	0.00
SD DCLAW EUC Rad 2	0.1	25	2	0.61	0.75	0.38	0.00	12	0.36	0.39	0.03	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	5	4	0.64	0.34	0.00
SD DCLAW EUC Rad 4	0.1	25	2	0.62	0.75	0.37	0.00	15	0.25	0.41	0.04	18	0.39	-0.08	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	10	4	0.64	0.34	0.00
SD DCLAW EUC Rad 8	0.1	25	2	0.62	0.75	0.37	0.00	10	0.50	0.39	0.02	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	5	4	0.64	0.34	0.00
SD DCLAW EUC Rad 16	0.1	25	2	0.62	0.75	0.37	0.00	8	0.46	0.36	0.01	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	20	4	0.64	0.34	0.00
SD DCLAW MAN Rad 1	0.1	25	2	0.61	0.75	0.39	0.00	13	0.41	0.40	0.04	18	0.39	-0.08	0.39	0.01	4	4	3	0.97	0.76	0.83	0.74	10	4	0.64	0.34	0.00
SD DCLAW MAN Rad 2	0.1	25	2	0.61	0.75	0.38	0.00	12	0.35	0.40	0.03	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	10	4	0.64	0.34	0.00
SD DCLAW MAN Rad 4	0.1	25	2	0.62	0.75	0.37	0.00	9	0.52	0.38	0.02	18	0.39	-0.08	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	10	4	0.64	0.34	0.00
SD DCLAW MAN Rad 8	0.1	25	2	0.62	0.75	0.37	0.00	8	0.53	0.38	0.02	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	20	4	0.64	0.34	0.00
SD DCLAW MAN Rad 16	0.1	25	2	0.62	0.75	0.37	0.00	10	0.42	0.39	0.02	18	0.39	-0.07	0.39	0.01	21	25	4	0.06	0.66	0.35	0.00	20	4	0.64	0.34	0.00
SD FDTW EUC Rad 1	0.1	25	2	0.54	0.84	0.35	0.00	12	0.38	0.40	0.03	23	0.42	0.00	0.42	0.02	14	20	5	0.09	0.66	0.36	0.00	10	4	0.61	0.34	0.00
SD FDTW EUC Rad 2	0.1	25	2	0.55	0.85	0.36	0.00	11	0.44	0.37	0.01	17	0.37	-0.01	0.37	0.00	21	25	8	0.18	0.66	0.37	0.02	5	4	0.63	0.34	0.00
SD FDTW EUC Rad 4	0.1	25	2	0.55	0.85	0.38	0.01	9	0.45	0.39	0.02	21	0.37	0.06	0.37	0.00	21	25	6	0.14	0.67	0.36	0.01	5	4	0.62	0.34	0.00
SD FDTW EUC Rad 8	0.1	25	2	0.55	0.85	0.38	0.01	11	0.38	0.42	0.04	23	0.39	0.04	0.39	0.01	4	4	3	0.97	0.75	0.67	0.52	10	4	0.62	0.34	0.00
SD FDTW EUC Rad 16	0.1	25	2	0.55	0.85	0.38	0.01	10	0.47	0.41	0.05	23	0.39	0.10	0.39	0.01	4	4	3	0.97	0.75	0.83	0.65	5	4	0.62	0.34	0.00
SD FDTW MAN Rad 1	**0.1**	**25**	**2**	**0.54**	**0.84**	**0.35**	**0.00**	**11**	**0.46**	**0.42**	**0.05**	**23**	**0.42**	**0.00**	**0.42**	**0.02**	**14**	**20**	**5**	**0.09**	**0.66**	**0.36**	**0.00**	**4**	**4**	**0.63**	**0.34**	**0.00**
SD FDTW MAN Rad 2	0.1	25	2	0.55	0.85	0.36	0.00	18	0.19	0.44	0.05	17	0.37	-0.01	0.37	0.00	21	25	8	0.18	0.66	0.37	0.02	15	4	0.63	0.34	0.00
SD FDTW MAN Rad 4	0.1	25	2	0.55	0.85	0.38	0.01	11	0.36	0.41	0.04	21	0.37	0.06	0.37	0.00	21	25	6	0.14	0.67	0.36	0.01	10	4	0.62	0.34	0.00
SD FDTW MAN Rad 8	0.1	25	2	0.55	0.85	0.38	0.01	11	0.51	0.40	0.03	23	0.39	0.04	0.39	0.01	4	4	3	0.97	0.75	0.67	0.52	20	4	0.62	0.34	0.00
SD FDTW MAN Rad 16	0.1	25	2	0.55	0.85	0.38	0.01	12	0.30	0.39	0.02	23	0.39	0.10	0.39	0.01	4	4	3	0.97	0.75	0.83	0.65	10	4	0.62	0.34	0.00

Table 1: Results in bold are for the representations of samples that produce the highest *k-NN* classification accuracy.

Lohrer, J., and Lienkamp, M. 2015. Building representative velocity profiles using fastdtw and spectral clustering. In *2015 14th International Conference on ITS Telecommunications (ITST)*, 45–49. IEEE.

McDonald, C. C.; Kandadai, V.; Loeb, H.; Seacrist, T. S.; Lee, Y.-C.; Winston, Z.; and Winston, F. K. 2015. Simulated driving assessment (sda) for teen drivers: results from a validation study. *Injury prevention* 21(3):145–152.

Ontañón, S.; Lee, Y.-C.; Snodgrass, S.; Winston, F. K.; and Gonzalez, A. J. 2017. Learning to predict driver behavior from observation. In *2017 AAAI Spring Symposium Series*.

Ratanamahatana, C. A., and Keogh, E. 2004. Making time-series classification more accurate using learned constraints. In *Proceedings of the 2004 SIAM international conference on data mining*, 11–22. SIAM.

Rosenberg, A., and Hirschberg, J. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 410–420.

Rousseeuw, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20:53–65.

Salvador, S., and Chan, P. 2007. Fastdtw: Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11(5):561–580.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *Departmental Papers (CIS)* 107.

Ten Holt, G. A.; Reinders, M. J.; and Hendriks, E. 2007. Multi-dimensional dynamic time warping for gesture recognition. In *Thirteenth annual conference of the Advanced School for Computing and Imaging*, volume 300, 1.

Walshe, E.; Oppenheimer, N.; and Winston, F. 2019. Comparison of virtual driving test performance and on-road examination for licensure performance: a replication study. In *Proceedings of the... international driving symposium on human factors in driver assessment, training and vehicle design*, volume 2019, 384–390. University of Iowa Public Policy Center.

Wong, J.; Hastings, L.; Negy, K.; Gonzalez, A. J.; Ontañón, S.; and Lee, Y.-C. 2018. Machine learning from observation to detect abnormal driving behavior in humans. In *The Thirty-First International Flairs Conference*.

Yasar, A.-U.-H.; Berbers, Y.; and Preuveneers, D. 2008. A computational analysis of driving variations on distributed multiuser driving simulators. In *IASTED International Conference on Modelling and Simulation*, number 19, 178–186. ACTA Press; Canada.