Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

April 2016

Advanced Driving Assistance Prediction Systems

Maedeh Hesabgar
The University of Western Ontario

Supervisor
Prof. Steven Beauchemin
The University of Western Ontario

Graduate Program in Computer Science

A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

© Maedeh Hesabgar 2016

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Artificial Intelligence and Robotics Commons, Databases and Information Systems Commons, and the Other Computer Sciences Commons

Recommended Citation
Hesabgar, Maedeh, "Advanced Driving Assistance Prediction Systems" (2016). *Electronic Thesis and Dissertation Repository*. 3695.
https://ir.lib.uwo.ca/etd/3695

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact tadam@uwo.ca.

Abstract

Future automobiles are going to experience a fundamental evolution by installing semiotic predictor driver assistance equipment. To meet these equipment, Continuous driving-behavioral data is observed and processed to construct powerful predictive driving assistance systems. In this thesis, we focus on raw driving-behavioral data and present a prediction method able to prognosticate the next driving-behavioral state. This method has been constructed based on the unsupervised double articulation analyzer method (DAA) which is able to segment continuous driving-behavioral data into a meaningful sequence of driving situations. Thereafter, our model by mining the driving data sequences situations can define and process the most influential data parameters. Our model can interpret the dynamic driving data and predict the next state of the determined vehicle by utilizing these parameters. This is a novel framework since the combination of main algorithms that we used differs from previous related works. Proficiency of this model has been evaluated with over three terabytes of driving behavioral data which include 16 drivers' data, for a total of more than 17 hours and over 456 Km.

Keywords

Driver Assistance, Driving Situations, Behavior Prediction, Double Articulation Analyzer (DAA)

Acknowledgments

I would like to express my sincere appreciation to my supervisor, Dr. Steven Beachemin, whose expertise, understanding, invaluable supervision and support made it possible for me to work on a topic that was of great interest to me. His technical and editorial advice was essential to the completion of this dissertation and has taught me innumerable lessons and insights on the workings of academic research in general.

Sincere thanks to the examining committee.

I owe infinite thanks to my lovely husband, Arash, for all the love, encouragement and continuous support he gave me. His support and encouragement was in the end what made this dissertation possible.

I am indebted to my family for their love and inspiration throughout my life. I am in particularly grateful beyond words to my mother, for guiding me in all difficult moments of my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

1 Introduction

Although motor vehicles have had a great influence on human life, they have always been a major cause of fatalities. Most vehicular crashes are the result of driver error. In the past decades, researchers have tried to devise safety systems to help drivers and even rectify driving errors, but still many deficiencies exist. There should be appropriate safety systems for recognizing a driver inattention and they should be capable of alerting drivers in hazardous situations. According to statistics, driver inattention or unintended maneuvers account for the highest percentage of deaths (80%) in the world (Angell, L. et al. 2006). This issue has motivated researchers to find solutions for coping with driving errors. Their main purpose is to develop safety systems which could help mitigate driver errors and hazardous driving situations (Brookhuis, K. A., & de Waard, D. 2010).

1.1 Problem statement

Driving is a difficult task because of the need to make correct decisions rapidly. Each decision a driver makes is important since it directly impacts traffic safety. Maneuvers involving changes in speed and steering wheel angle are the most influential factors concerning safety. Any abrupt change in speed or steering wheel can make the driving situation unsafe. Since each vehicle is surrounded by other vehicles, any inappropriate change in speed or steering wheel may have a cascading effect on vehicular safety.

In this thesis, the term “driver behavior” is used to signify the driver’s intent as it pertains to the most probable next maneuver. The goal of this thesis is listed as follows:

1- Understanding drivers’ behaviors; finding relationships between driving parameters such as speed, turn signals and steering wheel as inputs as necessary to assess a probability concerning the next maneuver.

2- Devising a powerful model for driving assistance systems able to predict the next driving maneuver before any specific or unintended maneuver begins. Most drivers are familiar with maneuvers such as passing, changing lanes to the left or right, starting, stopping, and turning left or right. In this thesis, the focus is placed on predicting a subset of canonical maneuvers such as speed and steering wheel angle.

1.2 Research Approach

In order to understand driver behavior, data recordings were needed to make important observations on drivers. Beauchemin et al. (2012) used car-mounted video cameras to capture surrounding forward traffic, the driver's head pose and gaze direction, and driving data from the vehicle's internal network. These data sources were recorded with 16 drivers over a pre-determined course inside the city of London. In total, 3TB of data was collected over more than 450 kilometers. In sum, stereo data from forward pointing cameras, vehicular attitude from the CANbus interface, and ocular movements from the drivers were recorded.

In this study, each driver was requested to drive about 28.5 kilometers and 60 minutes over a determined route. They were instructed to drive as they normally do. They did not have direct knowledge about the goals of the study

1.3 Thesis Organization

This thesis is organized as follows: Chapter 2 presents background information on advanced driving assistance systems (ADAS) necessary to understand concepts that are discussed in the next two subsections. The first subsection introduces the concept of driving maneuvers recognition and the second one provides the concept of driving maneuvers prediction frameworks. A comprehensive explanation of our research implementation found in Chapter 3. Chapter 4 discusses results and compares them with other methods. Chapter 5 presents a conclusion of the implemented methods and possible future work.

Chapter 2

2 Background and Literature Review

Since their introduction, Advanced Driver Assistance Systems (ADAS) remarkably reduced the number of transportation fatalities. ADAS improved vehicular safety by altering vehicular dynamics when necessary and by alerting drivers in dangerous situations.

Over the last decades, attempts have been conducted to model human driving behavior, in part to create more effective ADAS. To achieve this purpose, various driving assistance ideas and applications have been presented.

This Chapter provides a comprehensive report from papers pursuing the idea of understanding the behavior of drivers. It reviews the most important methods for driving maneuver recognition based on time-series data.

2.1. Driving Maneuvers Recognition

In recent years, several approaches for understanding and modeling complex behaviors such as driving have been proposed. Such behaviors may be considered as an ordered sequence of basic states happening over time. It is assumed that the basic states are the smallest “meaningful” units in the data sequence. Each basic state may be created during a variable period of time. For example, a change in steering wheel angle in order to change lanes takes shorter time in comparison to a right turn at an intersection. Basic states are related to each other with a logical relation; e.g. when a vehicle speed reduces continuously during a short time and it goes under 10 km/h, it is expected that the next state of the vehicle will be a full vehicle stop. Thus, there is a logical relation between speed reduction and stopping. To infer such logical relations between basic states, it is necessary to determine the set of possible transitions between them. In order to model basic states and model transitions, different frameworks such as the Dempster–Shafer framework, have been presented (Nigro, J. M., & Rombaut, M. 2003) (Hermann, A., & Desel, J. 2008).

Driving is a complex decision-making task. Drivers must understand the relations that exist between their vehicle and the environment. Based on this understanding, drivers make appropriate decisions and perform reliable changes in vehicle movement, such as stopping, turning right, changing lanes, etc. This paper uses the term “driving maneuvers” for this kind of changes in vehicle movement. Each driving maneuver is considered as a sequence of basic states. For example, when considering a “right turn” maneuver, there are some small units of meaningful operations such as “decreasing the speed”, “turn on the right signal”, “angle changes in steering wheel”, and “increasing the speed” that happen.

Each driver has an individual conception of the vehicle maneuvers and the surrounding environment. Also, each driver follows his/her individual driving style to perform driving maneuvers. Consequently, there is not any single model that can be specified. To consider these subjective components, probabilistic statements have been used to obtain a better understanding of maneuvers. For instance, Schneider et al. (2008) have presented a generic method for probabilistic identification of driving situations and maneuvers. This method separately models independent uncertainty situations. Bayesian networks and fuzzy features are applied in this method to model both the context and driving maneuvers. It is believed that using reliable driving assistance systems capable of learning and identifying drivers’ patterns, could be very helpful in completing a range of different tasks, particularly decision-making.

Hülnhagen et al. (2010) proposed a maneuver recognition method based on a Bayes filter algorithm. This method has a straightforward design. It combines probabilistic finite-state machines (Vidal, E. et al. 2005) with a fuzzy rule. To form a specific driving maneuver, probabilistic finite-state machines construct all possible sequences of basic maneuver states. Then, the method uses the fuzzy rule for modeling basic states. In other words, the fuzzy rule can model a sequence of basic states related to a specific driving maneuver.

For instance, the basic states of an “overtaking” maneuver after decomposition are as follows: approach leading vehicle, lane change to the left, passing, and finally lane change to the right.

In their approach, Hülnhagen et al. (2010) listed main driving variables such as velocity, acceleration, steering wheel, and indicator status. Then, they assigned an optional number of self-styled linguistic terms to each variable. For example, linguistic terms that they have assigned to the velocity variable are: none, very slow, slow, fast. Thereafter, they considered all basic elements and for each one computed the amount of membership each linguistic term has had. Finally, Bayes filters have been used to evolve which state of a driving maneuver should be chosen according to an assessment of the highest probability.

There are some advantages in Hülnhagen et al.’s (2010) proposed method. First, compared to neural networks, this method can explain decisions more easily. Second, it is flexible and easily expandable with additional basic elements; i.e. by adding more basic elements to the current model, the system can be developed without having any effect on other existing elements and maneuver models. Third, this method looks very appropriate to ADAS since it has low computational complexity. Finally, it is a robust method to recognize turn maneuvers and distinguish them from similar maneuvers.

2.2. Time Series and Driving Maneuver Prediction

In recent years, driving behavior prediction has become one of the most important challenges in ADAS development. Much work has been conducted to achieve reliable driving predictions. A reliable prediction model must have a set of all possible driving maneuvers such that it can correctly find the characteristics of the next maneuver.

In predictive systems, data captured in real time is usually analyzed immediately. In this thesis, captured data is considered as a time-series and analyzed after its collection.

In order to achieve a predictive model, we first have to find a model that can be adapted to time-series driving data. A good number of methods have been presented to model driving time series data. Among these, we review those shown to be best-in-class.

In order to predict driving maneuvers, Hidden Markov Models (HMM) (Rabiner, L. R., & Juang, B. H. 1986) have often been used to extract information from time-series data (Takano, W. et al. 2008) (Gales, M., & Young, S. 2008) (Mitrović, D. 2005). HMMs model hidden discrete states as x and observations as y. HMMs treat every observation as a statistically independent entity. A next generation approach developed and named AutoRegressive Hidden Markov Models (ARHMM) allow some stochastic dependencies to exist between observations, as shown in Figure 1. The current observation is dependent to a past observation, as there is a correlation between the two. For this reason, ARHMM can be used with dynamic behaviors since it is powerful enough to model feedback systems (Kishimoto, Y., & Oguri, K. 2008) (Stanculescu, I. et al. 2014).

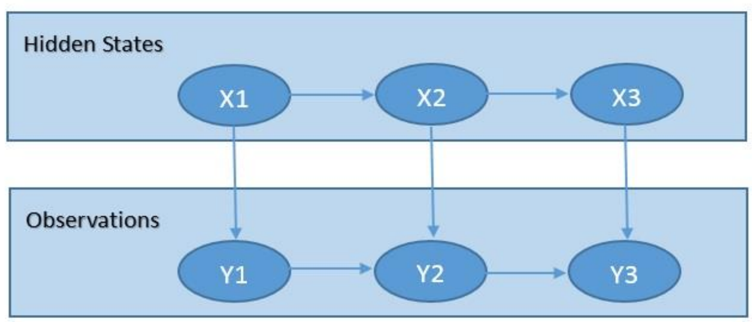


Figure 1: Auto Regressive Hidden Markov Model. The hidden state X3 is not visible to the observer, but we can say that there is a dependency between hidden states (X3 and X2) because of the existing dependency between observations (Y2 and Y3).

Both HMM and ARHMM encounter serious problems because they need to have a fixed number of states *a priori*. In order to address these problems, a novel and efficient method, called Beta Process AutoRegressive Hidden Markov Model (BP-AR-HMM) was proposed by Fox et al. (2009). This is a robust model that can simultaneously handle multiple related time-series data and automatically determine the number of states. Hamada et al. (2013) applied a BP-AR-HMM model to their driving dataset. Three different driving operations (gas pedal opening rate, brake pressure, and steering wheel angle) were considered. Four estimated state sequences were obtained. In each state, operations had different behaviors from the other states. Hamada et al. (2013) obtained good results. For example, they could find out which specific state is often followed by another state. According to their results, there were specific times at which, while brake pressure was increasing, the steering wheel angle was approximately zero, and after some time, the brake pressure was decreasing and steering wheel angle was increasing. These changes happened to create a specific latent state named “turning left”. As it seems, there are some specific driving patterns derived from multiple time-series dataset in this research. It clearly shows which states follow the previous ones. Later on, Hamada et al. (2013) used derived patterns to predict driving operations. Their method could predict sudden decreases of brake pressure that were just happening before left turns. This is a successful prediction approach since it can predict deceleration in a reasonable time before it happens (Hamada, R. et al. 2013).

2.2.1 Long-term Contextual Prediction

The capability to predict driver intent is an essential aspect of ADAS. Previous methods attempted to model and predict driving behavior within short time scales of their multiple time series-data set. To model long-term contextual information, the Double Articulation Analyzer (DAA) was proposed by Taniguchi et al. (2012). They suggested that contextual information and human driving behavior possess a double articulation structure.

The term “double articulation structure” was first presented in order to analyze a speech stream. A speech stream data possesses a dual layer of information that can be decomposed into several meaningful linguistic units, and each unit can be divided into meaningless elements. Meaningless elements called *phonemes* are at the lowest level of speech organization. Morphology, syntax, and semantics give the meaning to phonemes and they are the higher levels of speech organization.

In order to understand long-term human action, it has to be decomposed into short-term chunks. To extract long-term human action chunks, Taniguchi, T., and Nagasaka, S. (2011) presented a DDA framework that included both a language model; Nested Pitman-Yor (NPYLM), and a stochastic model; sticky Hierarchical Dirichlet Process Hidden Markov Model (sHDP-HMM). Sticky HDP-HMM is an augmented version of HDP-HMM (Fox, E. B. et al. 2007 in which the number of states is not predefined. Figure 2 shows the graphical representation of sticky HDP-HMM, which is an improvement over normal HDP-HMM. The main difference between HDP-HMM and sticky HDP-HMM is that HDP-HMM tends to give high posterior probability to states with rapid switching while sticky HDP-HMM fixes this problem. Parameter k in Figure 2 is the transition weight which is responsible for controlling rapid switching. If we set k = 0, sticky HDP-HMM algorithm behaves same as normal HDP-HMM.

DAA assumes that human action is a continuous series of smallest “meaningless” data time-series, and the smallest “meaningful” units as sequences of the meaningless elements. This structure can find and connect several short-term segments of human action. If we look at our spoken language, it also has a double articulation structure. For example, a sentence can be decomposed into single letters. Then single letters can be chunked into words. Letters individually do not have any meaning, however words do. Human action time-series have the same pattern. It is obvious that at each time, point data do not have any meaning individually, but when some of them come together, they form a meaningful segment. By expanding this idea, several successive segments of time series data form a meaningful sequence of a specific human action. A meaningful sequence of a specific human action is assumed to be a word. Taniguchi, T., and Nagasaka, S. (2011) method can extract unit actions by using NPYLM.

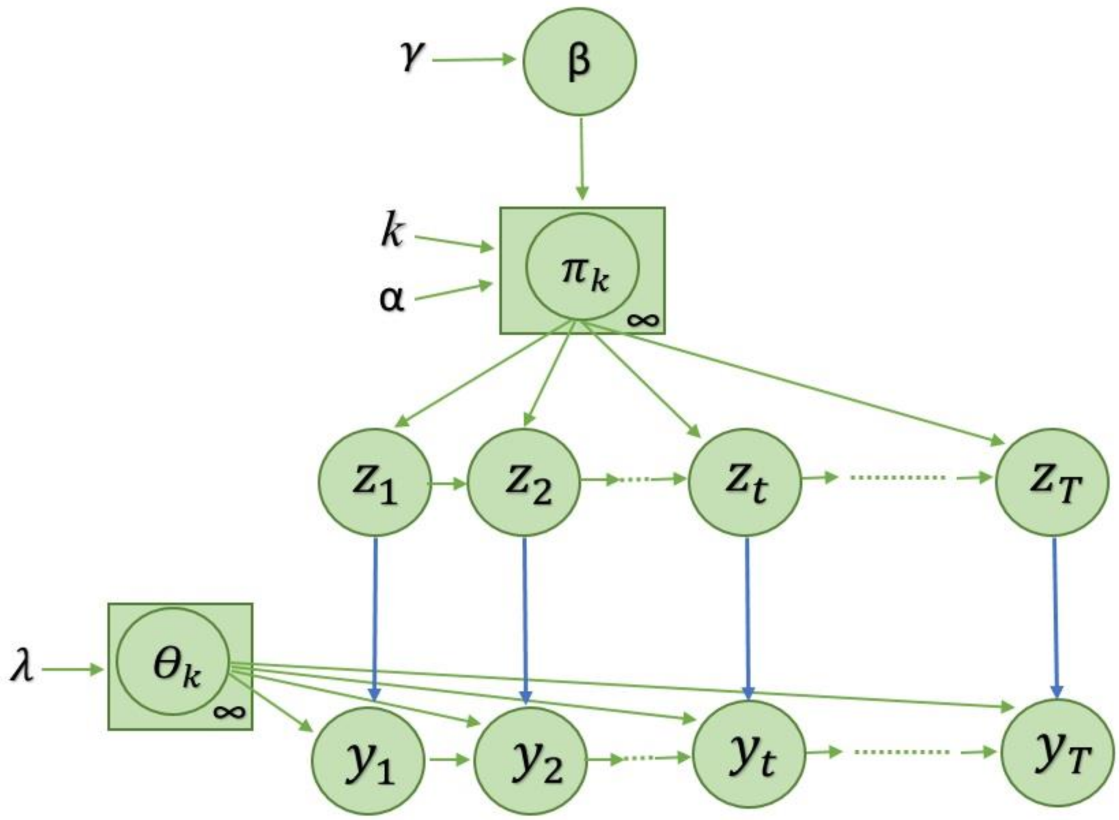


Figure 2: sticky HDP-HMM Graph. The variables have been introduced by Fox et al. (2009) as follows: y_t is extracted feature vector at time *t* and z_t is its related state label. *K* is an infinite transition bias. Original HDP-HMM, *k = 0.*

Both sticky HDP-HMM and NPYLM algorithms use a *Gibbs sampler* in their structure. A Gibbs sampler is used to approximate the joint distribution when there are difficulties with direct sampling and the joint distribution is unknown. The joint distribution is unknown because the full conditional distribution for each parameter (gas, brake, etc.) is also unknown. The full conditional distribution for each parameter (α_j) is $P(\alpha_j \mid \alpha_{-j,y})$ ($\alpha_{-j,y}$ is all parameters except α_j).

A Gibbs sampler is a Markov Chain Monte Carlo (MCMC) algorithm which generates a sequence of observations; observations are from a specified joint probability distribution.

A Gibbs sampler produces a Markov chain of nearby, correlated samples. For each iteration of this algorithm, an instance from the distribution of each variable one after the other is being generated. Finally, after convergence is achieved a desirable result is obtained.

In 2012, Taniguchi et al. (2012) also used sticky Hierarchical Dirichlet Process HMM (HDP-HMM) and Nested Pitman-Yor Language Model (NPYLM) together to develop the idea of a double articulation analyzer. They used sticky HDP-HMM as a prelude to converting time series data into sentences. Particularly, it has been used to obtain sequences of letters (hidden states).

Mochihashi, D. et al. 2009 used infinite HMM for the purpose of flexibly evaluating the number of hidden states based on the given training data. To extract words from sentences, an unsupervised morphological analysis method was employed. They have extended the unsupervised morphological analysis method to work on incomplete sentences by parsing incoming time -series data. Mochihashi et al.'s (2009) method is also based on Nested Pitman-Yor Language Model (NPYLM), which consists of two Hierarchical Pitman-Yor (HPY) processes; a language model and a word model. The language model, which is named NPYLM, enables the system to have an unsupervised chunking. Figure. 3 shows a graphical representation of the NPYLM model. NPYLM takes a text as input. The text must be a set of successive letters separated by spaces. The length of the text has an important effect on the final result in that we should make sure that the selected letter sequence is large enough to represent each letter frequency properly.

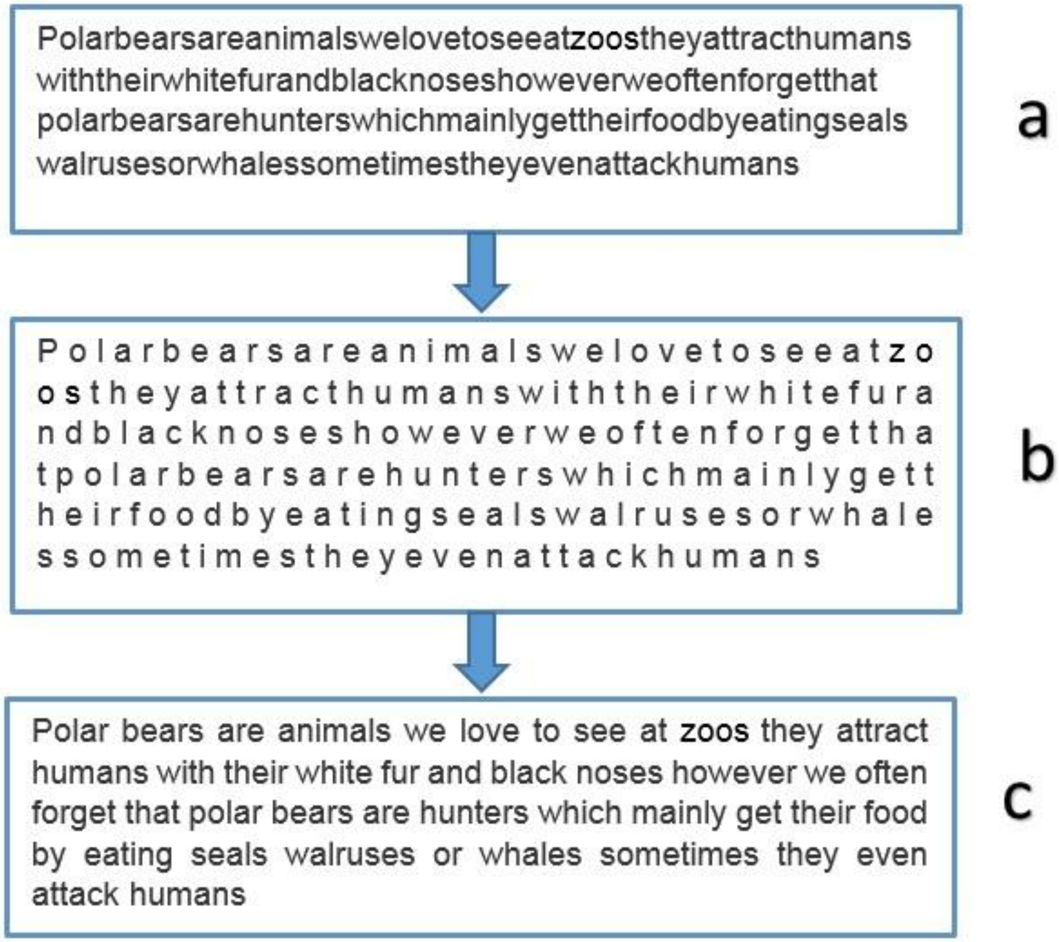


Figure 3: An example of the NPYLM procedure. (a) shows the coarse text which is a mixture of different letters, and hard to read. (b) is a preprocessing level, which is preparing the coarse text for main processing. (c) is a readable text which is the result of applying the NPYLM process.

Sticky HDP-HMM and NPYLM are able to extract meaningful chunks from the continuous time series-data. Compared to conventional HMM and simple NPYLM, Taniguchi et al.’s (2012) method is a better approach as it considers the incompleteness of observed sentences and improves the long-term prediction performance.

Taniguchi et al. (2012) first applied their method to a sentence data model. Their predictor method assumed that a number of words and states are unknown. In order to have a trained generative language model, a word model had to be trained. Thus, 100 artificial sentences consisting of 50 words each were generated. Then, for evaluating prediction performance, they used one of the sentences as test data and the rest of the sentences for training. Following this, they changed the test sentence by omitting the last parts of the sentence. The proposed method predicted the erased letters from the incomplete sentence. They then compared the results with other methods such as conventional Markov Models and simple NPYLM. The results show that NPYLM with prediction achieved better results. In particular, it could predict a much longer sequence of letters so many times.

In the next experiment, they applied their method to a real-world driving data model. To perform the experiment, vehicle data such as velocity, steering angle, brake pressure, and accelerator position, were collected. They also added two more features to the time-series driving behavior data; one of them was the temporal difference of velocity and the other one was the temporal difference of steering angle. Then, they applied sHDP-HMM to the acquired data and the unsupervised double articulation analyzer was used to segment and chunk the sequence of driving on the roadway. Following this, the same training and testing procedure as in the first experiment was performed on the data to evaluate the method. Results show that NPYLM with prediction outperformed conventional Markov Models and simple NPYLM.

Although DAA is an efficient segmentation method, it has some fundamental drawbacks. First, it is not able to perform segmentation and chunking simultaneously. Second, since it is a fully unsupervised method, it is not able to generate labels for the extracted driving words. Thus, it does not have any label to inform a driver. Third, it just concentrates on one estimation result and discards other possible predicted scenarios. Fourth, the duration of a chunk, which is related to a driving word, is not considered in this model. Fifth, the DAA model extracts too many different kinds of words: it can extract more than 400 kinds of driving words for 90 minutes of driving.

In 2013, Bando et al. (2013) proposed a new framework that can automatically translate driving data into sequences of “drive topics” in natural language. In this framework, brake pressure, throttle opening rate, steering wheel angle, and velocity are considered as physical features. Each of these had different frequencies for each word. They used Latent Dirichlet Allocation (LDA) to cluster extracted driving situations based on the existing frequency of physical behavioral features that are observed in each driving sequence. The distribution of the physical behavioral features included in each drive topic was used for automatic driving word labeling. The result of this step is a small number of drive topics, such as “accelerating” and “high speed”, assigned to each driving words. Although DAA and LDA are completely unsupervised methods, this framework creates human-understandable tags. Being independent of any human-created tags is one of the greatest benefits of this method.

Bando et al. (2013) used a multimodal latent topic model for estimating multimodal drive topics. Multimodal LDA (mLDA) has been proposed for annotating images by Wang et al. (2009). The probabilistic distribution of driving behavioral features, image features, and human annotated tags are used for estimating multimodal drive topics. Throttle opening rate, brake master-cylinder pressure, angle of the steering wheel, and vehicle velocity, and their different values form an eight-dimensional feature space for driving behavior. K-means has been used for clustering behavioral features in the eight-dimensional feature space. They tested data with two different values of K ($K = 10$ and $K = 100$). Prediction results coming from $K = 10$ topics were different from $K = 100$. In both cases, the experiment showed that a multimodal latent topic model is successful in that it is close to the performance of human annotators.

In 2014, Taniguchi et al. (2015) provided another contribution towards improving long-term driving behavior prediction and rectifying earlier problems. They proposed an unsupervised learning method, named Double Articulation Analyzer with Temporal Prediction (DAA-TP), which can model the duration of each driving behavior chunk (and in particular the *remaining duration* of a driving-behavior chunk). This is obtainable only under the assumption that driving behavior data has a doubly articulated structure. They used a Hierarchical Dirichlet Process Hidden Semi-Markov Model (HDP-HSMM) (Johnson, M. J., & Willsky, A. S. 2013) to enable the DAA to model the duration of each hidden state along with the ability of automatically estimating the number of hidden states. After using HDP-HSMM for estimation, they compressed subsequences of identical states into individual letters (from "aaaiirrrr" to "air", for instance) and used a Bayesian unsupervised morphological Analyzer (Mochihashi, D. et al. 2009) for chunking driving letters sequences into word sequences. According to this assumption, driving behaviors such as "turning right" and "going forward" can be considered as a chunk in a sentence.

This proposed method determines all possible latent words that can come after the specific chunk and complete the rest of a specific sentence. It also can identify the existing probabilistic transition rules between words. Since there are many possibilities of latent driving words and letters for completing a specific sentence, the remaining duration of the current driving word is not easily predictable and it is necessary for this model to take into consideration all the possible latent driving words.

In their experiments, they evaluated DAA-TP's prediction performance in finding the correct position of the next changing point of chunks. Since there was no any other competing comparative unsupervised learning method, they used linear regression and a recurrent neural network (RNN) as two conventional supervised methods to compare their technique with. The probability distribution of the estimated remaining chunk duration was calculated for each frame of observation data by these three methods. Velocity, steering angle, brake pressure, accelerator, temporal differences of both velocity and steering angle were used as input vectors for all three methods. Their results indicate DAA-TP is more accurate than RNN and linear regression in predicting the next termination time of the current chunk of driving-behavior data.

Chapter 3

3 Proposed Method and Its Implementation

The purpose of this study is to present an efficient optimized prediction method for advanced driving assistance systems (ADAS). In order to create an appropriate ADAS, we need a comprehensive knowledge of the surrounding traffic and driver behavior. These are obtained from the CANbus interface, faceLAB eye tracker, and the roadLAB frontal stereo-vision system. In this thesis, we only used the data coming from the CANbus interface of the vehicle. This interface recorded any change that occurred in gas pedal pressure, brake pedal pressure, steering wheel angle, blinker status, and the speed of the vehicle. The recording sampling rate was set to 15Hz. The CANbus interface recorded data from a total of 16 drivers.

We were inspired by the natural language processing models and considered two powerful methods to obtain a better understanding of the recorded data, namely: sticky HDP-HMM (Fox, E. B. 2011) and NPYLM (Mochihashi, D. et al. 2009). These two methods together make a robust model called a Double Articulation Analyzer (DAA) (Taniguchi, T., & Nagasaka, S. 2011).

Sticky HDP-HMM is an extension of Hidden Markov Models. This model accepts a sequence of data and assigns a label to each data frame, based on the some characteristics of the current along with those of its neighbors. HDP-HMM has been used successfully in various contexts such as music synthesis (Hoffman et al., 2008), visual scene recognition (Kivinen et al., 2007), and gene expression (Beal and Krishnamurthy, 2012). In addition, this model is completely independent from any human tags, which fits our problem requirements. Therefore, we decided to choose sticky HDP-HMM to label our data frames (sticky HDP-HMM is an augmented version of HDP-HMM that assign labels more accurately). We considered five values for each frame (gas pedal pressure, brake pedal pressure, steering wheel angle, navigators' status, and the car speed). If we look at each label individually, it is a label for only $\frac{1}{15}$ of a second and it is too short to convey any meaning; thus we use the term "letter" for sticky HDP-HMM generated labels.

NPYLM is a parsing method for natural language processing. It can parse a sequence of letters into words. These words do not belong to any predefined dictionary. I.e. NPYLM is completely independent from any predefined dictionary; it parses letters to unknown words based on letter frequencies. We look at the output of sticky HDP-HMM as a sequence of letters and use NPYLM to extract words from it. Each word refers to a time interval (a sequence of frames) of driving maneuver, which is meaningful in comparison to letters.

DAA is a two-layer method that converts a stream of speech into meaningless elements (we call them letters) (first layer) and extracts meaningful signs (we call them words) from those elements (second layer). In this thesis, we considered driving behaviors data as a stream of speech. And, we created a sequence of meaningful signs from our driving data to be able to predict the future driving maneuver based on the previous driving maneuvers. We used sticky HDP-HPP as the first layer of DAA, and NPYLM as the second layer.

In this research, we applied DAA to the CANbus dataset. In order to use this data for the purpose of driving behavior prediction, we presented a novel framework. It is novel because we used a different combination of main algorithms which differs from the combination that Taniguchi et al. (2012) used. Also, the database that we used is completely new. In addition, the chosen algorithms in our framework are completely unsupervised. As the result, it builds its own letters and words such that a pre-existing dictionary is not necessary. This framework is able to make a use of DAA to predict the next driving maneuver that has the most probability of occurring. We then explain the model procedure in detail. Figure. 4 briefly shows the steps involved in DAA.

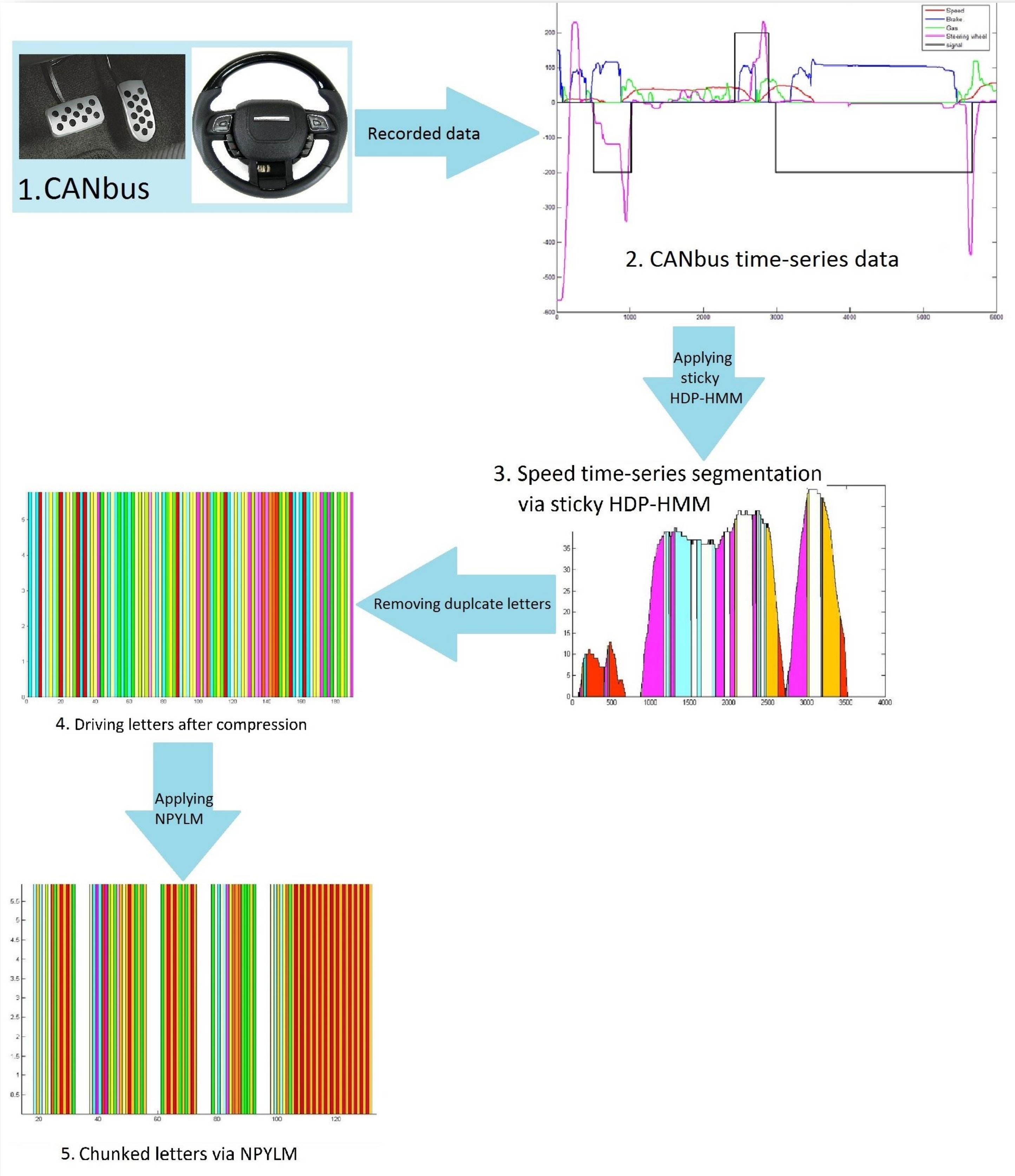

1.CANbus
Recorded data
2. CANbus time-series data
Speed
Brake
Gas
Steering wheel
signal
Applying
sticky
HDP-HMM
3. Speed time-series segmentation
via sticky HDP-HMM
Removing duplcate letters
4. Driving letters after compression
Applying
NPYLM
5. Chunked letters via NPYLM

Figure 4: Double Articulation Analyzer Steps. 1. CANbus interface of vehicle. 2. This picture shows the recorded time-series data (gas, brake, speed, steering wheel, and turn signals) coming from CANbus interface. 3. This graph represents the sequence of letters for the speed time-series which have been generated after we applied sticky HDP-HMM (different colors represent different letters).4. This picture shows the sequence of compressed letters. In this step, we have eliminate the letters are located beside each other and are duplicated; as an example, "tttthhiiiiisss" compresses to "this" (different colors represent different letters). 5. This graph represents the sequence of words that have been produced by applying NPYLM to the sequence of compressed letters (white space between colors are borders between words).

3.1 Semiotic segmentation using a Double Articulation Analyzer (DAA)

Using DAA to extract meaning from a massive sequence of raw data is the first and most important part of our research. To achieve this goal, we trained our model using the driving data collected from 15 drivers. The time each driver spent driving was approximately one hour. As a result, the total time of the CANbus data represents 15 hours of driving. Figure 5 shows the determined path each subject drove. The path is located in London, Ontario, Canada and it is highlighted in the map. In addition, Table 1 presents the collected driving behavioral data.

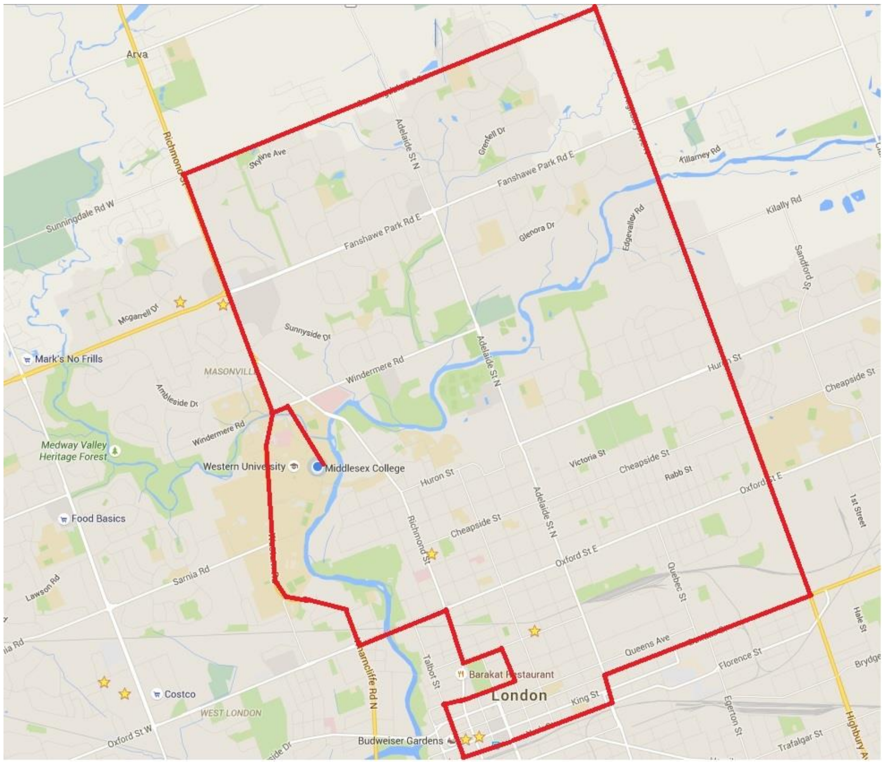


Figure 5: Pre-determined driving path on Google map.

Table 1: CANbus data. Training with 15 subjects to test prediction on 1 subject distinct from the 15 training sets. Sampling rate is 15 frames per second for both training and test sets.

	Training	Test
Sampling rate	15 Hz	15 Hz
Number of subjects	15	1
Total distance	427.5 km	28.5 km
Total data length	12.9822 hours	1.1236 hours
Mean velocity	39.3579 km/h	32.1254 km/h

3.1.1 Data selection

As an initial step, we need to determine what type of data we want to work on. In this research, the CANbus network data recorded speed, brake and accelerator actuator pressure, steering wheel rotation, and turn signals. By understanding the relationships between these elements (if they exist), predicting the most probable next driver maneuver may be possible.

Three different combinations of these data items have been selected and tested in this research. First, speed, brake, and gas were selected. Following this, we added steering wheel to observe the result of the new combination and compare it with previous results. Finally, a new parameter, turn signals, was added to find out whether it can improve the model result or not. In the next two sections, the implementation of DAA for our model is explained in detail.

3.1.2 Segmentation via sticky HDP-HMM

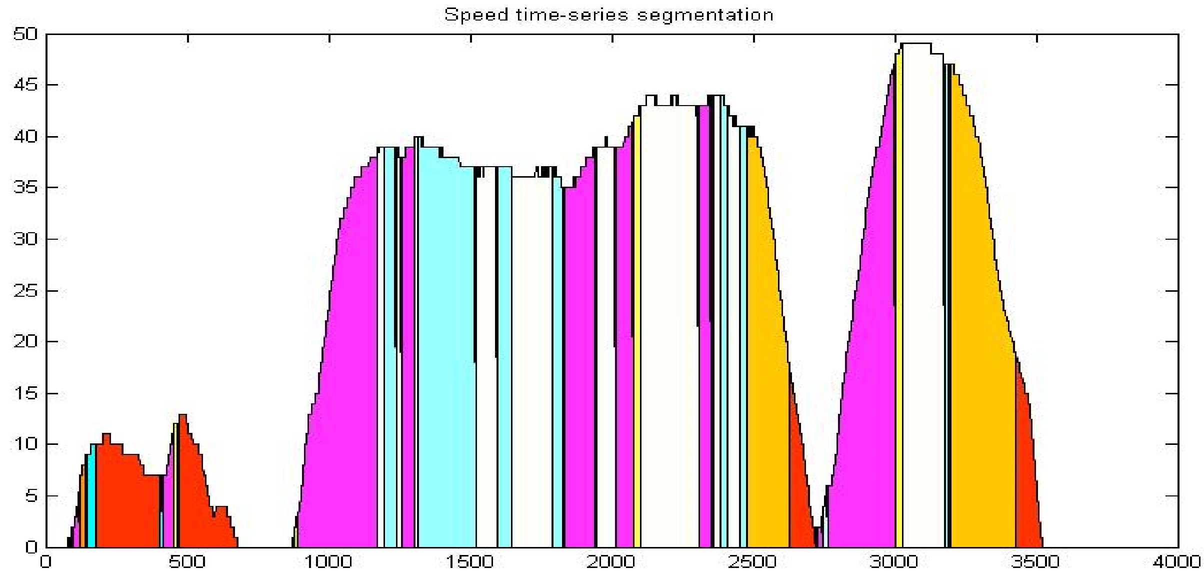
Once the raw data is selected, sticky HDP-HMM is used to find meaningful patterns within the time-series data. This algorithm accepts multiple time series data as input. For instance, speed values, brake and gas pressure values over the time can be the input of sticky HDP-HMM algorithm. Sticky HDP-HMM segments the driving data into a sequence of driving behavioral primitives (letters). It is possible to determine the number of unique driving primitives, as we can set the number of unique letters as a parameter in sticky HDP-HMM. In this study, the number of unique letters is set to $M = 25$ and $M = 50$. We run this algorithm with different M values. We noticed that the range between 25 and 50 generates most accurate results because choosing a number larger than 50 leads to having different states for similar driving primitives. On the other hand, choosing a value less than 25 leads to having same states for dissimilar driving primitives. Sticky HDP-HMM is an unsupervised method that works simultaneously on different numbers of time-series and produces one driving primitive (letter) for each frame. Sticky HDP-HMM uses a blocked Gibbs sampler (Fox, E. B. et al. 2007) to find joint changing points within the time-series. Then, Sticky HDP-HMM considers joint changing points over time-series, and extracts the sequence of driving letters. Speed, brake, gas, steering wheel, and turn signals are observation states for our implementation of sticky HDP-HMM and driving letters are the hidden states. Driving letters are the output of sticky HDP-HMM. I.e. sticky HDP-HMM assign a letter to each data frame.

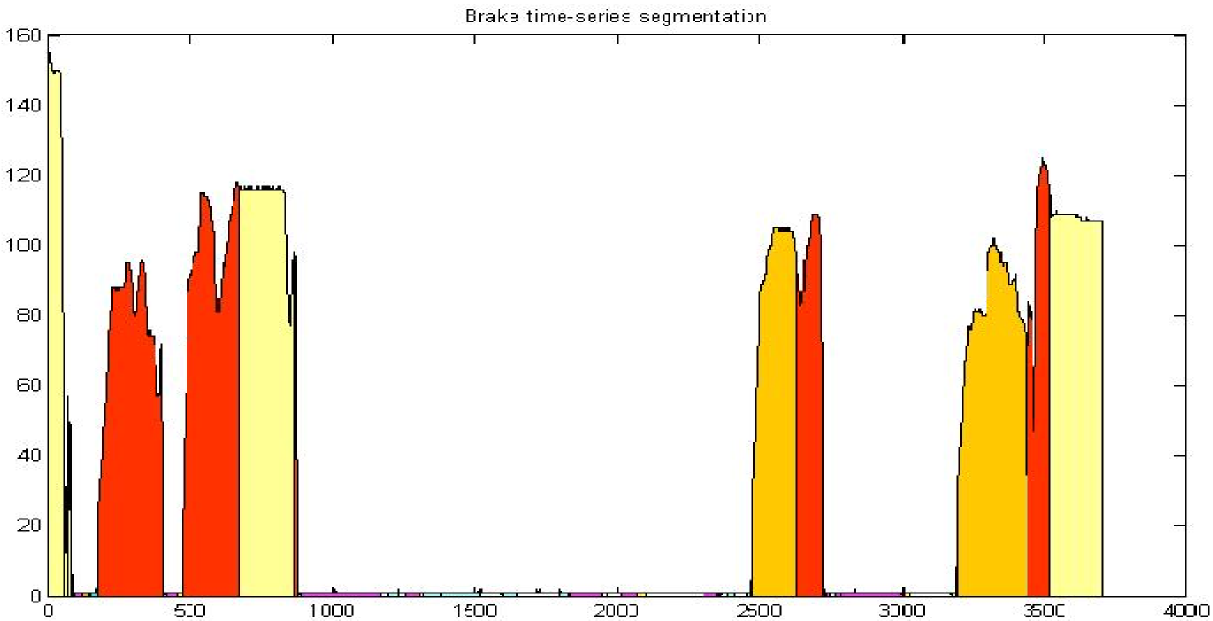
Figure 6 represents the sticky HDP-HMM result when we used just speed, brake and gas time-series as observations. In this figure, the first, second, and third graph are from the speed, brake, and gas time series, respectively. Colors represent driving letters (the output of sticky HDP-HMM). It is noticeable that the sequence of letters for all three graphs is the same. That is, sticky HDP-HMM assigns a letter to each frame considering all time-series together. For example, from time 500 to time 750, the three graphs all show the color red.

Figure 7 is similar to Figure 6, but it represents the sticky HDP-HMM results when we used speed, brake, gas, and steering wheel together.

Figure 8 also displays the sticky HDP-HMM results when speed, brake, gas, steering wheel and signals were used. There is an interesting story behind these three figures; the results of sticky HDP-HMM shows that the characteristics of an individual letter do not change over different time-series.

By comparing these three figures, it is observed that the results in Figure 8 are more precise in segmentation since the characteristics of the same color letters are matched better than the two other results. It means that adding steering wheel and turn signals data has a positive effect on the results of the segmentation.


Speed time-series segmentation
0 5 10 15 20 25 30 35 40 45 50
0 500 1000 1500 2000 2500 3000 3500 4000


Brake time-series segmentation
0 20 40 60 80 100 120 140 160
0 500 1000 1500 2000 2500 3000 3500 4000

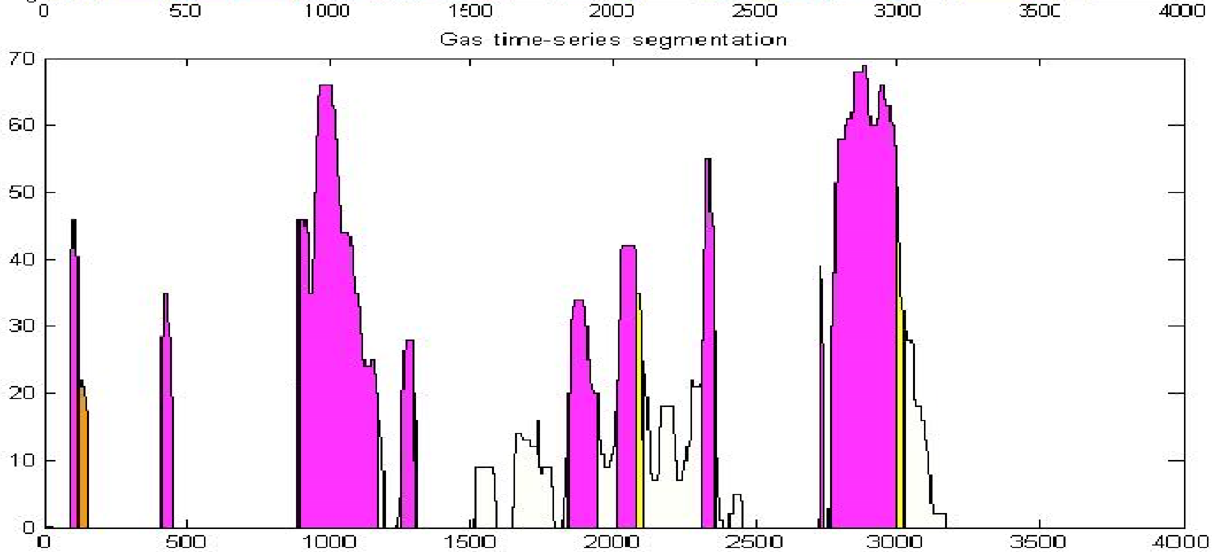

Gas time-series segmentation
0 10 20 30 40 50 60 70
0 500 1000 1500 2000 2500 3000 3500 4000

Figure 6: Sticky HDP-HMM results using speed, brake, and gas. Each color represents a unique driving letter. Colors in the 3 graphs at each point of time are the same, because sticky HDP-HMM generates the result based on all time series together.

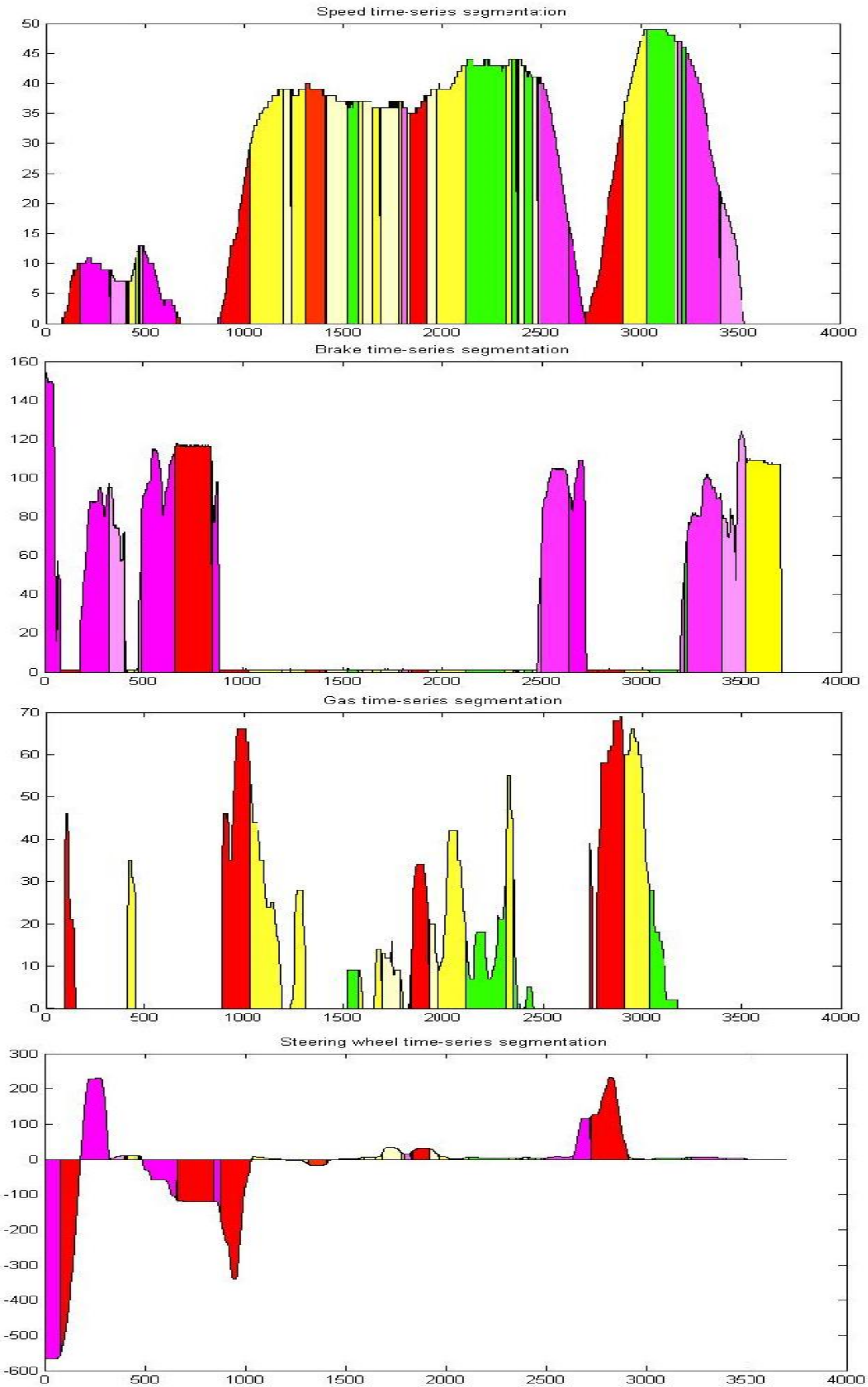

Speed time-series segmentation
Brake time-series segmentation
Gas time-series segmentation
Steering wheel time-series segmentation

Figure 7: Sticky HDP-HMM results using speed, brake, gas, and Steering wheel. Each color represents a unique driving letter. Colors in the 4 graphs at each point of time are the same, because sticky HDP-HMM generates the result based on all time series together.

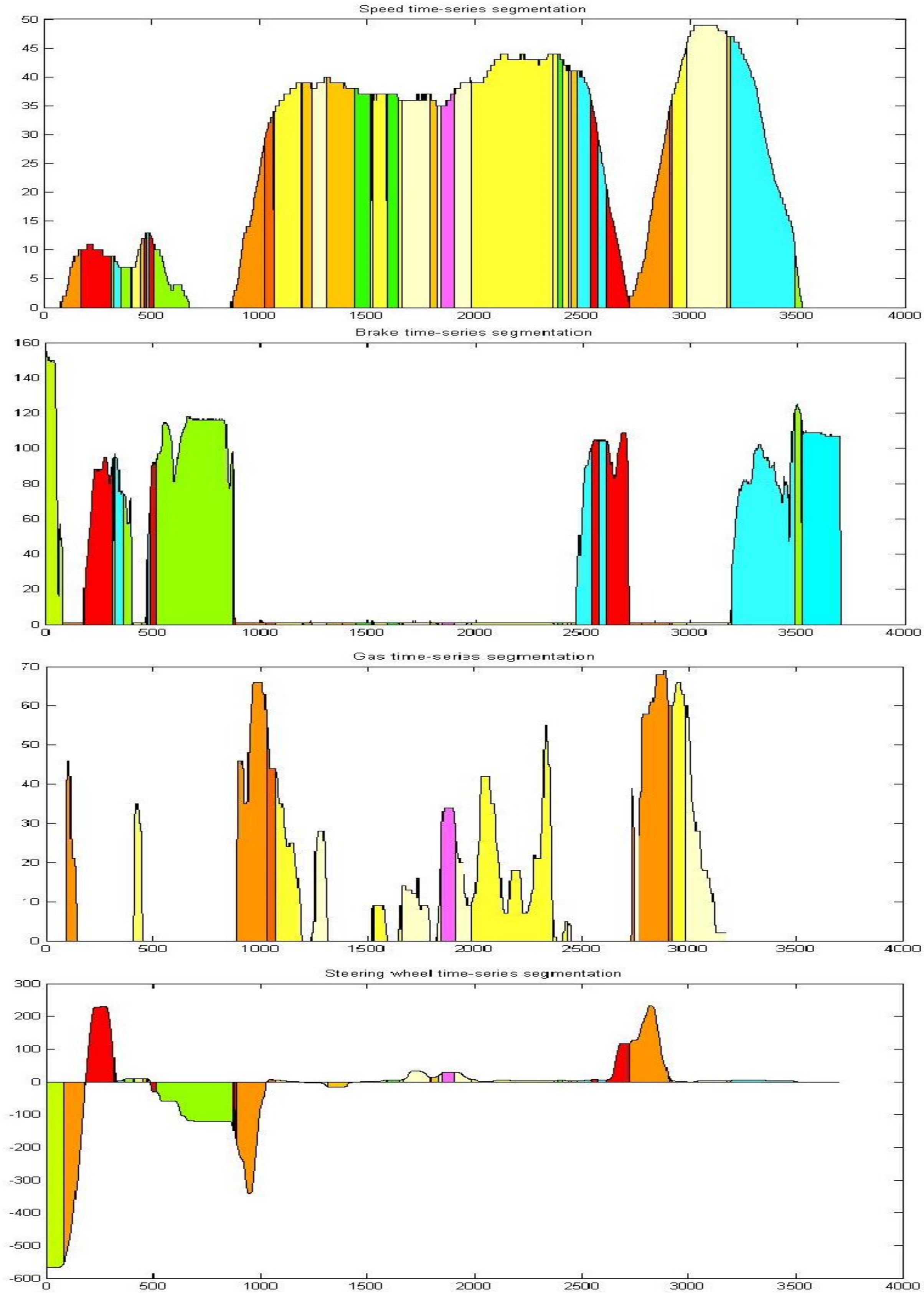

Speed time-series segmentation
Brake time-series segmentation
Gas time-series segmentation
Steering wheel time-series segmentation

Figure 8: Sticky HDP-HMM results using speed, brake, gas, Steering wheel, and signals. Each color represents a unique driving letter. Colors in the 4 graphs at each point of time are the same, because sticky HDP-HMM generates the result based on all time series together.

3.1.3 Chunking via NPYLM

After sticky HDP-HMM gives the sequence of letters, we preprocess it to be prepared for NPYLM. I.e. we compress the duplicate letters from the same state to prevent confusion and false (invalid) results. For example, “tttthhiiiiisss“ would be replaced with “this“.

NPYLM was used to parse sequence of letters (sticky HDP-HMM results after the preprocessing step) into words. Without assuming any predefined dictionary and by using a Gibbs sampler, NPYLM extracts words from the sequence of letters. This process happens based on the letters frequency. In this research, NPYLM accepts the compressed sequence of letters as input and extracts more than 345000 non-unique words from 15 hours of driving. The output of NPYLM is a sequence of words that belongs to all time series that we have. I.e. our multiple time series data now can be considered as a sequence of words. Each word is a sequence of letters. In contradiction to letters, these words represent a small chunk of the driver’s behavior that have meaning. Figure 9 shows speed, gas, and brake time series values. In this figure, each color belongs to a specific letter. Picture. 1 at this figure represents the output of sticky HDP-HMM. Picture. 2 shows the sequence of words. There is a white space between words. As an example, the first word has three letter that the first and third letter are the same. This word convey some meanings. For instance, picture.1 time intervals indicates that speed is increasing, brake pressure is decreasing, and gas pressure is increasing.

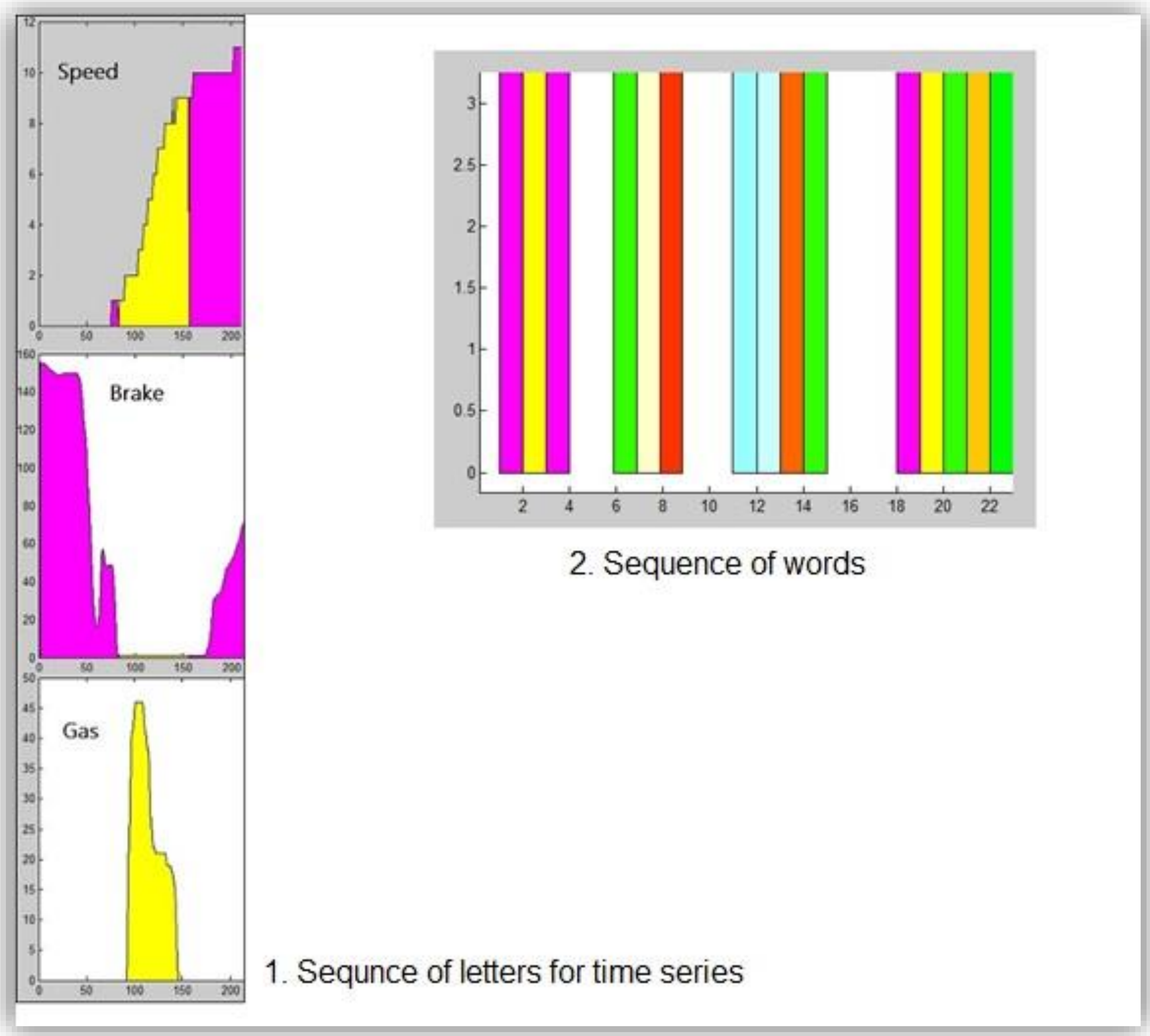


Figure 9 : NPYLM result. Picture 1. represents sticky HDP-HMM result. Each color belongs to a specific letter. Picture 2. shows the result of NPYLM. It is a sequence of words. First word belongs to Picture .1. I.e. purple, yellow, purple is a word. The meaning of this word is: speed is increasing.

3.2 Finding patterns for predicting driving behaviors using DAA

Around 450 unique words have been extracted from the CANbus data set. This is a large amount of data that is not easily understandable. I.e. extracted words are not applicable enough for drivers to understand intuitively. Therefore, in this study we want to present a novel applicable structure which is able to find common patterns among the training words data set.

Extracted word boundaries have the most important role in finding common features, because they are actually the contextual changing points of time-series data. Such a large amount of words does not mean large word diversity. Indeed, by analyzing driving behavioral words, it is possible to recognize patterns among words. What follows, are describe the steps involved in clustering driving behavioral words, constructing a distribution table, and predicting driver behavior.

3.2.1 Driving behavioral feature extraction

In this research, we determined the most influential parameters for analyzing the words. These parameters were calculated for each word and are referred to as “driving behavioral features”. Driving behavioral features are listed as acceleration and mean for speed, brake, gas, steering wheel and mean of signal data for each word (9 features in total). These features have been calculated in their determined time duration. Access to these driving behavioral features is a fundamental achievement for further analysis. From the viewpoint of statistics, driving behavioral features are very worthwhile to consider. For example, if the rate of speed acceleration is negative, we can expect speed decreasing in the next seconds. In some cases, this scenario may be accompanied with the blinking of the right signal, and interpreted as a right turn in the next few seconds.

3.2.2 Clustering driving behavioral words

After DAA provided a sequence of words and all of their driving features were calculated, our proposed method clusters them into a number of classes (number of classes = k). Clustering is being performed based on the words features.

K-mean is used for clustering the words in k different classes. Because our driving features showed quite different variances, we standardized them before using K-means.

It is important to note that we can determine k in our clustering algorithm. We set the value of k as k = 10 since it gave the best result for our data set. The words feature vector is a 9-dimensional vector that has been calculated individually for each word. Using the smaller k gave some classes that became unity while their feature vectors were not parallel. Thus, they must not have been located in the same class. On the other hand, increasing k did not increase the result accuracy.

After the clustering step, each word belongs only to one class (we call these classes as "states"). Words with the same state have similar features. For example, the speed acceleration for all members of a determined state should be similar.

In addition, we calculated 9 features (acceleration and mean for speed, brake, gas, steering wheel and mean of signal data) for our states in the same way as our words. These features are needed in the subsequent steps.

3.2.3 Constructing a probabilistic distribution table

Since the state of each word was determined, it is time to find out what is the relationship between sequential states. One of the most important questions is: Is there any specific pattern emerging from states in the word sequence and if yes, how can we find them? To answer this question, we have to find the probability of transition between any two different driving states. In order to better understand how transitions occur between various driving states, we have to introduce a new term: Previous Fixed Lag (PFL). PFL is the number of previous observed states that we need to predict the next future state. We set PFL = 2 because numbers larger than 2 did not improve the results significantly. In addition, larger PFL increases the running time of our algorithm.

After setting the value of PFL, it is time to construct the probabilistic distribution table. This table indicates the probability of transition between different states. In more detail, when PFL is 2, from each two states in the past, there is a transition to a next specific state. Thus, based on the frequency of the observed states in the training dataset, it is possible to construct a probabilistic distribution table that can specify which state has the highest likelihood to happen in the next.

The size of the table depends on the value for PFL and the number of features:

$$\text{table size} = \text{ number of classes}^{PFL} * \text{ number of classes}$$

In addition, the value of each cell is calculated from the following formula:

$$p(i,j) = \frac{n(i,j)}{\sum_{k=1}^{N} n(i,k)}$$

$p(i,j) = The\ probability\ of\ going\ to\ state\ j\ from\ state\ i$

$n(i,j) = number\ of\ times\ that\ state\ j\ appears\ directly\ after\ state\ i$

$N = number\ of\ classes$

3.3 Driving behaviors' prediction using test dataset

Predicting the upcoming driving behavior is one of the most challenging issues in high technology cars. This feature gives the ability to predict high-risk driving behaviors and prevent accidents. In this research, we focused on this issue and presented a novel framework that is able to assist drivers by predicting driving behavior parameters such as speed and steering wheel in the upcoming seconds.

We defined two parameters to be able to work with test data. The first parameter is the Number of Previous Frames (NPF) and the second one is the Number of Next Frames (NNF). The NPF parameter indicates how far in the past we want to use driving behavior data. In other words, NPF indicates the amount of driving history that we need to predict the future. Conversely, the NNF parameter determines how far in the future we want to put a benchmark for predicting the driving behavior. Different scenarios have followed to choose the best values for these parameters. Among them, we provide the results for NPF values of 15 and 30 which means that we considered the vehicular states one and two seconds before the current time. The NNF was set to 8 and 15, which means that we want to predict the vehicular behavior for the next half second to a full second.

By setting a value for NPF, it is possible to extract the 9 features (acceleration and mean for speed, brake, gas, steering wheel and mean of signal data) for the past time interval. Then, we used a multi Support Vector Machine (multi SVM) to compare extracted features with our states features (see Section 3.2.2) to find the appropriate states for the previous time intervals.

After the states are indicated for previous time intervals, our framework predicts the state of the future time interval based on the probability table.

Now that we can indicate which state the next state of the current driving behavior will be, we can then predict the speed value and the angle of steering wheel for the next upcoming seconds. After determining the next state of the vehicle, it is straightforward to calculate the value of speed for the next vehicular state. As an example, assume that our car is at second t, we want to predict the value of speed at second $t + r$. Assume that we have attained the predicted state for the time $t + r$ and it is equal to s; in this step, we look at the features of s and find the value of speed acceleration a. After finding the speed acceleration value, just a simple substitution is needed; we use $V = at + V_0$. Hence, the result is the predicted value for speed at time $t + r$.

Chapter 4

4 Results and evaluation

To evaluate the effectiveness of our proposed method, we conducted an experiment to predict speed values for our test data set. The first 15 drivers' data were selected as the training dataset and driver number 16 was selected for the test data set. We repeated the procedure with different parameters (NPF= {15, 30} , NNF = {8, 15}).

We did also the cross validation on the test data set to achieve certain results. We used the leave-one-out cross-validation in a way that 15 drivers were selected for the training set and the last driver was selected for the test case for every validation cycle.

Figure 10 shows the real and the predicted speed values for driver number 16. And Figure 10 shows the real and the predicted steering wheel values for the same driver. As we can see in Figure 10, although the two diagrams do not have a complete overlap, they show very similar values at each point in time; and there is not any significant difference between the real values and the predicted values.

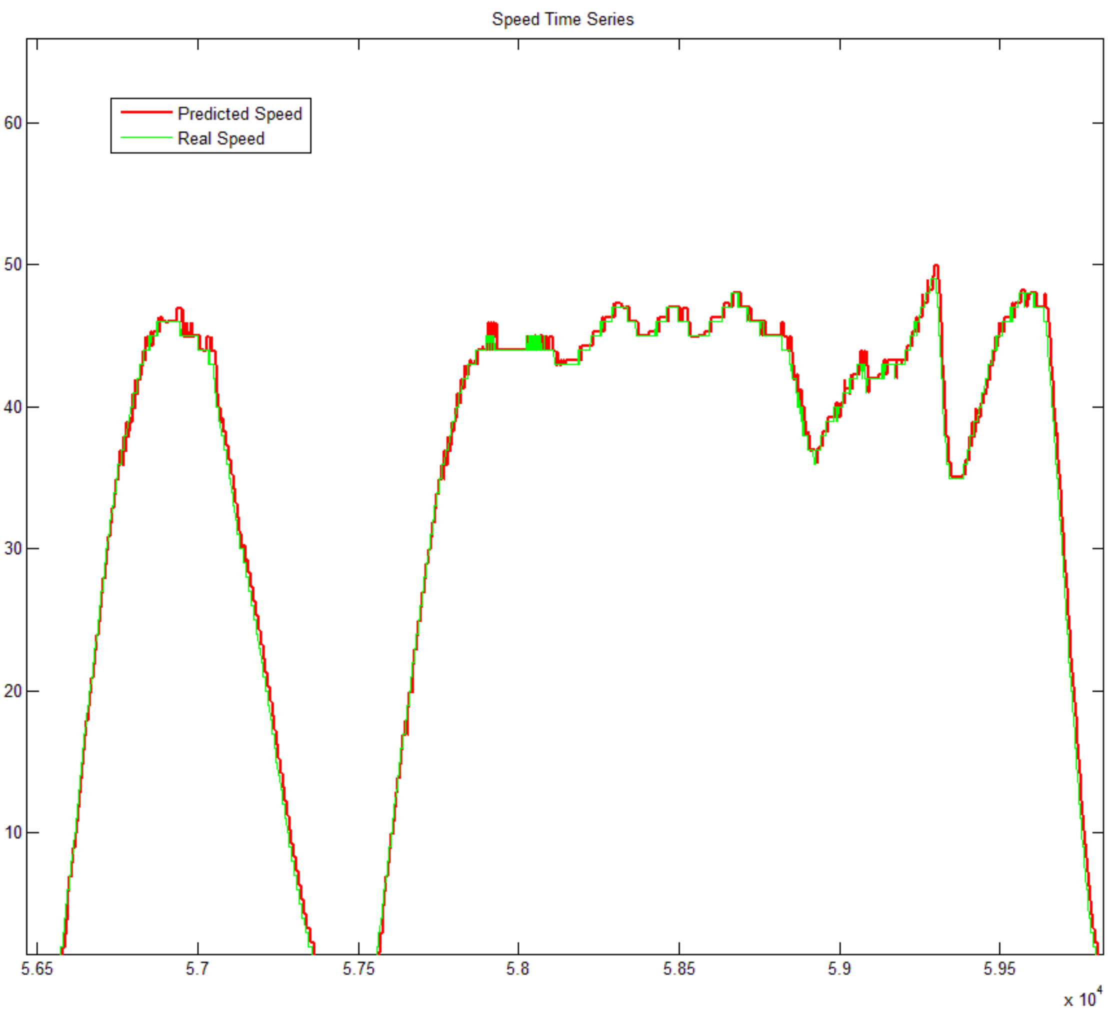


Figure 10 : The real and the predicted speed values for driver number 16.

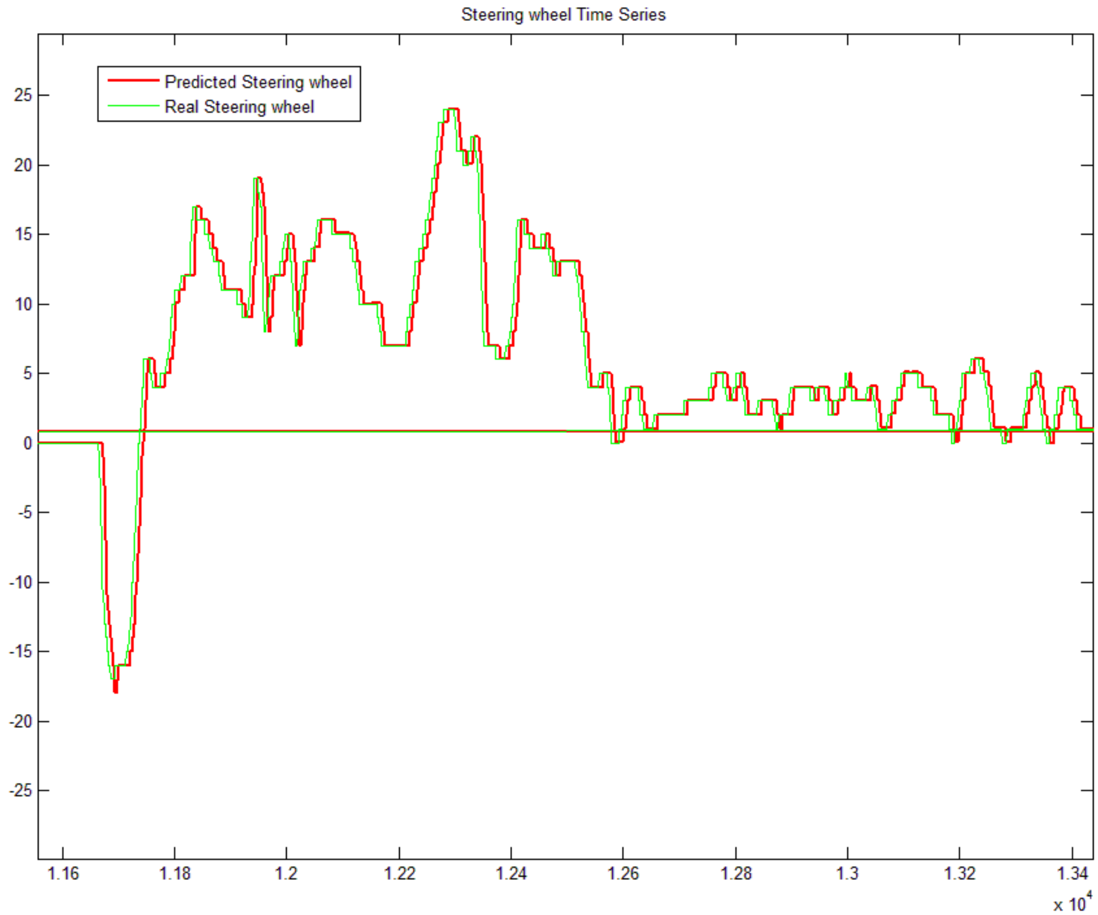


Figure 11 : The real and the predicted steering wheel values for driver number 16.

To quantitatively evaluate the results, we calculated Mean Squared Error (MSE) for the predicted values to see the accuracy of the algorithm in different scenarios. In addition, we defined a confidence interval for each predicted value, which is shown in below:

$$| \text{Predicted\_value} - \text{Real\_value}| \leq 1.50$$

We set confidence interval equal and less than 1.50, because predicting the car speed for the next second with 1.5 kilometer per hour error is considered as a reliable value in our work. I.e. the system can correctly detect dangerous situation as long as we predict the car speed in the confidence interval range. Same situation applies to steering wheel angle confidence interval. Based on the confidence interval, it is easy to find out what percentage of our results is reliable. Table 2, Table 3, and Table 4 present the framework results for the speed prediction, while Table 5 and Table 6 present results for the steering wheel prediction.

Table 2: Speed prediction results using gas, brake and speed datasets. This table displays the percentage of reliable predicted values satisfying | Predicted_value - Real_value| ≤ 1.50.

Number of letters	Time interval in the past	Time interval in the future	MSE	Reliable predicted values
25	1 second	0.5 second	0.5055	95.28%
25	2 second	1 second	1.4351	82.73%
50	1 second	0.5 second	0.6673	93.96%
50	2 second	1 second	1.9793	77.88%

Table 3: Speed prediction results using gas, brake, speed and steering wheel datasets. This table displays the percentage of Reliable predicted values satisfying | Predicted_value - Real_value| ≤ 1.50.

Number of letters	Time interval in the past	Time interval in the future	MSE	Reliable predicted values
25	1 second	0.5 second	0.4839	95.19%

25	2 second	1 second	1.3653	83.19%
50	1 second	0.5 second	0.4566	95.29%
50	2 second	1 second	1.3477	82.95%

Table 4: Speed prediction results using gas, brake, speed, steering wheel and signals datasets. This table displays the percentage of reliable predicted values satisfying | Predicted_value - Real_value| ≤ 1.50.

Number of letters	Time interval in the past	Time interval in the future	MSE	Reliable predicted values
25	1 second	0.5 second	0.5004	95.04%
25	2 second	1 second	1.4356	83.29%
50	1 second	0.5 second	0.4921	94.26%
50	2 second	1 second	1.3611	83.38%

According to the predicted speed results in Table 2, Table 3, and Table 4, it is noticeable that adding the steering wheel parameter to speed, brake, and gas parameters could improve the final results, while adding the turn signals parameters did not provide any improvement. Consequently, the combination of speed, gas, brake, and steering wheel outperformed other combinations. The best result has been obtained by setting the number of letters to 50, the past time interval to 1 second, and the future time interval to half of a second. About 95.29% of the predicted speed values are in the confidence interval and the total mean squared error for the predicted speed values is 0.4566 for the best result.

It is important to notice that increasing the number of letters does not make considerable changes to final results. In contrast, using larger time intervals in the past and future causes worse results, which are not precise enough to be considered herein.

Table 5: Steering wheel prediction results using gas, brake, speed and steering wheel datasets. This table displays the percentage of reliable predicted values satisfying |Predicted_value - Real_value| ≤ 1.50.

Number of letters	Time interval in the past	Time interval in the future	MSE	Reliable predicted values
25	1 second	0.5 second	19.1612	88.70%
25	2 second	1 second	62.6855	79.21%
50	1 second	0.5 second	19.7761	85.99%
50	2 second	1 second	64.4277	76.76%

Table 6: Steering wheel prediction results using gas, brake, speed, steering wheel and signals datasets. This table displays the percentage of reliable predicted values satisfying |Predicted_value - Real_value| ≤ 1.50.

Number of letters	Time interval in the past	Time interval in the future	MSE	Reliable predicted values
25	1 second	0.5 second	19.1387	89.64%
25	2 second	1 second	62.6632	80.48%
50	1 second	0.5 second	19.1213	89.66%

50	2 second	1 second	62.4713	79.89%

In accordance with the results in Table 5 and Table 6, it is obvious that adding turn signal parameters could significantly decrease the total mean squared error. There is not a considerable difference in results when we change the number of letters. The best result has been obtained when the past time interval was set to 1 second, and the future time interval to half of a second. It shows about 89.66% of the predicted steering wheel values are in the confidence interval and the total mean squared error is 19.1213.

Chapter 5

5 Summary and Conclusions

Although the invention of automobiles has been an important achievement in human life, unfortunately it is one of the major causes of death and injuries. In most cases, human driving errors are reported as the main cause for transportation accidents. In order to prevent crashes and mitigate their fatalities, there has been a consistent need to improve safety and create assistance systems for automobiles. In recent years, researchers focused on Advanced Driving Assistance Systems (ADAS) to reduce the number and the severity of accidents. In this thesis, we studied the existing literature in the field of ADAS development, as it pertains to maneuver prediction. Thereafter, we presented a novel framework for ADAS able to predict some aspects of driver behavior for short periods of time.

5.1 Conclusions

Our proposed framework is able to predict the next driving behavior in one or half of a second based on previously observed driving behavior. Our data includes speed, brake, gas, steering wheel, and turn signals time-series. We could not find any other research in this field that uses this variety of sensors together (CANbus database time-series). Different combinations of CANbus data were considered to see which one is more efficient. In order to obtain a powerful prediction model, first sticky HDP-HMM and NPYLM models were used together prior to DAA to chunk time-series into a sequence of time intervals (words).

Following this, a number of features were defined for each word such as speed, acceleration, and mean velocity. Observing these features over time has a critical impact on finding an appropriate pattern for further analysis. For example, knowing that speed is increasing or decreasing in a number of specific intervals leads us to predict similar driving behaviors. Accordingly, time intervals (words) were clustered into a number of classes based on their features. Each class consists of a number of words with similar features that differ from other classes' members. For example, it is possible that the speed decreases in two classes while the steering wheel angle changes in only one of them. The first class behavior is interpreted as "turning" while the second one is interpreted as "stopping" or "speed decreasing". Once the data was trained and patterns were found, the model was applied to the test data. We trained our model with three different combinations of time series and predicted the speed and steering wheel values for the test data. As previously mentioned, the combination of speed, brake, gas, and steering wheel was the most successful combination for speed prediction. Our model is able to predict speed values in 95% of cases, subjected to the confidence interval. Moreover, the combination of speed, brake, gas, steering wheel, and signals lead our method to predict the steering wheel angle with more than 89% accuracy, subjected to the same confidence interval.

A major distinction of our framework for driver behavior prediction consists in the fact that the sum of our algorithms run automatically and are independent of any human annotation or tagging. Moreover, the ability to predict upcoming driving behaviors relatively precisely for the next half second offers any existing ADAS ample time to intervene and mitigate consequences, should the next predicted maneuver be inconsistent with the current driving situation.

5.2 Future Work

All algorithms of the presented framework have been programmed in a very friendly, expandable, and reusable structure that could be used for further development. This implementation has the potential to be combined other, supplemental data sources in order to increase the accuracy and range of maneuver prediction.

Bibliography

Angell, L. S., Auflick, J., Austria, P. A., Kochhar, D. S., Tijerina, L., Biever, W., ... & Kiger, S. (2006). Driver workload metrics task 2 final report (No. HS-810 635).

Bando, T., Takenaka, K., Nagasaka, S., & Taniguchi, T. (2013, June). Unsupervised drive topic finding from driving behavioral data. In *Intelligent Vehicles Symposium (IV), 2013 IEEE* (pp. 177-182). IEEE.

Bando, T., Takenaka, K., Nagasaka, S., & Taniguchi, T. (2013, November). Automatic drive annotation via multimodal latent topic model. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on* (pp. 2744-2749). IEEE.

Beal, M., & Krishnamurthy, P. (2012). Gene expression time course clustering with countably infinite hidden Markov models. arXiv preprint arXiv:1206.6824.

Beauchemin, S. S., Bauer, M., Kowsari, T., & Cho, J. (2012). Portable and scalable vision-based vehicular instrumentation for the analysis of driver intentionality. Instrumentation and Measurement, IEEE Transactions on, 61(2), 391-401.

Brookhuis, K. A., & de Waard, D. (2010). Monitoring drivers' mental workload in driving simulators using physiological measures. Accident Analysis & Prevention, 42(3), 898-903.

Fox, E. B., Sudderth, E. B., Jordan, M. I., & Willsky, A. S. (2007). The sticky HDP-HMM: Bayesian nonparametric hidden Markov models with persistent states. *Arxiv preprint*.

Fox, E. B., Jordan, M. I., Sudderth, E. B., & Willsky, A. S. (2009). Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems* (pp. 549-557).

Fox, E. B., Sudderth, E. B., Jordan, M. I., & Willsky, A. S. (2011). A sticky HDP-HMM with application to speaker diarization. The Annals of Applied Statistics, 1020-1056.

Gales, M., & Young, S. (2008). The application of hidden Markov models in speech recognition. *Foundations and trends in signal processing*, *1*(3), 195-304.

Hamada, R., Kubo, T., Ikeda, K. I., Zhang, Z., Shibata, T., Bando, T., & Egawa, M. (2013, May). Towards prediction of driving behavior via basic pattern discovery with BP-AR-HMM. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on* (pp. 2805-2809). IEEE.

Hermann, A., & Desel, J. (2008, September). Driving situation analysis in automotive environment. In *Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on* (pp. 216-221). IEEE.

Hoffman, M., Cook, P., & Blei, D. (2008). Data-driven recomposition using the hierarchical Dirichlet process hidden Markov model. In Proc. International Computer Music Conference.

Hülnhagen, T., Dengler, I., Tamke, A., Dang, T., & Breuel, G. (2010, June). Maneuver recognition using probabilistic finite-state machines and fuzzy logic. In *Intelligent Vehicles Symposium (IV), 2010 IEEE* (pp. 65-70). IEEE.

Johnson, M. J., & Willsky, A. S. (2013). Bayesian nonparametric hidden semi-Markov models. The Journal of Machine Learning Research, 14(1), 673-701.

Kishimoto, Y., & Oguri, K. (2008, September). A modeling method for predicting driving behavior concerning with driver's past movements. In*Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on* (pp. 132-136). IEEE.

Kivinen, J. J., Sudderth, E. B., & Jordan, M. I. (2007, October). Learning multiscale representations of natural scenes using Dirichlet processes. In Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on (pp. 1-8). IEEE.

Mitrović, D. (2005). Reliable method for driving events recognition. *Intelligent Transportation Systems, IEEE Transactions on*, *6*(2), 198-205.

Mochihashi, D., Yamada, T., & Ueda, N. (2009, August). Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In*Proceedings of the Joint Conference of the 47th Annual Meeting* *of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1* (pp. 100-108). Association for Computational Linguistics.

Nigro, J. M., & Rombaut, M. (2003). IDRES: A rule-based system for driving situation recognition with uncertainty management. Information Fusion, 4(4), 309-317.

Rabiner, L. R., & Juang, B. H. (1986). An introduction to hidden Markov models. *ASSP Magazine, IEEE*, *3*(1), 4-16.

Schneider, J., Wilde, A., & Naab, K. (2008, June). Probabilistic approach for modeling and identifying driving situations. In *Intelligent Vehicles Symposium, 2008 IEEE* (pp. 343-348). IEEE.

Stanculescu, I., Williams, C. K., & Freer, Y. (2014). Autoregressive Hidden Markov Models for the Early Detection of Neonatal Sepsis. *Biomedical and Health Informatics, IEEE Journal of*, *18*(5), 1560-1570.

Takano, W., Matsushita, A., Iwao, K., & Nakamura, Y. (2008, September). Recognition of human driving behaviors based on stochastic symbolization of time series signal. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (pp. 167-172). IEEE.

Taniguchi, T., & Nagasaka, S. (2011, December). Double articulation analyzer for unsegmented human motion using pitman-yor language model and infinite hidden markov model. In *System Integration (SII), 2011 IEEE/SICE International Symposium on* (pp. 250-255). IEEE.

Taniguchi, T., Nagasaka, S., Hitomi, K., Chandrasiri, N. P., & Bando, T. (2012, June). Semiotic prediction of driving behavior using unsupervised double articulation analyzer. In *Intelligent Vehicles Symposium (IV), 2012 IEEE* (pp. 849-854). IEEE.

Taniguchi, T., Nagasaka, S., Hitomi, K., Takenaka, K., & Bando, T. (2015). Unsupervised hierarchical modeling of driving behavior and prediction of contextual changing points. *Intelligent Transportation Systems, IEEE Transactions on*, *16*(4), 1746-1760.

Vidal, E., Thollard, F., De La Higuera, C., Casacuberta, F., & Carrasco, R. C. (2005). Probabilistic finite-state machines-part I. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *27*(7), 1013-1025.

Wang, C., Blei, D., & Li, F. F. (2009, June). Simultaneous image classification and annotation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (pp. 1903-1910). IEEE.

Curriculum Vitae

Name:	Maedeh Hesabgar
Post-secondary Education and Degrees:	Sharif University of Technology Tehran, Iran 2008-2012 B.Sc.
	The University of Western Ontario London, Ontario, Canada 2014-2015 M.Sc.
Honors and Awards:	Western Graduate Research Scholarship 2014-2015
Related Work Experience	Research Assistant University of Western Ontario 2014-2015
	Teaching Assistant University of Western Ontario 2014-2015
	Teaching Assistant Sharif University of Ontario 2011-2012