Master's Thesis Defense

Use of Information Theory Measures Extracted From OBD-II Interface Data for Driver Identification.

Gean da Silva Santos
geans.santos@gmail.com

Advisor:

Dr. André Luiz Lins de Aquino

Maceió
Março, 2024

Gean da Silva Santos

Use of Information Theory Measures Extracted From OBD-II Interface Data for Driver Identification.

Master's Thesis presented by Gean da Silva Santos to the Postgraduate Program in Informatics at the Federal University of Alagoas as a requirement for obtaining the Master's degree in Informatics.

Advisor:

Dr. André Luiz Lins de Aquino

Maceió
Março, 2024

Catalogação na Fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecário: Marcelino de Carvalho Freitas Neto – CRB-4 - 1767

S237u Santos, Gean da Silva.

Use of information theory measures extracted from OBD-II interface data for driver identification / Gean da Silva Santos. – 2024.

45 f. : il.

Orientador: André Luiz Lins de Aquino.

Dissertação (mestrado em informática) - Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2024.

Texto em inglês.

Bibliografia: f. 30-34.

1. Identificação de motorista (Aprendizado do computador). 2. Teoria da informação. 3. Complexidade estatística. 4. Shannon, Entropia de. 5. Fisher, informação de. 6. Aprendizado do computador. 7. Dados de veículo. 8. Sistema de Diagnóstico Embarcado. I. Título.

CDU: 004.85

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE ALAGOAS
INSTITUTO DE COMPUTAÇÃO
Av. Lourival Melo Mota, S/N, Tabuleiro do Martins, Maceió - AL, 57.072-970
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO (PROPEP)
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Folha de Aprovação

GEAN DA SILVA SANTOS

USO DE MEDIDAS DE TEORIA DA INFORMAÇÃO EXTRAÍDAS DE DADOS DA INTERFACE OBD-II PARA IDENTIFICAÇÃO DO MOTORISTA

USE OF INFORMATION THEORY MEASURES EXTRACTED FROM OBD-II INTERFACE DATA FOR DRIVER IDENTIFICATION

Dissertação submetida ao corpo docente do Programa de Pós-Graduação em Informática da Universidade Federal de Alagoas e aprovada em 25 de março de 2024.

Banca Examinadora:

ANDRE LUIZ LINS DE AQUINO:03235015400
Assinado de forma digital por ANDRE LUIZ LINS DE AQUINO:03235015400
Dados: 2024.04.22 20:16:42 -03'00'

Prof. Dr. ANDRE LUIZ LINS DE AQUINO
UFAL – Instituto de Computação
Orientador

gov.br Documento assinado digitalmente
OSVALDO ANIBAL ROSSO
Data: 01/05/2024 17:38:56-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. OSVALDO ANIBAL ROSSO
UFAL – Instituto de Física
Examinador Externo

gov.br Documento assinado digitalmente
FABIANE DA SILVA QUEIROZ
Data: 25/04/2024 10:16:08-0300
Verifique em https://validar.iti.gov.br

Profa. Dra. FABIANE DA SILVA QUEIROZ
UFAL – Campus de Engenharia e Ciências Agrárias
Examinador Interno

gov.br Documento assinado digitalmente
DENIS LIMA DO ROSARIO
Data: 23/04/2024 09:08:34-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. DENIS LIMA DO ROSARIO
UFPA-Universidade Federal do Pará
Examinador Externo

gov.br Documento assinado digitalmente
RAQUEL DA SILVA CABRAL
Data: 24/04/2024 20:46:59-0300
Verifique em https://validar.iti.gov.br

Profa. Dra. RAQUEL DA SILVA CABRAL
UFAL – Campus Arapiraca
Examinador Interno

Abstract

We investigate the use of Machine Learning tools applied to driver identification. We propose using Information Theory measures as features in Machine Learning models. The measures used are Statistical Complexity, Permutation Entropy, and Fisher Information. We calculate these measures over the raw data of On Board Diagnostics version 2 (OBD-II) sensor values and use them as new features applying to the following classification models: k-Nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, Multi-Layer Perceptron, and Naive Bayes. We evaluate the driver identification scenario considering sliding windows of 120 samples and an overlap of 60 samples. In each window, the well-established models are trained and evaluated. The models are analyzed by accuracy, by the Area Under the Curve of the Receiver Operating Characteristic Curve (which determines how relevant the classifier is in terms of its sensitivity and specificity), by precision and by recall. We observed significant gains for all models considered in the scenario. Following the traditional procedure, we obtained the values: 78.5 to 91.1 % of accuracy, 74.5 to 87.8 % of ROC AUC, 62.5 to 84.2 % of precision, and 60.6 to 85.3 % of recall. While using our proposal, we obtained the values: 94.3 to 99.9 % of accuracy, 91.7 to 99.9 % of ROC AUC, 89.8 to 99.9 % of precision, and 86.4 to 99.9 % of recall.

Keywords: driver identification, information theory, statistical complexity, Shannon entropy, Fisher information, machine learning, vehicle data, on board diagnostic (OBD).

Resumo

Investigamos o uso de ferramentas de Aprendizado de Máquina aplicadas à identificação de motoristas. Propomos o uso de medidas de Teoria da Informação como recursos em modelos de Aprendizado de Máquina. As medidas utilizadas são Complexidade Estatística, Entropia de Permutação e Informação de Fisher. Calculamos essas medidas sobre os dados brutos dos valores dos sensores do Sistema de Diagnóstico Embarcado versão 2 (OBD-II) e as utilizamos como novas características aplicadas aos seguintes modelos de classificação: k-Nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, Multi-Layer Perceptron e Naive Bayes. Avaliamos o cenário de identificação de motorista considerando janelas deslizantes de 120 amostras e uma sobreposição de 60 amostras. Em cada janela, os modelos bem estabelecidos são treinados e avaliados. Os modelos são analisados pela acurácia, pela Área Sob a Curva da Curva Característica de Operação do Receptor (que determina quão relevante o classificador é em termos de sua sensibilidade e especificidade), pela precisão e pela cobertura. Observamos ganhos significativos para todos os modelos considerados no cenário. Seguindo o procedimento tradicional, obtivemos os valores: de 78,5 a 91,1% de acurácia, de 74,5 a 87,8% de ROC AUC, de 62,5 a 84,2% de precisão e de 60,6 a 85,3% de revocação. Enquanto usando nossa proposta, obtivemos os valores: de 94,3 a 99,9% de acurácia, de 91,7 a 99,9% de ROC AUC, de 89,8 a 99,9% de precisão e de 86,4 a 99,9% de cobertura.

Palavras-chave: identificação de motorista, teoria da informação, complexidade estatística, entropia de Shannon, informação de Fisher, aprendizado de máquina, dados de veículo, diagnóstico embarcado (OBD).

Acknowledgement

I extend my heartfelt gratitude to my beloved wife, Jéssika, and our cherished daughters, Abigail, Noemi, and Debora. Additionally, I wish to express my deepest appreciation to my parents, Gerson and Josenilda, who have always provided me with unwavering support whenever I needed it.

My aspiration is for my daughters to one day surpass both Jéssika and me. For now, our focus remains on being the best possible examples for them.

Jéssika has been a steadfast presence by my side, offering unwavering support to both me and our children. She is a beacon of light, always guiding us in the right direction with her faith, wisdom and discernment.

Contents

List of Figures

List of Tables

1

Introduction

Driving is essential in people's daily lives. Moreover, people relate directly or indirectly to driving. In other words, even those who do not drive vehicles, for instance, use public transportation, receive packages, cross busy streets, among other things.

Modern automobiles have several sensors that govern numerous operations, such as safety features, vehicle control, and infotainment (4). These sensors are connected through an in-vehicle network called the Controller Area Network (CAN). That is, CAN is an internal vehicle network where the values of all vehicle sensors are transmitted. To externalize these data, the On-Board Diagnostic version 2 (OBD-II) interface is used. This interface is usually located near the driver's seat and below the steering wheel as shown in Figure 1.1. The OBD standard arose from the need to reduce the costs of diagnosing cars in workshops and, historically, the need for pollutant emission control.

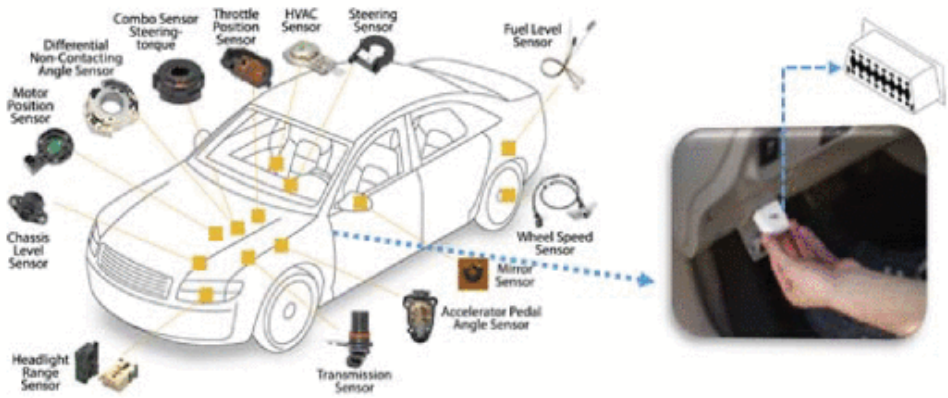


Figure 1.1: Car's sensors and OBD-II interface (1)

The vehicle sensor data, when collected throughout the journey, can represent the driver's driving behavior. Studies in the literature utilize this data for various applications, such as:

- *Driver identification*: to offer data to car insurers, to suspect theft, or even to have greater control over a fleet of vehicles.
- *Prediction of polluting gas emissions*: for environmental control.
- *Identification of road/traffic conditions*: to assist in monitoring.
- *Identification of driver emotions*: for customized ITS applications, or for risk alert.

Studies using vehicle data usually fit into one of these problems: classification problem, clustering problem, and regression problem. Examples of classification problems are the identification of drivers (1) and the classification of driver behavior. In a classification problem, we seek to predict the category (or class) of an observation, instance, or input. A regression problem is similar to a classification problem, but it tries to predict a real value in a continuous domain and not a class (5). Examples of regression problems are predicting fuel consumption, polluting gas emissions, driver risk, and wear and tear of parts. At the same time, a clustering problem aims to group observations into clusters (6). Examples of clustering problems are grouping drivers to identify driving profiles (6) or grouping trip data to associate with different road types or conditions (such as highway, urban area, unpaved road as in (7)).

Many studies aiming for driver identification employ deep learning models to achieve high accuracy scores. However, the performance in other important metrics such as precision and recall is often overlooked. Additionally, this type of model requires extensive hyperparameter optimization and training time.

Moreover, other studies have used Information Theory measures to characterize time series data. We hypothesize that extracting Information Theory measures from vehicle driving data can be valuable resources for machine learning models to become more effective. Thus, our objectives are to describe a method for constructing a new dataset from Information Theory measures and to compare the models performance using the proposed method and the default method from literature.

We apply our proposal to driver identification using the dataset "This car is mine!" (8). This dataset has 51 features, each from a vehicle sensor, with a public parameter ID (PID) regulated by SAE J1979 (9). We compare our proposal with standard preprocessing method in the literature. We apply the preprocessed data to some classic classification algorithms used in the literature for driver identification. In our experiments, the classifiers presented better accuracy using our proposal. This improvement is also evidenced by showing the Area Under the Curve (AUC) for Receiver Operating Characteristic Curve (ROC), the precision, and the recall results.

The remainder of the work is organized as follows: Chapter 2 contains a discussion of work related to driver identification and Information Theory; Chapter 3 contains the preprocessing carried out following the literature and the proposal, and the description of the Machine Learning model used; Chapter 4 describes the metrics used, how the Information Theory measures were extracted, and how the models and methods were evaluated; Chapter 5 concludes the work.

2

Related Works and Concepts

We consider different research sources by the various works that address the classification of drivers or prediction of polluting gas emissions. One of the most reliable sources of information about the car's movement is via simulator (10, 11, 12) or via OBD-II (1, 13), one of the most prevalent technologies which provide data from the vehicle ECUs (14). Alternatively, we can obtain the information through smartphone sensors (accelerometer, GPS, etc.). Some other works include data from physiological sensors (15). Finally, works still add other sensors such as cameras and microphones in the vehicles (16, 17). However, we consider only OBD-II data because they provide information already available in the car, not requiring the addition of sensors.

According to Manderna and Kuma (4), combining vehicle telematics data with relevant Machine Learning algorithms enables the recognition of distinct driving styles, the revelation of driving behavior and patterns, and even the detection of dangerous driving behaviors. Furthermore, applications for identifying patterns and characterization (18), and prediction (19) can also be achieved due to the time series characteristic of the data, and path kind identification (i.e., dirt road, road with bumps, and highway) (7). Finally, some studies aim to cluster the data (20, 6), which may have the following purposes: identifying driver profiles (aggressive, moderate, calm), identifying road conditions (motorway, urban area, unpaved) or traffic conditions (slow, moderate, fast).

Recent works use different models for their applications. In particular, for the case of classification, studies have been using both classic Machine Learning (21, 22, 23, 24) and Deep Learning models (25, 1, 13, 26). Among the Machine Learning algorithms are kNN, Random Forest, Decision Tree, SVM, and Naive Bayes. And among the Deep Learning algorithms are Long Short-Term Memory (LSTM), Generative Adversarial Networks (GAN), and Autoencoder.

In this work, we will use Information Theory measures for training and evaluating Machine Learning models for driver identification. Section 2.1 discuss the application case of driver identification and Section 2.2 discusss the Information Theory measures.

2.1 Driver Identification Case

2.1.1 Driving Data and Application

Different research addressed the drivers' classification, utilizing various data collection methods. Some of the most reliable sources of information about the car's movement are simulators (10, 11, 12), and OBD-II technology (1, 13), which provide data from the vehicle's ECUs (14). Alternatively, they obtain the information through smartphone sensors such as accelerometers or GPS. Additionally, some studies incorporate data from physiological sensors (15), cameras, and microphones in vehicles (16, 17). However, our research solely considers OBD-II data due to its availability within the car without requiring additional sensors.

Xun et al. (22) and Uvarov et al. (21) raise a problem with advertising parameters read by the OBD-II interface. This problem is because manufacturers may not identify all data present on the CAN bus, keeping them confidential. However, Uvarov et al. deal with this problem using only some of the public data standardized by SAE J1979 (9), while Xun et al. use all data regardless of whether they are publicly available. In our work, we use the public data restriction, as we observed that all 51 sensors present in the (8) dataset are public PIDs.

Combining vehicle telematics data with Machine Learning algorithms enables the recognition of distinct driving styles, behaviors, and patterns (4). Furthermore, the time-series nature of the data allows for improvement in the performance of different applications such as pattern identification, characterization (18), and prediction (19), identifying various types of roads (dirt roads, roads with bumps, or highways) (7).

2.1.2 Machine Learning Applied to Driver Identification

Most studies follow a preprocessing stage before applying the data to the models. This preprocessing typically involves several standard techniques, including removing missing data, eliminating invariant features or features highly correlated with others, and selecting the best features through feature selection methods. Other techniques are also employed: value rescaling (normalization) to ensure all features are on the same scale and Principal Component Analysis (PCA) to reduce the number of features by discarding components with less variation. Furthermore, we used a sliding window to increase the number of sequences evaluated.

By employing these preprocessing techniques, researchers aim to enhance the quality and relevance of the data, thus improving the performance of the subsequent classification models in the driver identification task. Furthermore, Priyadharshini et al. (27) analyze the feature selection problem for subjective driver identification and then propose using feature selection algorithms to analyze each feature's importance. As our proposal, this work presents an improvement in the data preprocessing stage.

Particularly in the case of classification, they use classic Machine Learning algorithms

such as k-Nearest Neighbors (kNN), Random Forest, Decision Trees, Support Vector Machines (SVM), or Naive Bayes have been employed (21, 22, 23, 24). They also use Deep Learning models such as Long Short-Term Memory (LSTM), Generative Adversarial Networks (GAN), or Autoencoders (25, 1, 13, 26). By considering these diverse approaches and models, researchers have made significant progress in driver identification, behavior analysis, pattern identification, and predictive modeling, leveraging various data collection methods and Machine Learning algorithms.

The problem of driver identification is a classification one. Considering the characteristics of the time series data from various sensors, we have a multivariate time series classification problem. To address this problem, we employ Machine Learning models for classification. Several studies on driver identification propose the use of classic Machine Learning models (21, 22), while others suggest the utilization of Deep Learning models (25, 1, 4).

Girma et al. (1) and Mekki et al. (25) propose new models based on Deep Learning, seek to identify different drivers, analyze the model's accuracy with anomalies in the data, and use other datasets. Similarly, Manderna et al. (4) propose a new model for driver identification, compare it with some classic algorithms, and evaluate performance through accuracy, precision, F1-score, and recall metrics. They measure loss and accuracy over the algorithm's training epochs. In contrast, our proposal increments the preprocessing phase of well-established classifiers, analyzing the accuracy, precision, recall, and ROC AUC scores in each data window and at the preprocessing time. Our proposal improved the performance of non-Deep Learning models for values very close to the maximum of the metrics used.

Park and Kim (13) raised the problem of having only data from a vehicle owner to train a model that identifies that driver. They propose using Recursive Generative Adversarial Networks (RGAN) to stop such issues. This work clearly defines the preprocessing step used in our work. We use the dataset they made available in this work.

Table 2.1 summarizes the qualitative analysis mentioned Machine Learning methods. However, it is essential to note that many of these previous works propose novel models, introduce feature constraints, minimize training time, or optimize input size. Only Priyadharshini et al. (27) analyze approaches to improve data preprocessing.

2.2 Information Theory Measures

2.2.1 Ordinal series from Bandt-Pompe

The quantifiers from Information Theory rely on a probability distribution associated to the time series. The usual histogram technique is inadequate since the data are treated purely stochastic and the temporal information is completely lost, in the meantime, Bandt-Pompe methodology used to take into account time ordering of the time series by comparing neighboring values in a time series (2).

Table 2.1: Qualitative analysis of related works.

Work	Dataset	Preprocessing	Algorithm	Evaluation	Accuracy (%)
(25)	Security Driving Dataset, UAH Driveset, HCILAB Driveset, OSF Dataset	Data cleaning, feature selection, normalization, slid window: 60s.	LSTM, StackedRNN, NoPoolCNN, CNN	Accuracy, cross-validation	57.2 to 95.1
(21)	OCS Lab security	Add features (acceleration, jerk), normalization, slid windows:30s.	Random Forest, Decision Tree, kNN, Gradient Boosting, SVM	Accuracy, cross-validation	79 to 99
(1)	Security Driving Dataset, Vehicular data trace Dataset-1, Vehicular data trace Dataset-2	No preprocessing	LSTM, FCNN, Decision Tree, Random Forest	Accuracy, precision, f1-score, recall	97.2 to 99.1
(22)	Own dataset	Normalization, PCA (to reduce features)	Naive Bayes, kNN, Naive Bayes + kNN	Accuracy, confusion-matrix	28 to 95
(4)	Own dataset	Remove miss values, normalization	LSTM, SVM, Decision tree, kNN, MLP 3 layer	Accuracy, precision, f1-score, recall	99
(27)	HCILAB Driver-dataset	Normalization, dimensionality reduction	KNN, SVM, MLP, Random Forest, Decision Tree	Accuracy, training time	96.04
(13)	“This car is mine" (8)	Remove highly-correlation and non-influential features, sliding window, normalization	RGAN	Accuracy, precision, f1-score, recall, training time	88.4
Our	“This car is mine" (8)	Remove highly-correlation and non-influential features, sliding window, normalization, and add features (Information Theory measures)	KNN, Linear SVM, RBF SVM, Random Forest, Decision tree, MLP, Naive Bayes	Accuracy, precision, recall, ROC AUC, cross-validation	89.4 to 99.9

To extract the time series measures, we used the Bandt-Pompe (BP) method (28) to transform raw data into a histogram. Specifically, the BP symbolization method assigns probability distributions from the time series under consideration, i.e., the temporal causality of the process. In this sense, given a time series $\boldsymbol{X}(t) = \{x_t : t = 1,\ldots,N\}$, an embedding dimension $D \geq 2 (D \in \mathbb{N})$, and an embedding delay time $\tau \in \mathbb{N}$, we compute the ordinal patterns of order D (pattern length) generated by

$$(s) \mapsto \left(x_{s-(D-1)\tau}, x_{s-(D-2)\tau}, \ldots, x_{s-\tau}, x_s\right) \tag{2.1}$$

Afterward, we assign each point in time s with a D-dimensional vector resulting from evaluating the sequence at time $s-(D-1)\tau$, $\ldots, s-\tau, s$. We incorporate more information about the past into the vector by considering a higher D value. According to the pattern, the meaning of order D with respect to time s is permutation $\pi = \{r_0, r_1, \ldots, r_{D-1}\}$ of $\{0, 1, \ldots, D-1\}$ is defined as

$$x_{5-r_{D-1}\tau} \leq x_{5-r_{D-2}\tau} \leq \cdots \leq x_{5-r_1\tau} \leq x_{5-r_0\tau} \tag{2.2}$$

Thus, we convert the data produced by Eq. 2.1 to the unique symbol π. To get unambiguous results, we set $r_i < r_{i-1}$ if $\chi_{s-r_i} = \chi_s r_{i-1}$. If $X(t)$ follows a slightly continuous distribution, the probability of equal values is zero. Therefore, we calculate the associated relative frequencies for all $D!$ and the possible permutations π of order D, which this particular ordered sequence was found in the time series divided by the total number of sequences. Hence, the histogram $P \equiv \{p(\pi)\}$ is defined as

$$p(\pi) = \frac{\#\{s \text{ of type } \pi : s \leq N-(D-1)\tau\}}{N-(D-1)\tau}, \tag{2.3}$$

where # is the cardinality of the set.

2.2.2 Shannon Entropy

The second concept is Shannon Entropy, a global measure of self-information. Shannon Entropy, developed by Claude Shannon in 1948 (29), is a measure that quantifies the average uncertainty in a source of information or dataset. It is calculated as the weighted average of the probabilities of each event occurring, indicating greater uncertainty when the probabilities are more evenly distributed. This measure is associated with the concept that the more uncertain the outcome of a random experiment, the more information is gained by observing its occurrence.

Let $\mathcal{X} = \left\{x_j : j = 1,\ldots,M\right\}$ be a discrete random variable of length $M < \infty$ whose distribution features is the probability function $P = \{p_i : i = 1,\ldots,M\}$. p_i represents the probability of state i, and $\sum_{i=1}^{M} p_i = 1$, and M is the number of possible states of the checked system. The

well-known Shannon entropy is

$$S[P] = -\sum_{i=1}^{M} p_i \ln p_i, \tag{2.4}$$

Among them, $p_i \ln p_i = 0$ if $p_i = 0$, it is related to the physical process described by P. Once the Shannon entropy $S[P] = 0$, the information (knowledge) of the underlying process described by P is maximal, and we can predict possible outcomes with complete certainty. On the other hand, if the physical process follows a uniform probability distribution $P_e = \{p_i = 1/M, \forall i = 1, \ldots, M\}$ then little knowledge is obtained (28).

It is also helpful to define the so-called normalized Shannon entropy to evaluate the self-information in a normalized way, denoted by

$$\mathcal{H}[P] = \frac{S[P]}{S_{max}} = \frac{S[P]}{S[P_e]} = \frac{S[P]}{ln M}. \tag{2.5}$$

2.2.3 Statistical Complexity

Statistical Complexity is a measure that assesses the structural irregularity of datasets, such as time series. It aims to capture the diversity and patterns of order within the data. Thus, López-Ruiz *et al.* (1995) (30) proposed a definition of complexity based on intuitive notions from physics and statistics. A system is considered simple when it can be described with little *information*, whereas a system is deemed complex when a large amount of *information* would be required to describe its state.

We need to find a proper measure of complexity based on classification or information alone. In this case, Lamberti *et al.* (2004) (31) proposed a Statistical Complexity ($\mathcal{C}$) measure $\mathcal{C}_{JS[P]}$, which can identify critical dynamic details in the temporal series. López-Ruiz *et al.* (30) proposed this complexity measure based on the product of functions,

$$\mathcal{C}_{JS}[P] = \mathcal{H}[P]\, Q_{JS}[P, P_e], \tag{2.6}$$

where $\mathcal{H}[P] \in [0, 1]$ is the normalized Shannon Entropy, and Q_{JS} is the disequilibrium based on the Jensen-Shannon (JS) divergence. In this sense, Q_{JS} is expressed by

$$\begin{aligned} Q_{JS} &= Q_0 \mathcal{J}_S[P, P_e] \\ &= Q_0 \left\{ S\left[\frac{P + P_e}{2}\right] - \left[\frac{S[P] + S[P_e]}{2}\right] \right\}, \end{aligned} \tag{2.7}$$

Where Q_0 is a normalizing constant, while $\mathcal{J}_S$ is the JS divergence to quantify the difference between probability distributions. The presence of correlation structure is quantified in the SC (32), which measures time series complexity. In the case where the signal from the dynamical system is ultimately ordered or completely random, the value of $\mathcal{C}_{JS}[P]$ is the same as `null`,

i.e., the signal has no structure. In between these two extremes, dynamic systems can perform every possible level of physical form. These phases are reflected in the obtained features of the Probability Density Function (PDF) and quantified by no-null $\mathcal{C}_{JS}[P]$. The global property of SC is that its value does not change with different PDF layouts. Thus, $\mathcal{C}_{JS}[P]$ quantifies disorder but also the degree of correlated structure.

It has been shown that there are limit curves for complexity: for a given value of $\mathcal{H}$ and any data set, the possible $\mathcal{C}$ values vary between a minimum $\mathcal{C}_{min}(\mathcal{H})$ and a maximum $\mathcal{C}_{max}(\mathcal{H})$, restricting the possible values of the complexity measure (2, 33)

2.2.4 Fisher Information

We used the Fisher Information (FI) to analyze local aspects of changes in the information content given by a time series. It has different interpretations and calculations; among other things, the amount of information extracted from a process measures the ability to estimate parameters or the disordered state of a system or phenomenon (32). We define it as

$$FI[P] = F_0 \sum_{i=1}^{N-1} (\sqrt[2]{p_{i+1}} - \sqrt[2]{p_i})^2, \tag{2.8}$$

where F_0 is a normalization constant defined by

$$F_0 = \begin{cases} 1 & \text{if } p_{i^*} = 1 \text{ for } i^* = 1 \text{ or } i^* = N \\ & \text{and } p_i = 0 \forall i \neq i^* \\ 1/2 & \text{otherwise.} \end{cases} \tag{2.9}$$

According to Olivares *et al.* (32), the local sensitivity of FI to discrete PDFs requires the order of i of discrete values in $P = \{p_i : i = 1, \ldots, N\}$ when summing from Eq. 2.8. It is the distance between two related probabilities. Therefore, different orders will result in different FI values and, thus, their local nature.

2.2.5 Causal information planes

To characterize a given dynamical system described by a time series, we are able to use two representation spaces: (a) one with global-global characteristics called *causality entropy-complexity plane* ($\mathcal{H}x\mathcal{C}$) and (b) one with global-local characteristics called *causality Shannon-Fisher plane* (FxS), respectively (32).

The variation range of the causality entropy–complexity plane is $[0,1] \times [C_{LS}^{min}, C_{LS}^{max}]$, where C_{LS}^{min} and C_{LS}^{max} are the minimum and maximum values of Statistical Complexity. The values of C_{LS}^{min} and C_{LS}^{max} can be calculated through a geometric analysis of the probability space and depend only on the embedded dimension D (34). For causality Shannon-Fisher plane the range

is presumably [0, 1] × [0, 1]; no limit curves have been shown to exist so far (2).

As described in (35) and (2), chaotic systems have intermediate entropies $\mathcal{H}$ and F (between 0.45 and 0.7), and complexity $\mathcal{C}$ close to maximum. For regular processes, $\mathcal{H}$ and $\mathcal{C}$ have small values, close to zero, while the Fisher information is close to one. Uncorrelated stochastic processes (as white noise) have $\mathcal{H}$ near one, $\mathcal{C}$ near zero and F near zero too, and colored noises are in the middle, with medium $\mathcal{C}$ and medium-high $\mathcal{H}$ values, and medium-low F values; Regular oscillations have low $\mathcal{H}$ and $\mathcal{C}$. Figures 2.1 and 2.2 show examples of well-known series in these planes.

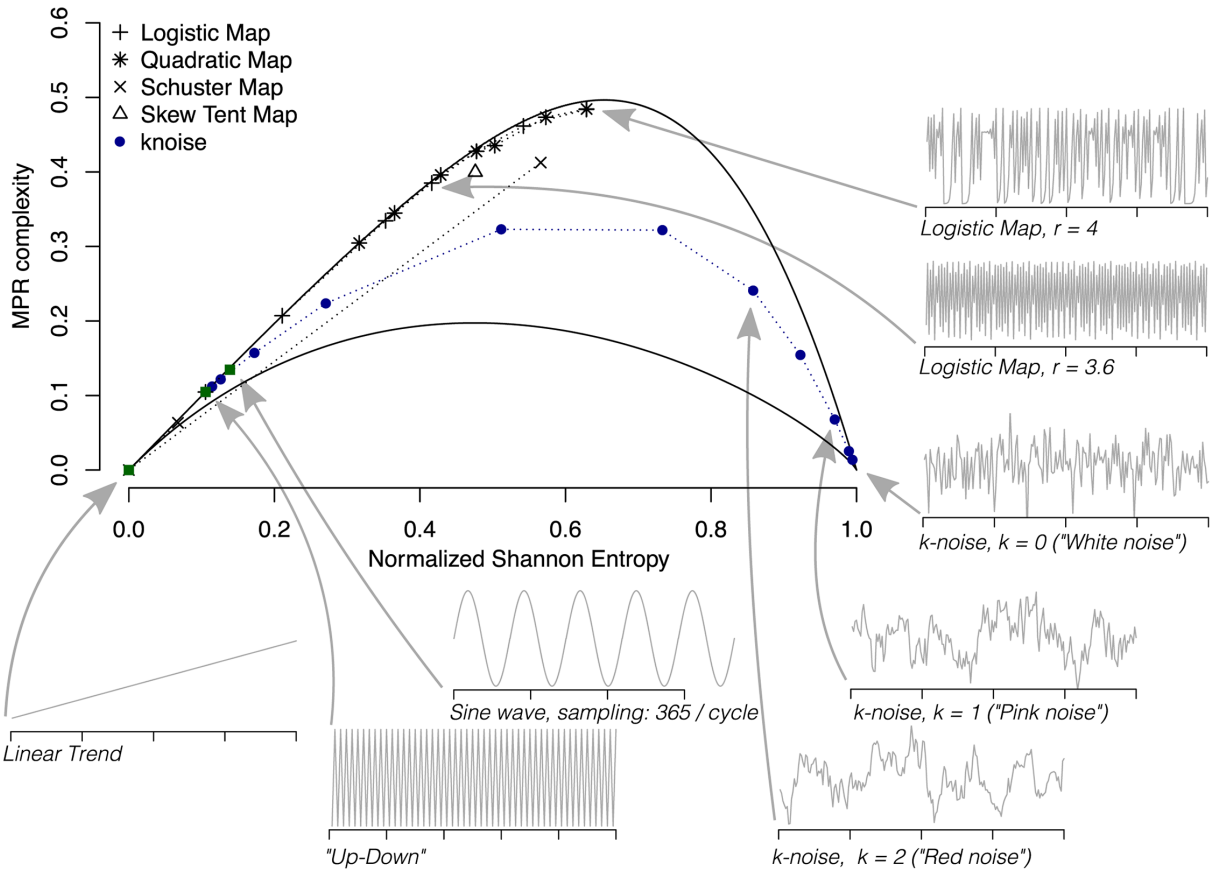


Figure 2.1: Position of series in the Complexity-Entropy plane (2).

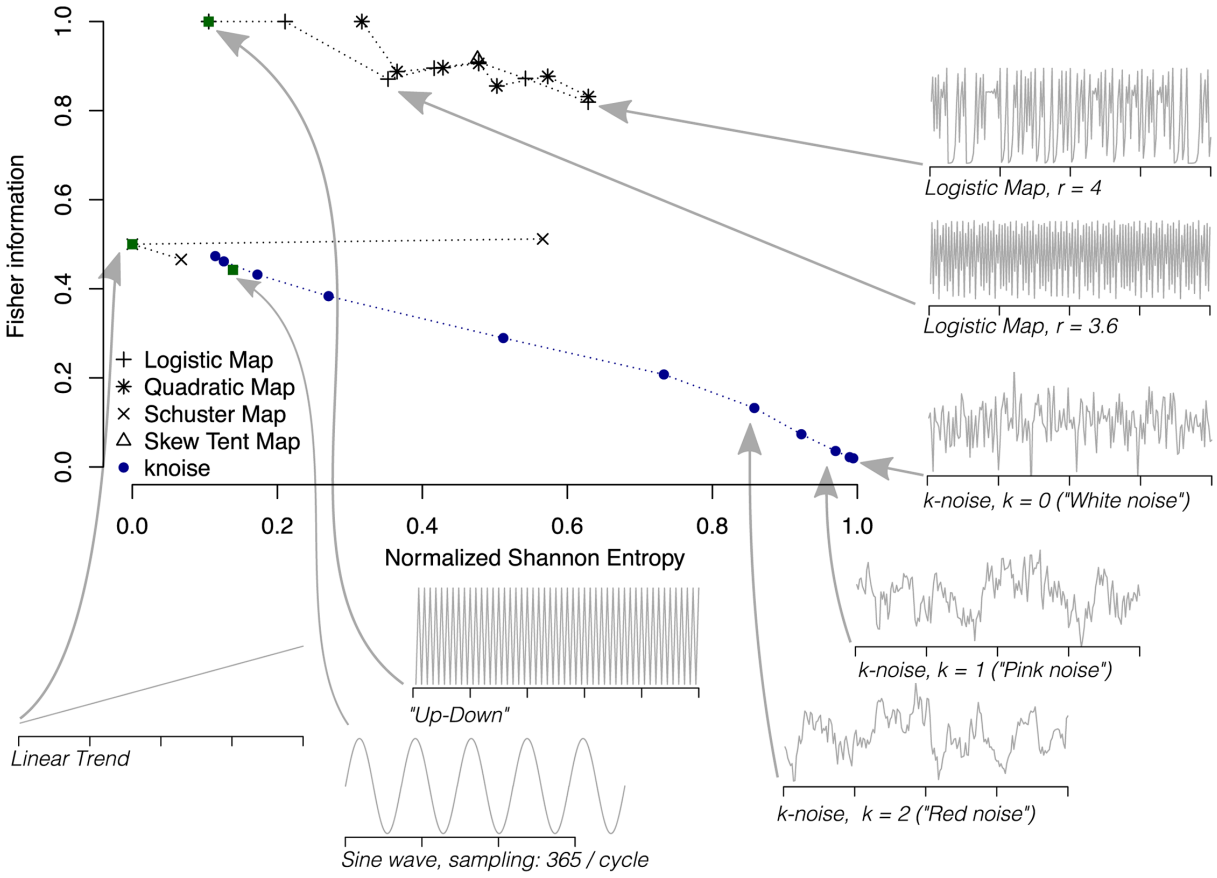


Figure 2.2: Position of series in the Fisher-Shannon plane (2).

3

Methodology and Proposal

As mentioned previously, most of the observed studies seek to train a model for classification following the steps: data extraction, cleaning and transformation data, training model, and evaluation model. Therefore, we propose to include measures from Information Theory in that process.

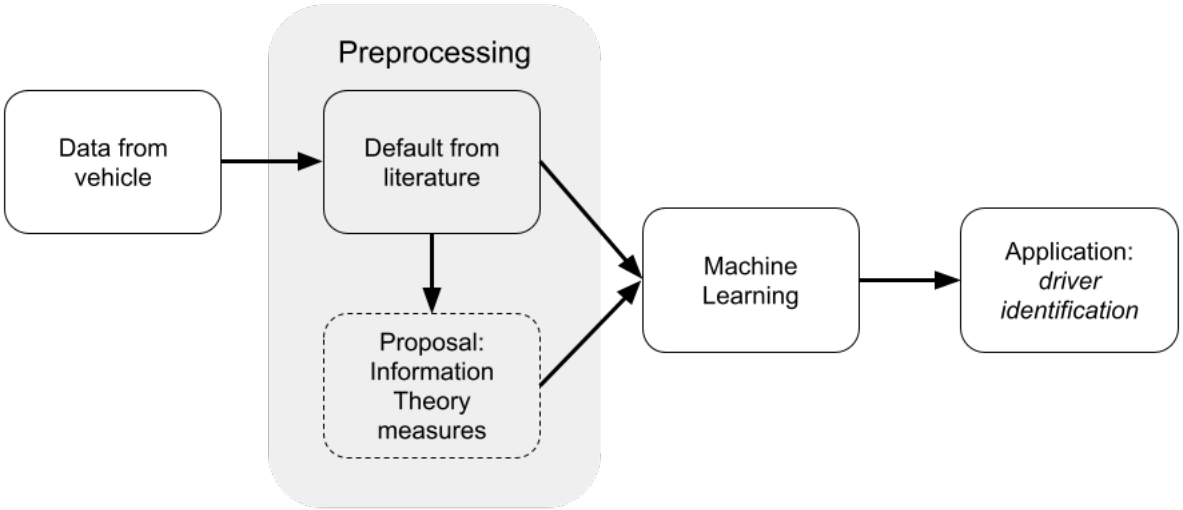


Figure 3.1: Proposed process.

First, data from the vehicle's sensors are obtained and then preprocessed according to the literature. In this step, we remove some features and instances with missing data. We normalize the remaining data between zero and one.

Then, we transform the preprocessed data into Information Theory measurements. Each feature is treated as a time series at this stage and converted into a series of ordinals according to Bandt-Pome (36). Note that data normalization does not interfere with this transformation, because the Bandt-Pompe ordinal patterns are calculated based on the relative order of values

in a time series, rather than on the absolute magnitude of the values. Based on the series of ordinals, Information Theory measures are calculated (Shannon Entropy (37), Fisher Information (38), and Complexity Statistic (30)).

Finally, the data is applied to train and execute the machine learning model. A summary of the steps is listed below. Numbers 1, 2, 4, and 5 are standard in the literature, but we propose adding step 3.

1. Extraction of driving data from a vehicle via OBD-II.
2. Preprocessing is the process of cleaning and transforming the data before being applied to training or running the model to improve its performance; therefore, this process varies depending on the algorithm used in the model.
3. Extracting Information Theory measures from preprocessed data to apply to machine learning model (our proposal).
4. Training Machine Learning model for classification.
5. Run the trained machine learning model to classify new data.

3.1 Vehicle Data

We obtain vehicle data from the database in (8). This database is composed of the data generated during the driving of four drivers who took the same route and vehicle, extracting 51 features (each one related to a sensor). For each trip, we have a file with the data, and the files of each driver were separated into a folder as shown in the Figure 3.2.

Although there are few drivers in this dataset, it does not contain missing values and has a relatively large amount of data for each driver. Drivers A and B made eight trips, C made five, and D made nine. Each trip generated a data file, and each file contains between 1444 and 2296 lines of data.

3.2 Preprocessing and Calculation of Information Theory measures

To allow a fair comparison with the proposed technique, we consider the preprocessing commonly used in the literature (13), such as: remove feature invariant, highly correlated features and normalization. Our proposal adds a new step to generate a new dataset containing information theory measures.

The dataset samples comprised the data from the first trip of each driver. Subsequent feature selection steps were then performed on this sample. The preprocessing steps performed in this work were as follows:

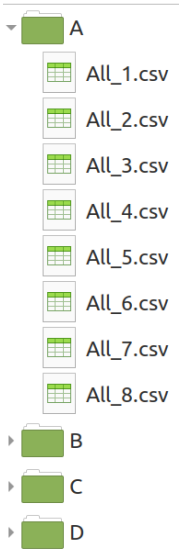


Figure 3.2: Dataset folders

1. Get data from dataset raw: there are a total of 51 features.

2. Preprocessing from literature

   (a) Feature indifferent: Analyzing the data, we observe that the value of some characteristics did not change for each driver. We identify a set of features whose value did not vary over time for each driver. At this stage, 11 features were eliminated and 40 remained.

   (b) Feature invariant: These are features whose value remains unchanged in all instances, that is, they have zero variance. At this stage, 3 features were eliminated and 37 remained.

   (c) Exclusion of highly correlated Features: After removing invariant features, we observe that, among the remaining features, some are similar. They have a high level of correlation. Figure 3.3 visualizes the correlation matrix of the 37 features. Most researchers agree that a coefficient below 0.1 indicates an insignificant relationship, while above 0.9 indicates a very strong relationship (39, 40). Thus, features with a correlation greater than 0.9 were removed, and 25 features remained.

   (d) Normalization: To cancel out the effect of different sensor measurement scales, we standardize the dataset using (1)

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}}. \tag{3.1}$$

   The normalization step will transform data to be ranged between 0 and 1 and thus treated equally by the classification algorithms (25). We use the minimum and maximum, considering all data from all drivers.

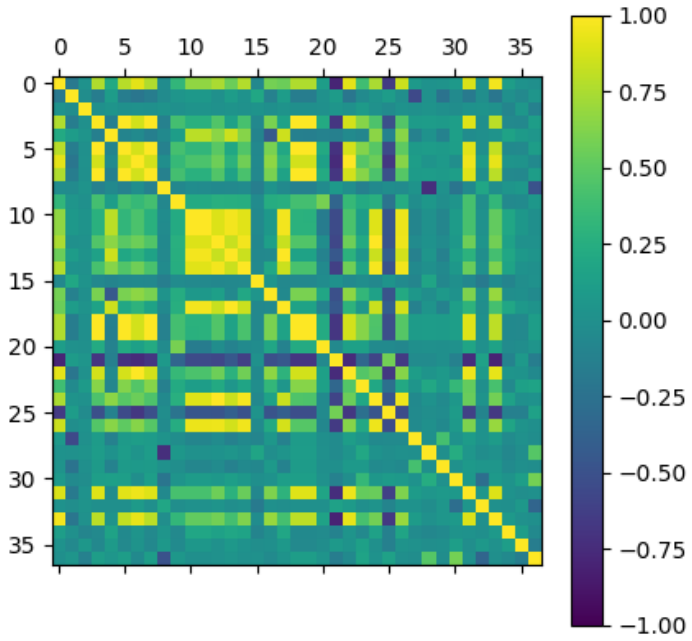


Figure 3.3: Correlation among features.

3. Information theory measures

   (a) From each preprocessed series, several subseries are generated using sliding windows that slide to each sample (Figure 3.4). To obtain the maximum amount of data, we make the maximum overlap between the windows, that is, we move the window by one sample. Each subseries applies the Bandt and Pompe symbolization approach to obtain a sequence of ordinal patterns (41). Then, for each sequence of ordinals, the Information Theory measures are calculated, namely Shannon Entropy, Statistical Complexity, and Fisher Information.

   (b) We remove duplicate sequences from each subseries, and if the remaining sequence is smaller than the embedding dimension D, the information measures take on a null value (NaN). The size of the embedding dimension D is explained in Section 4.2.

   (c) We remove features that contain any null values; 10 features were removed, as seen in Table 3.1.

Table 3.2 shows the obtained, generated and remaining features at each step, while the Table 3.1 shows the features removed in each step. Step 2 reduces the number of features to 31, and from these features, Shannon Entropy, Statistical Complexity, and Fisher Information measures are calculated. However, please note that not all features from Step 2 are used to get Information Theory measures. This is because after Step 3b, some features have missing values and these features were removed in Step 3c.

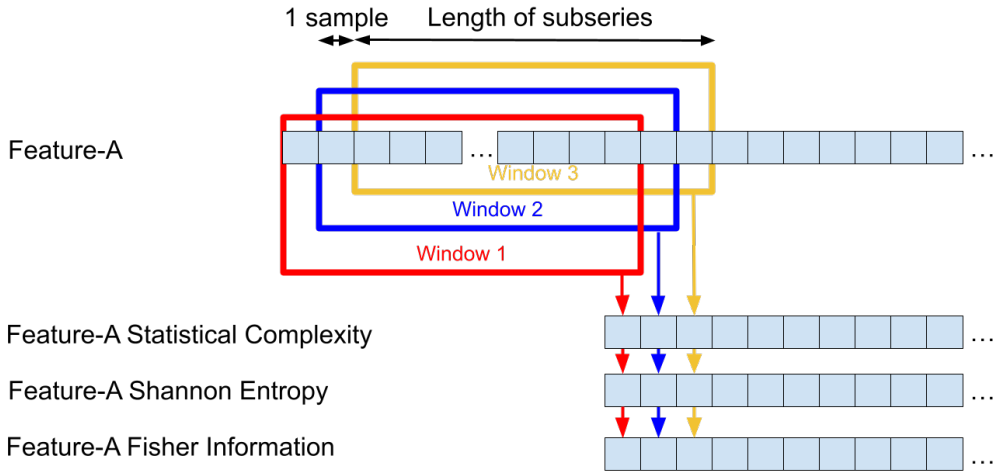


Figure 3.4: Information Theory computation.

Table 3.1: Features removed.

Step	Name of removed features	#
2a	Latitude acceleration, glow plug limit request, brake cylinder, fuel pressure, flywheel torque revised, accelerator position filtered, clutch check, flywheel torque, longitude acceleration, torque transform coefficient, and fire angle delay tcu.	11
2b	Engine pressure maintenance time, target engine velocity lockup, and compressor activation	3
2c	Wheel velocity back right, throttle position, wheel velocity front right, engine torque max, wheel velocity front left, engine torque, engine torque revised, accelerator position, torque converter turbine speed, calculation overhead, car speed, and throttle position abs	12
3c	Reduce block fuel, gear choice, engine velocity increase tcu, mission oil temp, block fuel, and standard torque ratio.	6

3.3 Machine Learning

We used seven different classification models to compare our proposal with the literature. These are non-Deep Learning classifiers commonly used in the literature. In each of them, we apply preprocessed data according to the literature and then the preprocessed data according to our proposal. We describe the classification algorithms used in the following subsections.

Table 3.3 shows the parameters of each algorithm. We do not intend to optimize the parameters for these algorithms. Thus, we use the default values from the library (42), except for Nearest Neighbors (kNN). In this case, we use $k = floor(\sqrt{N})$ (where the floor function returns the largest integer less than or equal). N is the window size and is 120. This neighbor's value is commonly used and linked to the sample size. For Multi-Layer Perceptron (MLP), we use the iteration limiter equal to 1000 to guarantee an acceptable training time for the solution.

Table 3.2: Features remaining.

Step	Name of remaining features	#
1	Fuel usage, accelerator position, throttle position, short fuel bank, inhale pressure, accelerator position filtered, throttle position abs, engine pressure maintenance time, reduce block fuel, block fuel, fuel pressure, long fuel bank, engine speed, engine torque revised, friction torque, flywheel torque revised, current fire timing, cooling temperature, engine idle slippage, engine torque, calculation overhead, engine torque min, engine torque max, flywheel torque, torque transform coeff, standard torque ratio, fire angle delay tcu, engine torque limit tcu, engine velocity increase tcu, target engine velocity lockup, glow plug limit request, compressor activation, torque converter speed, current gear, mission oil temp, wheel velo front left, wheel velo backright, wheel velo frontright, wheel velo backleft, torque converter turbin speed, clutch check, converter clutch, gear choice, car speed, logitude acceleration, brake switch, brake sylinder, road slope, latitude acceleration, steering wheel acceleration, and steering wheel angle.	51
2	Long fuel bank, torque converter speed, reduce block fuel, short fuel bank, steering wheel angle, cooling temperature, engine torque limit tcu, fuel usage, brake switch, engine speed, standard torque ratio, engine torque min, mission oil temp, road slope, steering wheel acceleration, engine idle slippage, wheel velocity back left, inhale pressure, block fuel, gear choice, current fire timing, engine velocity increase tcu, friction torque, current gear, and converter clutch	25
3	*Statistical Complexity, Shannon Entropy, and Fisher Information of*: Long fuel bank, short fuel bank, steering wheel angle, cooling temperature, engine torque limit tcu, friction torque, fuel usage, brake switch, engine speed, engine torque min, road slope, steering wheel acceleration, engine idle slippage, wheel velocity back left, inhale pressure, current fire timing, torque converter speed, current gear, and converter clutch	57

3.3.1 k-Nearest Neighbors

K-Nearest Neighbors (kNN) can be considered a voting system, where the majority class label determines the class label of a new data point among its nearest k (where k is an integer) neighbors in the feature space (43). Neighbors-based classification is instance-based or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores training data instances (42).

The number of neighbors (k) is a parameter of this algorithm. Adjusting to a higher value of k can suppress noise effects. However, it makes the classification limits less distinct. That is, the classifier is more susceptible to errors at the boundaries between the classes.

We can adjust the classifier regarding the weights assigned to each instance. With uniform weights, each of the k nearest neighbors has the same contribution. With weighted weights, the

Table 3.3: Parameters of the classification algorithms used.

Classifier	Parameter
K-Neighbors	n_neighbors=$floor(sqrt(120))$, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski'
SVC linear	C=1.0, kernel='linear', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False
SVC	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False
Decision Tree	criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, min_impurity_decrease=0.0, ccp_alpha=0.0
Random Forest	n_estimators=100, criterion='gini', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', min_impurity_decrease=0.0, bootstrap=True, oob_score=False, verbose=0, warm_start=False, ccp_alpha=0.0
MLP	hidden_layer_sizes=100, activation='relu', *, solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, tol=0.0001, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10, max_fun=15000
Gaussian NB	var_smoothing=1e-09

closest neighbors contribute more to the adjustment. The weights are proportional to the inverse of the distance from the query point.

3.3.2 Support Vector Machine

In 1992, Vapnik and coworkers proposed a supervised algorithm for classification that has since evolved into what is now known as Support Vector Machines (SVMs) (44). Its difference is that the algorithm seeks to separate the classes by a hyperplane, maximizing its margin and the distance between the hyperplane and the data.

We use two kernels in this work: linear and radial. SVM with the linear kernel separate data points using a linear hyperplane with the most significant margin (45). SVM with Radial Basis Function kernel (RBF SVM) separates data points using a higher dimensional space. A higher value of the gamma parameter will perfectly fit the training dataset, which causes over-fitting.

3.3.3 Tree algorithms

Decision Tree (DT) is a non-parametric supervised learning method. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The deeper the tree, the more complex the decision rules and the fitter the model (46). In other words, A decision tree is a predictor, $h : X \rightarrow Y$, that predicts the label associated with an instance x by traveling from a root node of a tree to a leaf, that is, the nodes contain the decision rules and the leaves contain the classes of interest (47).

The advantages of this classifier are that it can work with both numerical and categorical data and is easy to understand and interpret. Trees can be visualized. Decision tree-based classifiers may need to generalize classifications better depending on the problem and data, in addition to dealing with the imbalance in the amount of data in each class.

Decision trees are built by adding nodes with rules to separate the data. The rules that most clearly separate the data are chosen; we try to make rules that best separate the classes.

Random Forest (RF) is a meta-estimator that fits several decision tree classifiers on various sub-samples of the dataset and then takes the most popular result (48).

3.3.4 Multi Layer Perceptron

A Multi-Layer Perceptron (MLP) is based on Perceptron, also called a neuron, and it can have one or more layers of neurons between the input and output layers (3) as shown in figure 3.5. The layers are entirely connected, and learning occurs when the weights of these connections change.

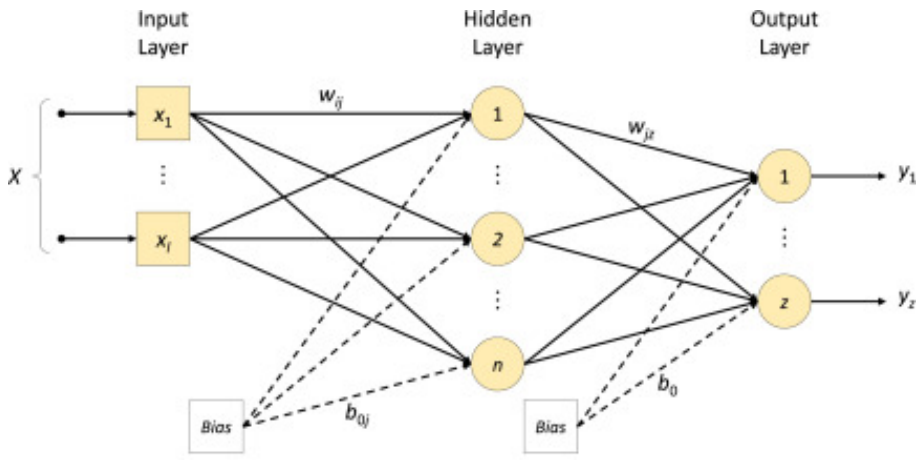


Figure 3.5: Structure of MLP with a single hidden layer (3).

Given a set of features and a target, an MLP can retain a non-linear function approximator for either classification or regression (49).

3.3.5 Naive Bayes

Widely discussed in (50), this method uses a probabilistic approach based on Bayes' Theorem. The "naive" assumption of conditional independence between every pair of features given the value of the class variable (51).

Given an element $x = x_1, x_2, \ldots, x_n$ with n attributes, Naive Bayes predicts the class C_k for x using the probability calculated by Bayes' Theorem,

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)} = \frac{p(x_1, x_2, \ldots, x_n|C_k)p(C_k)}{p(x_1, x_2, \ldots, x_n)} \tag{3.2}$$

where C is the set of k possible classes. After calculating the probabilities, the C_k class with the highest probability is assigned to element x.

4

Results and Discussion

This chapter is organized as follows: Section 4.1 describes the metrics used, Section 4.2 outlines how the Information Theory measures employed in this work were extracted and organized, and Section 4.3 details how the models were evaluated and presents the results comparing the methods.

4.1 Metrics used

We also use accuracy, precision, and recall measures. To understand these measures, we suppose a binary classification task with a positive class C_+ and a negative class C_-, the confusion matrix or contingency table counts correctly and incorrectly classified samples in a decided manner, see Table 4.1.

So we define the precision as the fraction of predicted positives to be actual positives (52). This metric shows the correct proportion of identifications, also called True Positive (TP). Considering our case of owner-driver identification, we want to know what proportion of data is actually from the owner-driver out of all the data that the model labeled as owner-driver data.

$$Precision = \frac{TP}{(TP+FP)}. \tag{4.1}$$

Table 4.1: Binary Confusion Matrix: True Positives (TP), False Positives (FP), True Negatives (TN), False Negatives (FN).

		True class	
		C_+	C_-
Predicted	C_+	TP	FP
	C_-	FN	TN

The accuracy used in the experiments is as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{correct\ classification}{all\ classification}. \tag{4.2}$$

where TP is number of true positives (correctly identified owner driver data), TN is the number of true negatives (correctly identified thief driver data), FP is the number of false positives (incorrectly identified owner driver data), and FN is the number of false negatives (incorrectly identified thief driver data) (25). Considering our case, recall correctly identifies the owner-driver data among all the owner-driver data.

$$Recall = \frac{TP}{(TP+FN)}. \tag{4.3}$$

We also used the Area Under Curve (AUC) value of the Receiver Operating Characteristic (ROC) Curve. This metric evaluates both the sensitivity (recall) and specificity. Specificity refers to the test's ability to correctly reject data that does not belong to the target driver (owner):

$$Specificity = \frac{TN}{(TN+FP)}. \tag{4.4}$$

4.2 Information Theory measures extraction

To obtain Information Theory measures that improve the performance of a classifier, we vary two parameters: the series size and the value of D. The value of D is the embedding dimension on the horizontal axis. It means the number of ordinals that form each symbol. For example, for D equal to 3, we have the ordinals 1, 2, and 3. With these ordinals, we can form six symbols: $(1,2,3)$, $(1,3,2)$, $(2,1,3)$, $(2,3,1)$, $(3,1,2)$, and $(3,2,1)$. Thus, the larger the D, the more information we can obtain from the series; however, if the number of symbols is huge for the size of the series, it may be that many symbols do not appear in the series. Therefore, we choose the smallest value for D so that $D!$ equals or exceeds the series size.

First, we carried out a visual analysis of some features in the Complexity-Entropy and Fisher-Shannon planes to provide evidence of the feasibility of using these measures. We then evaluate the best parameters to calculate the measurements. Finally, we use the measurements to train the models and compare them with models trained with preprocessed data according to the literature.

We observed the positioning of Information Theory measures of some characteristics in the Complexity-Entropy planes and in the Fisher-Shannon plane, so we sought to see if these measures could help in training machine learning models. We use each driver trip to generate a point in the plane, drivers A, B, C, and D. In this scenario, the length of the series corresponds to the

number of instances in the file representing each trip. With the files containing between 1444 and 2296 lines of data, we selected an embedded dimension value of 7 ($D = 7$). This choice ensures that the factorial of 7 is greater than the maximum number of lines in the files.

Figures 4.1 and 4.2 are examples of the observed planes, showing only the sensors: accelerator position, steering wheel angle, car speed, and cooling temperature. In Figures 4.1, the solid lines from the Complexity-Entropy planes are the upper and lower limits of the Complexity value. The closer to the maximum limit, the more information is required to describe the series. In other words, the further it is from a uniform distribution. Figure 4.2 shows that the Fisher-Shannon planes, as the Complexity-Entropy planes, can reveal the particularities of each driver. Nevertheless Fisher Information unveils local characteristics, indicating the frequency of abrupt changes within the series, while also assessing the data's ordering.

Figures 4.1a and 4.1c have greater Entropy, indicating that the drivers' behavior considering accelerator position and steering wheel angle represents a chaotic system with a medium-high entropy and an high complexity. The accelerator position behavior highlights driver D. The steering wheel angle behavior can distinguish the drivers. These measure values can change immediately with the driver's action.

However, Figures 4.1e and 4.1g show values with slightly medium-lower Entropy, as they come from sensors that are more indirectly affected by the driver's action. Thus, it is not easy to distinguish among the drivers evaluated. This behavior suggests that it is feasible to use a set of Information Theory measures from each sensor as new features for classification algorithms to distinguish between drivers.

To obtain several Information Theory ($\mathcal{HC}$) points, we take several subsequences from the series that represent each driver's trip. Then, we vary the size of this subsequence: 20, 120, 300, 720, 900. The sequence length is the number of instances and the time in seconds. We consider values below 20 instances with little information to be obtained and values above 900 seconds as a very long time to start generating data in an application and would possibly generate little information measurement data; for example, the files used have between 1444 and 2296 rows of data which is between 1444 and 2296 seconds.

Our previous studies suggest that tree-based algorithms outperformed other methods for classifying drivers. Therefore, we employed a Random Forest classifier to investigate the parameter variation for Bandt-Pompe symbolization. Our aim is to determine the optimal number of samples required to achieve information measure values that enhance classifier performance.

Figure 4.3 depicts a more pronounced increase between 300 and 540, followed by a smaller increase between 540 and 840. It's noteworthy that the four largest sizes (720, 780, 840, and 900) exhibit the highest values but are statistically indistinguishable. In Figure 4.6, the preprocessing time of both the proposed method and the standard method from the literature is illustrated. It's evident that the computational cost significantly escalates beyond size 780. This escalation is attributed to the increase in the embedded dimension (D) from 6 to 7, given that the number of possible symbols for each D follows a factorial relationship. Finally, we used the

amount of 720 samples (equivalent to 720 seconds) in our evaluation.

4.3 Proposal evaluation

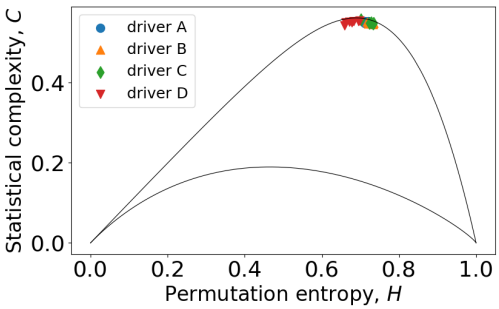
Driver C has five trips and makes the fewest trips, so we use the first 5 trips for each driver. Just as done by Girma et al. (1), we divided each driver's dataset into 120-second windows (120 samples) with a shift of 60 seconds, as shown in Figure 4.4. The overlapping window is important to smooth the flow of sequence, capture sequential information, and increase training data size for better generalization (1). So we formed a new dataset with each driver's windows. We apply a cross-validation with five folds in this dataset to analyze each classifier. We also made a binary distinction for each driver with the others.

Figure 4.5a shows the accuracy of each classifier in each method using the literature method and the proposed. We note that our approach improved the accuracy of all evaluated classifiers. The improvement may be because the algorithms evaluated in this work do not intrinsically consider the temporal characteristics of the data. Bandt-Pompe symbolization and the values of Information Theory used in this work add global, local and temporal information to the input of these algorithms.

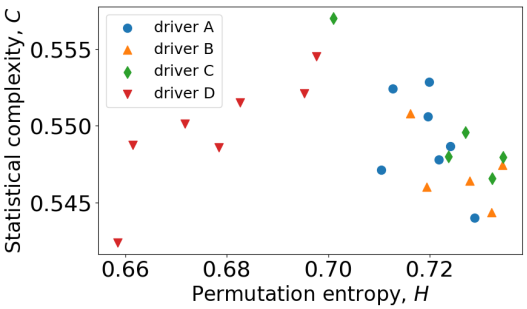
Due to the construction of the dataset, we have a minority of the data as being in the true class (owner-driver - 25%) and a majority in the false class (theft - 75%). To evaluate the sensitivity and specificity of the classifier, we observed Figure 4.5b that shows the value of the ROC AUC. The AUC value varies from 0.5 to 1.0. The closer to 1.0, the better the classifier. However, closer to 0.5 means it is effectively not classifying, as it is similar to a random classification. The AUC values show that in our proposal, the classifiers perform better in finding true cases (owner) and false ones (theft).

We further assessed the classifiers based on precision and recall to obtain a more comprehensive qualitative perspective. Our observations revealed that incorporating Information Theory measures significantly enhanced the performance of all classifiers across the four metrics employed. Notably, the KNN, RBF SVM, and Random Forest models achieved scores very close to the maximum across all evaluated metrics.

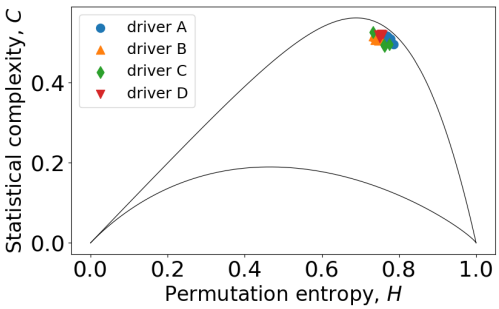
According to the literature, another metric we look at is that the total time for calculating Information Theory measures and training is longer in the proposed method than in preprocessing. Figure 4.6 shows that to generate the Bandt-Pompe symbolization and calculate the information measures (Statistical Complexity, Shannon Entropy and Fisher Information) the time was significantly longer than using the standard preprocessing in the literature. However, the time to maximum for the sample size we used (720 samples) does not exceed 30 milliseconds. The high difference is because the other strategies are high-speed. However, based on the requirements of driver detection applications, this execution time still preserves the effectiveness of our proposal.



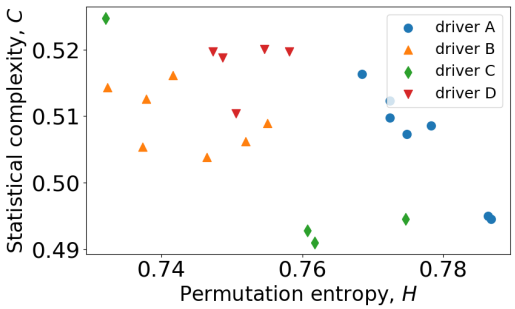
(a) Accelerator position



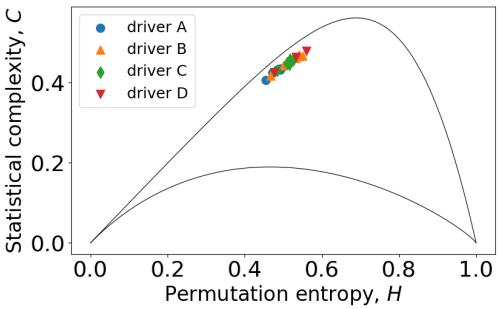
(b) Zoom of Accelerator position



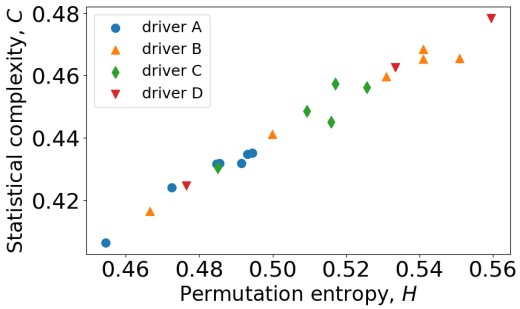
(c) Steering wheel angle



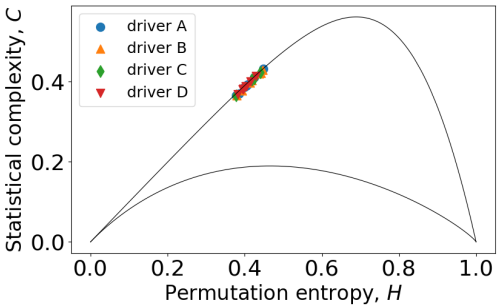
(d) Zoom of Steering wheel angle



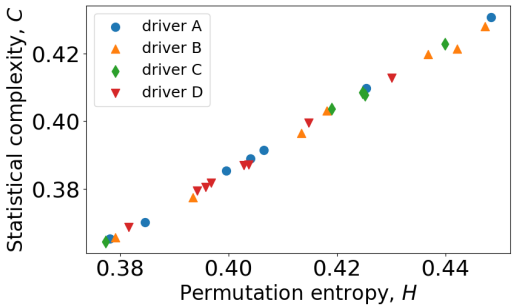
(e) Car speed



(f) Zoom of Car speed

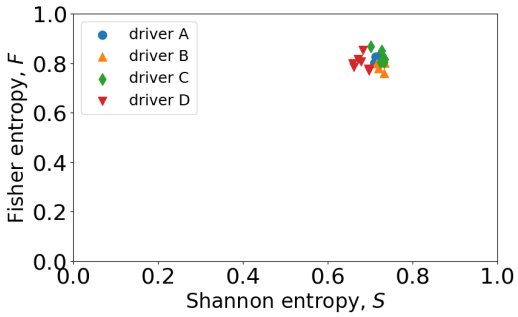


(g) Cooling temperature

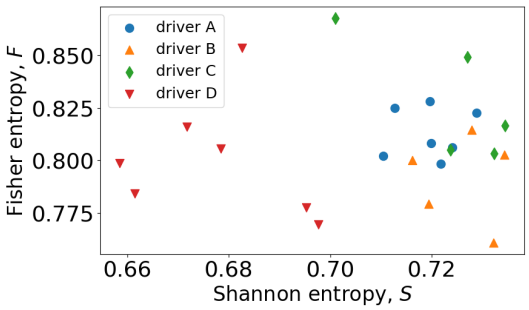


(h) Zoom of Cooling temperature

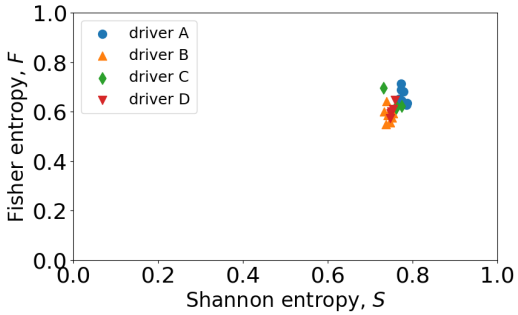
Figure 4.1: Complexity-Entropy planes for each trip with $D = 7$.



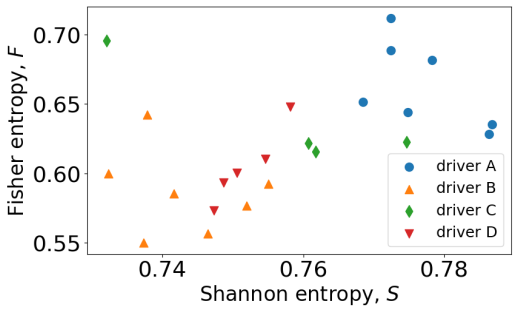
(a) Accelerator position



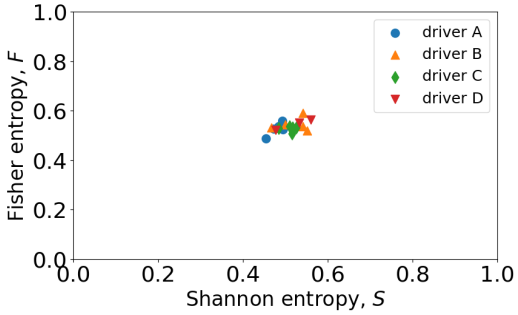
(b) Zoom of Accelerator position



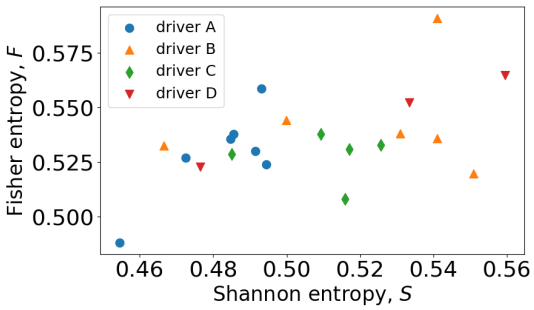
(c) Steering wheel angle



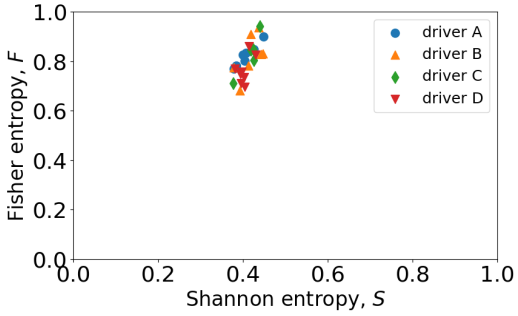
(d) Zoom of Steering wheel angle



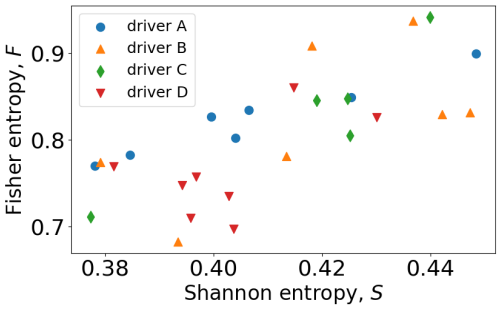
(e) Car speed



(f) Zoom of Car speed



(g) Cooling temperature



(h) Zoom of Cooling temperature

Figure 4.2: Fisher-Shannon planes for each trip with $D = 7$.

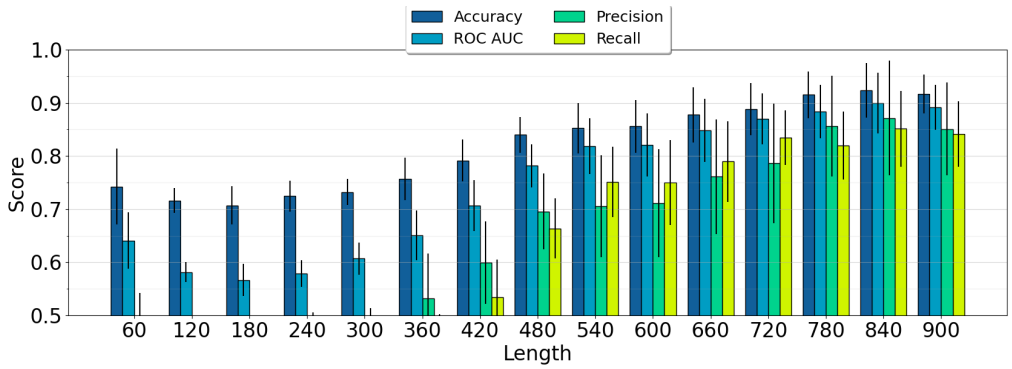


Figure 4.3: Measurements for different parameters in the Complexity and Entropy calculation.

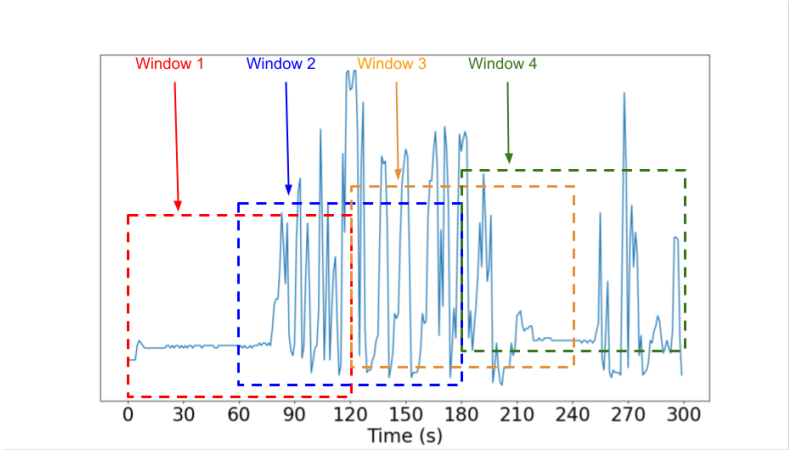
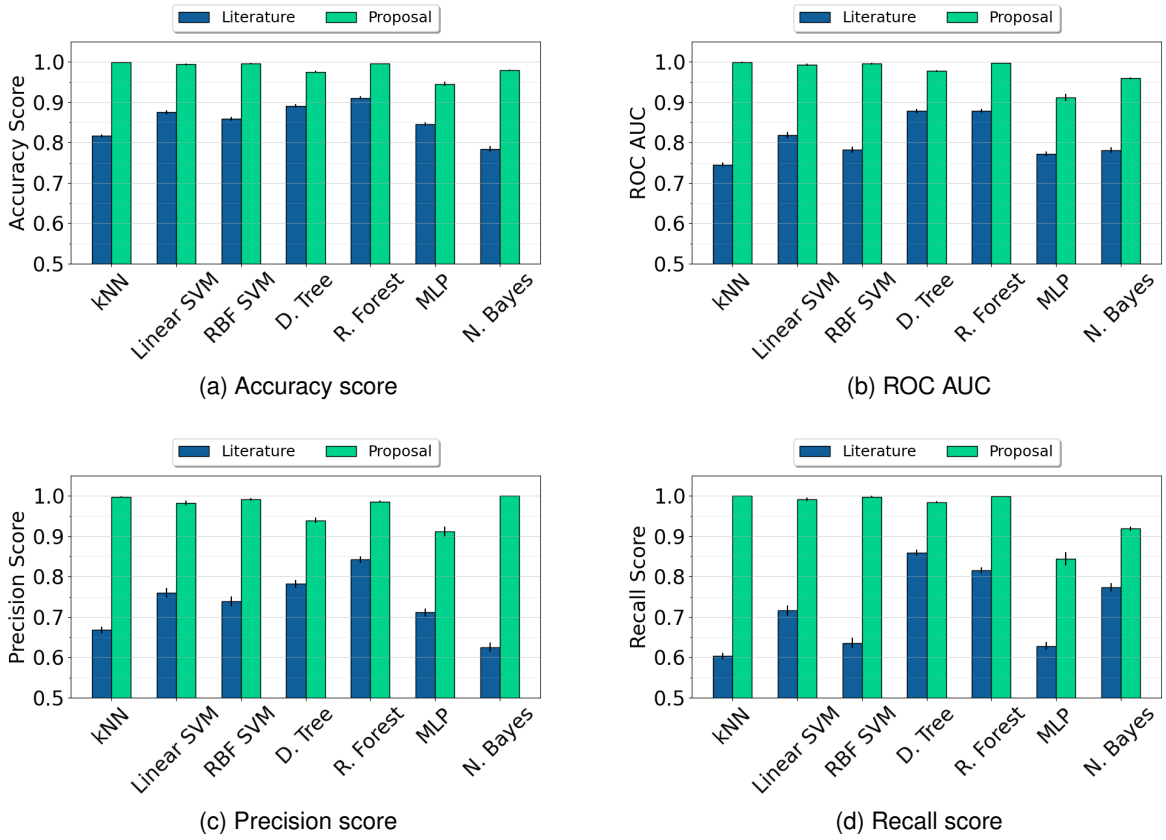


Figure 4.4: Sliding windows on features.

The working environment of the experiment was as follows:

- CPU 11th Gen Intel Core i7-1165G7 @ 2.80GHz × 4.
- GPU Intel Corporation TigerLake-LP GT2 [Iris Xe Graphics].
- RAM 16GB.



(a) Accuracy score

(b) ROC AUC

(c) Precision score

(d) Recall score

Figure 4.5: Experiment results.

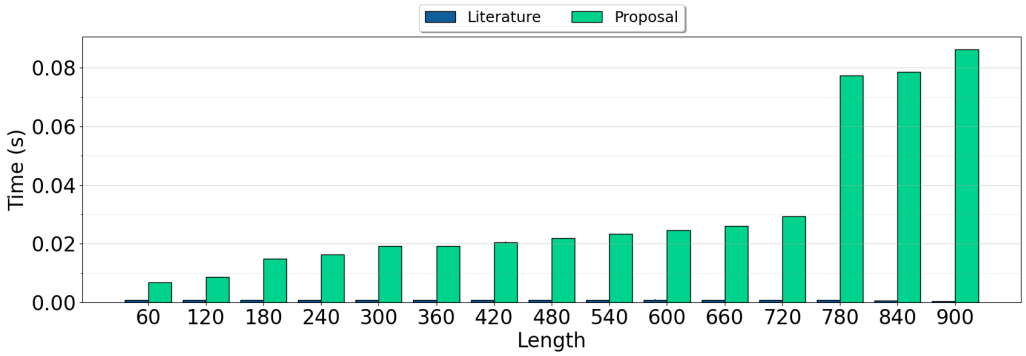


Figure 4.6: Time for preprocessing.

5

Final Considerations

This work aims to improve the performance of machine learning models by adding measures from Information Theory to time series data in the particular case of automotive driving data.

It was possible to compare the preprocessing described in the literature with a method that extracts measures from Information Theory for driver identification. We have shown that the proposal improved the performance of all classifiers analyzed for the parameters considered to calculate the Statistical Complexity, Permutation Entropy, and Fisher Information measures.

We observed that, with Information Theory measures, the models can identify a driver with an accuracy above 94.5% for the classifiers used, and some classifiers getting scores very close to maximum. In addition, some algorithms also achieved scores very close to the maximum for presicion, recal and ROC AUC metrics.

It is essential to consider that the total time for calculating Information Theory measures and training is longer in the proposed method, but the case of greater processing considered in this work is equivalent to a maximum of 30 ms for the environment used. We also intend to apply the proposed method to other datasets for driver identification and use it in other classification applications.

Additionally, we aim to assess the models using datasets featuring various class balances, incorporate additional public datasets, and evaluate deep learning algorithms. Finally, as future work, we may include the following topics: optimization of classifier hyperparameters, utilization of feature selection techniques, and incorporation of other measures: False Acceptance Rate (FAR) and False Rejection Rate (FRR).

Bibliography

1 GIRMA, A.; YAN, X.; HOMAIFAR, A. Driver identification based on vehicle telematics data using lstm-recurrent neural network. In: **2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)**. [S.l.: s.n.], 2019. p. 894–902.

2 SIPPEL, S. et al. Diagnosing the dynamics of observed and simulated ecosystem gross primary productivity with time causal information theory quantifiers. **PLOS ONE**, v. 11, n. 10, p. 1–29, 2016.

3 PARMEZAN, A. R. S.; SOUZA, V. M. A.; BATISTA, G. E. Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. **Information Sciences**, v. 484, p. 302–337, 2019. ISSN 0020-0255.

4 MANDERNA, A.; KUMAR, S. Effective long short-term memory based-driver identification in its. In: **2022 International Conference on Inventive Computation Technologies (ICICT)**. [S.l.: s.n.], 2022.

5 SINGH, M.; DUBEY, R. K. Deep learning model based co2 emissions prediction using vehicle telematics sensors data. **IEEE Transactions on Intelligent Vehicles**, 2023.

6 BARRETO, C. A. S.; JúNIOR, J. C. X. Machine learning using for car usage pattern identification based on car data. **Repositório Intitucional UFRN**, 2018.

7 BERNARDI, M. L. et al. Driver and path detection through time-series classification. **Journal of Advanced Transportation**, v. 2018, p. 1758731, 2018.

8 PARK, K. H.; KWAK, B. I.; KIM, H. K. **This Car is Mine!: Driver Pattern Dataset extracted from CAN-bus**. [S.l.], 2020.

9 WEBSITE, S. of Automotive Engineers official. **SAE J1979**. 2017.

10 WAKITA, T. et al. Driver identification using driving behavior signals. In: **Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005.** [S.l.: s.n.], 2005.

11 MENG, X.; LEE, K. K.; XU, Y. Human driving behavior recognition based on hidden markov models. In: **2006 IEEE International Conference on Robotics and Biomimetics**. [S.l.: s.n.], 2006.

12 MIYAJIMA, C. et al. Driver modeling based on driving behavior and its evaluation in driver identification. **Proceedings of the IEEE**, v. 95, n. 2, p. 427–437, 2007.

13 PARK, K. H.; KIM, H. K. This car is mine!: Automobile theft countermeasure leveraging driver identification with generative adversarial networks. **CoRR**, abs/1911.09870, 2019.

14 MARTINELLI, F. et al. Who's driving my car? a machine learning based approach to driver identification. In: **Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP**. [S.l.: s.n.], 2018. p. 367–372. ISBN 978-989-758-282-0. ISSN 2184-4356.

15 SCHNEEGASS, S. et al. A data set of real world driving to assess driver workload. In: **Proceedings of the 5th International Conference on Automotive User Interfaces and Interactive Vehicular Applications**. New York, NY, USA: [s.n.], 2013. p. 150–157. ISBN 9781450324786.

16 WU, J. D.; YE, S. H. Driver identification using finger-vein patterns with radon transform and neural network. **Expert Syst. Appl.**, v. 36, p. 5793–5799, 2009.

17 WAHAB, A. et al. Driving profile modeling and recognition based on soft computing approach. **IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council**, v. 20, p. 563–82, 2009.

18 MARTINELLI, F. et al. Human behavior characterization for driving style recognition in vehicle system. **Computers & Electrical Engineering**, v. 83, 2020.

19 SINGH, M.; DUBEY, R. K. Deep learning model based co2 emissions prediction using vehicle telematics sensors data. **IEEE Transactions on Intelligent Vehicles**, v. 8, n. 1, p. 768–777, 2023.

20 VAITI, T. et al. Traffic emissions clustering using obd-ii dataset based on machine learning algorithms. **Transportation Research Procedia**, v. 64, p. 364–371, 2022.

21 UVAROV, K.; PONOMAREV, A. Driver identification with obd-ii public data. In: **2021 28th Conference of Open Innovations Association (FRUCT)**. [S.l.: s.n.], 2021. p. 495–501.

22 XUN, Y.; SUN, Y.; LIU, J. An experimental study towards driver identification for intelligent and connected vehicles. In: **ICC 2019 - 2019 IEEE International Conference on Communications (ICC)**. [S.l.: s.n.], 2019.

23 EZZINI, S.; BERRADA, I.; GHOGHO, M. Who is behind the wheel? driver identification and fingerprinting. **Journal of Big Data**, v. 5, 2018.

24 KWAK, B. I.; WOO, J.; KIM, H. K. Know your master: Driver profiling-based anti-theft method. In: **2016 14th Annual Conference on Privacy, Security and Trust (PST)**. [S.l.: s.n.], 2016. p. 211–218.

25 MEKKI, A. E.; BOUHOUTE, A.; BERRADA, I. Improving driver identification for the next-generation of in-vehicle software systems. **IEEE Transactions on Vehicular Technology**, v. 68, n. 8, p. 7406–7415, 2019.

26 DONG, W. et al. Autoencoder regularized network for driving style representation learning. **CoRR**, abs/1701.01272, 2017.

27 PRIYADHARSHINI, G.; UKRIT, M. F. An empirical evaluation of importance-based feature selection methods for the driver identification task using obd data. **International Journal of System Assurance Engineering and Management**, 2022. ISSN 0976-4348.

28 AQUINO, A. L. L. et al. Characterization of electric load with information theory quantifiers. **Physica A: Statistical Mechanics and its Applications**, v. 465, p. 277–284, 2017.

29 SHANNON, C. E. A mathematical theory of communication. **The Bell System Technical Journal**, v. 27, n. 3, p. 379–423, 1948.

30 LóPEZ-RUIZ, R.; MANCINI, H. L.; CALBET, X. A statistical measure of complexity. **Physics Letters A**, v. 209, n. 5-6, p. 321–326, 1995. ISSN 03759601.

31 LAMBERTI, P. W. et al. Intensive entropic non-triviality measure. **Physica A: Statistical Mechanics and its Applications**, v. 334, n. 1-2, p. 119–131, 2004.

32 FELIPE, O. et al. Informational time causal planes: A tool for chaotic map dynamic visualization. In: **Nonlinear Systems-Theoretical Aspects and Recent Applications**. [S.l.: s.n.], 2019.

33 MARTIN, M. T.; PLASTINO, A.; ROSSO, O. A. Generalized statistical complexity measures: Geometrical and analytical properties. **Physica A: Statistical Mechanics and its Applications**, v. 369, n. 2, p. 439–462, 2006. ISSN 0378-4371.

34 SILVA, M. J. **Characterization and sensitivity analysis of vehicular mobility models using theory information quantifiers**. Tese (Doutorado) — Institute of Exact and Biological Sciences, Federal University of Ouro Preto, 2020.

35 ZANIN, M.; OLIVARES, F. Ordinal patterns-based methodologies for distinguishing chaos from noise in discrete time series. **Communications Physics**, v. 4, n. 1, p. 190, 2021. ISSN 2399-3650.

36 BANDT, C.; POMPE, B. Permutation entropy: A natural complexity measure for time series. **Phys. Rev. Lett.**, v. 88, p. 174102, 2002.

37 NASCIMENTO, W. S.; PRUDENTE, F. V. Study of shannon entropy in the context of quantum mechanics: An application to free and confined harmonic oscillator. In: **Quim. Nova**. [S.l.: s.n.], 2016. v. 39, n. 6, p. 757–764.

38 MATEOS, D. M.; RAMíREZ, J. G.; ROSSO, O. A. Using time causal quantifiers to characterize sleep stages. **Chaos, Solitons & Fractals**, v. 146, p. 110798, 2021. ISSN 0960-0779.

39 SCHOBER, P.; BOER, C.; SCHWARTE, L. A. Correlation coefficients: Appropriate use and interpretation. **Anesthesia & Analgesia**, v. 126, n. 5, 2018. ISSN 0003-2999.

40 MUKAKA, M. M. Statistics corner: A guide to appropriate use of correlation coefficient in medical research. **Malawi Med J**, Malawi, v. 24, n. 3, p. 69–71, 2012.

41 PESSA, A. A. B.; RIBEIRO, H. V. ordpy: A python package for data analysis with permutation entropy and ordinal network methods. **Chaos: An Interdisciplinary Journal of Nonlinear Science**, v. 31, n. 6, p. 063110, 2021.

42 PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

43 GUO, G. et al. Knn model-based approach in classification. In: SPRINGER. **On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings**. [S.l.], 2003. p. 986–996.

44 CRISTIANINI, N.; RICCI, E.; KAO, M. Y. Support vector machines. In: ____. **Encyclopedia of Algorithms**. Boston, MA: [s.n.], 2008. p. 928–932. ISBN 978-0-387-30162-4.

45 NOBLE, W. S. What is a support vector machine? **Nature biotechnology**, v. 24, n. 12, p. 1565–1567, 2006.

46 KINGSFORD, C.; SALZBERG, S. L. What are decision trees? **Nature biotechnology**, v. 26, n. 9, p. 1011–1013, 2008.

47 SHALEV-SHWARTZ, S.; BEN-DAVID, S. **Understanding Machine Learning: From Theory to Algorithms**. [S.l.]: Cambridge University Press, 2014.

48 BIAU, G.; SCORNET, E. A random forest guided tour. **Test**, v. 25, p. 197–227, 2016.

49 MURTAGH, F. Multilayer perceptrons for classification and regression. **Neurocomputing**, v. 2, n. 5, p. 183–197, 1991. ISSN 0925-2312.

50 LEWIS, D. D. Naive (bayes) at forty: The independence assumption in information retrieval. In: NÉDELLEC, C.; ROUVEIROL, C. (Ed.). **Machine Learning: ECML-98**. Berlin, Heidelberg: [s.n.], 1998. p. 4–15. ISBN 978-3-540-69781-7.

51 WEBB, G. I. Naïve bayes. In: ____. **Encyclopedia of Machine Learning**. Boston, MA: Springer US, 2010. p. 713–714. ISBN 978-0-387-30164-8.

52 BANSAL, A.; SINGHROVA, A. Performance analysis of supervised machine learning algorithms for diabetes and breast cancer dataset. In: **2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)**. [S.l.: s.n.], 2021. p. 137–143.