Anwar Mehmood Sohail\textsuperscript{1}, Ying Wah Teh\textsuperscript{1}, Nauman Khan\textsuperscript{2,3,4,5}, Zahid Khan\textsuperscript{3}, and Anis Koubaa\textsuperscript{3}

\textsuperscript{1}Department of Information Systems, Faculty of Computer Science and Information Technology, Universiti Malaya
\textsuperscript{2}Department of Computer System and Technology, Faculty of Computer Science and Information Technology, Universiti Malaya
\textsuperscript{3}College of Computer and Information Sciences, Prince Sultan University
\textsuperscript{4}Department of Computer Science and Information Technology, University of Malakand
\textsuperscript{5}KPK

March 06, 2024

Abstract

Driver identification is crucial for various applications including automotive security, law enforcement, and the ride-sharing industry, as well as for advanced driver assistance systems, fleet management, and usage-based insurance. Machine learning and deep learning techniques have emerged as promising approaches for accurate identification, yet a comprehensive analysis of existing methods remains unattainable. Despite significant research efforts, no in-depth survey has thoroughly evaluated and compared existing driver identification techniques, impeding the development of optimal solutions. This gap restricts our understanding of their strengths, weaknesses, and potential impact. This paper addresses this critical gap by comprehensively reviewing and analyzing existing driver identification techniques. We delve into various methodologies, including preprocessing feature extraction, classification algorithms, and deep learning architectures. We critically evaluate their performance, highlighting unique features, potential advantages, and limitations. Additionally, we proposed the future framework for driver identification with Large Language Model (LLM) and explored the developing potential of LLM in this domain. We employ relevant research articles published in prominent scientific databases. Our analysis reveals diverse driver identification techniques, each offering unique advantages and disadvantages. Traditional methods like Support Vector Machines and Random Forests provide reliable performance, while deep learning architectures achieve higher accuracy but require larger datasets and computational resources. We also identify potential synergies between established techniques and emerging technologies like LLM, suggesting promising avenues for future research. We propose several key research directions based on our findings to further advance driver identification accuracy and robustness. These include exploring hybrid approaches combining traditional and deep learning methods and investigating transfer learning techniques for efficient adaptation to new data sets. This comprehensive review provides a valuable resource for researchers and practitioners interested in driver identification. We highlight the strengths and weaknesses of existing techniques, identify potential for future advancements, and suggest promising research directions to ultimately achieve highly accurate and reliable driver identification systems.
A Comprehensive Review: Analysis of Machine Learning, Deep Learning, and Large Language Model Techniques for Revolutionizing Driver Identification

Anwar Mehlood Sohail\textsuperscript{a}, Ying Wah Teh\textsuperscript{a}, Nauman Khan\textsuperscript{b,c,d}, Zahid khan\textsuperscript{c}, Anis Koubaa\textsuperscript{d},

\textsuperscript{a}Department of Information Systems, Faculty of Computer Science and Information Technology, Universiti Malaya, Kuala Lumpur, 50603, Malaysia
\textsuperscript{b}Department of Computer System and Technology, Faculty of Computer Science and Information Technology, Universiti Malaya, Kuala Lumpur, 50603, Malaysia
\textsuperscript{c}College of Computer and Information Sciences, Prince Sultan University, Riyadh, 11586, Saudi Arabia
\textsuperscript{d}Department of Computer Science and Information Technology, University of Malakand, Chakdara (Dir lower), KPK, 18800, Pakistan

Abstract

Driver identification is crucial for various applications including automotive security, law enforcement, and the ride-sharing industry, as well as for advanced driver assistance systems, fleet management, and usage-based insurance. Machine learning and deep learning techniques have emerged as promising approaches for accurate identification, yet a comprehensive analysis of existing methods remains unattainable. Despite significant research efforts, no in-depth survey has thoroughly evaluated and compared existing driver identification techniques, impeding the development of optimal solutions. This gap restricts our understanding of their strengths, weaknesses, and potential impact. This paper addresses this critical gap by comprehensively reviewing and analyzing existing driver identification techniques. We delve into various methodologies, including preprocessing feature extraction, classification algorithms, and deep learning architectures. We critically evaluate their performance, highlighting unique features, potential advantages, and limitations. Additionally, we propose the future framework for driver identification with Large Language Model (LLM) and explored the developing potential of LLM in this domain. We employ relevant research articles published in prominent scientific databases. Our analysis reveals diverse driver identification techniques, each offering unique advantages and disadvantages. Traditional methods like Support Vector Machines and Random Forests provide reliable performance, while deep learning architectures achieve higher accuracy but require larger datasets and computational resources. We also identify potential synergies between established techniques and emerging technologies like LLM, suggesting promising avenues for future research. We propose several key research directions based on our findings to further advance driver identification accuracy and robustness. These include exploring hybrid approaches combining traditional and deep learning methods and investigating transfer learning techniques for efficient adaptation to new data sets. This comprehensive review provides a valuable resource for researchers and practitioners interested in driver identification. We highlight the strengths and weaknesses of existing techniques, identify potential for future advancements, and suggest promising research directions to ultimately achieve highly accurate and reliable driver identification systems.

KEYWORDS:
Driver Identification, Driver Recognition, Machine Learning, Deep Learning, Large language Model (LLM), Review, ChatGPT, GPT-4, Bert

1. Introduction

Driver identification, also called driver fingerprinting, is the process of discerning the identity of the individual operating a vehicle [1, 2]. This process aims to recognize or verify the identity of a person who is driving a vehicle, which involves the classification or identification of a driver. Human drivers exhibit a diverse range of behaviors while operating a vehicle [3]. Consequently, this distinct driving behavior can serve as a digital fingerprint characterized by a unique style specific to each driver. Drivers demonstrate distinct driving patterns influenced by their characteristics and skills [4]. For instance, drivers may exhibit variations in their maximum and minimum speeds when traveling on the same road. Certain drivers may engage in frequent gas or brake pedal operations, particularly in heavy traffic or on uneven roads, while others may not exhibit the same behavior. Hence, these distinctive driving patterns can be effectively utilized to identify drivers [5, 6]. Exploring this unique aspect of individual driving behavior style has been an area of extensive research in recent years and continues to be actively investigated. This research area has gained significant momentum, particularly with the increasing number of sensors and the abundance of data available in modern vehicles.

Fig 1 illustrates the various domains in which this technology holds promise for numerous advantages, such as in the use of ride-sharing industry, automotive security, Advance driving assistance system (ADAS) [7], usage-based insurance (UBI) systems by penalizing insurers who lend cars to other drivers, law enforcement agencies by confirming identity of persons who break traffic laws, etc. [8, 9, 10].

1.1. Automotive security:

Vehicle transportation plays a crucial role in the lives of contemporary individuals. A significant portion of the population owns vehicles...
and utilizes them regularly. Nevertheless, there has been a notable rise in the occurrence of vehicle theft over the past decade. According to the FBI Uniform Crime Reports, the number of stolen vehicles has increased by 5% over the past five years. Additionally, statistics from the Home Office indicate that a staggering 111,999 vehicles were reported stolen between 2017 and 2018 [3].

Driver identification can be used to prevent unauthorized access to vehicles, protect against theft, and improve safety by ensuring that only authorized drivers operate the vehicle [11, 12, 13]. In their study, [14] introduce a driver authentication approach that utilizes the vehicle’s sensors to analyze patterns of driver behavior. By employing Machine Learning algorithms, they generate individual driver profiles for the verification process, which are subsequently compared to existing profiles.

1.2. Law enforcement:

Driver identification is crucial for law enforcement agencies to ensure road safety and prevent criminal activities involving vehicles. With driver identification technology, law enforcement agencies can accurately identify drivers involved in road accidents, traffic violations, and criminal activities and take appropriate legal action. The technology can also help agencies track stolen vehicles and identify suspects involved in hit-and-run cases [15, 16, 17, 18]. According to the research conducted by [16, 17] (Dolos, Meyer et al. 2020, Dolos, Meyer, et al. 2021), the utilization of in-vehicle digital data for driver identification within the realm of digital forensics holds the possibility for resolving hit and run accidents and other incidents involving vehicles.

1.3. Ride sharing-hailing industry:

The transportation industry has undergone a significant transformation with the emergence of ride-sharing services, which cater to both professional drivers and the sharing of vehicles among multiple passengers. Driver identification is an important aspect of the ride-sharing industry such as Uber or Cabify [19, 20]. It helps to ensure the safety and security of passengers by verifying the driver’s identity and ensuring that they are licensed and authorized to operate the vehicle. It also helps to prevent fraudulent activity by ensuring that only authorized drivers can provide rides through the platform [21].

1.4. Advanced Driver Assistance Systems (ADAS):

Driver identification can also play a vital role in ADAS by personalizing the system settings based on the driver’s preferences and driving behavior [22]. For example, once the driver is recognized, the ADAS can automatically adjust the seat position, mirror angles, and steering sensitivity [23, 24]. This can improve the overall driving experience and provide greater comfort for drivers. In their study, [25] tackle driver identification within infotainment. They propose that depending on who is driving, the ADAS in the car may be customized to accommodate particular family members.

1.5. Insurance:

Driver identification is crucial for insurance companies to accurately assess the risk posed by each driver and determine the appropriate insurance premiums. By analyzing data about the driving behavior of each individual driver, insurance companies can provide more personalized insurance plans and reward safe driving practices, leading to more fair and accurate pricing for customers. Additionally, driver identification can help prevent fraudulent claims and reduce insurance fraud, ultimately benefiting both insurance companies and their customers [15].

1.6. Fleet management:

Driver identification plays a crucial role in fleet management by allowing companies to monitor and track the driving behavior of individual drivers. This information can be used to optimize route planning, monitor fuel consumption, and reduce vehicle wear and tear, ultimately enhancing efficiency and resulting in cost savings [28, 29].

Previous studies in driver identification have established a standardized framework called the driver identification pipeline, which outlines the procedures involved in designing and implementing driver identification systems. This pipeline encompasses various stages, including driving data collection, data preprocessing, feature extraction, dimensionality reduction, building classification models, as well as testing and validating the learning models. Figure 3 illustrates the process.

The driver identification process initiates with the collection of data, as illustrated in Figure 3, this involves gathering various types of data sources, such as biometric, behavioral, and contextual data about the driver using sensors, cameras, and other devices. These data sources are discussed in detail in Section 2. Data pre-processing is another crucial step, which entails transforming the raw driving data into a suitable representation. The pre-processing begins with filtering to remove noise [1, 5]. Data cleaning [30, 31, 32]. Feature scaling [33, 32, 5]. Time windowing divides the data into smaller segments using either overlapping or fixed window sizes, enabling the extraction of distinctive features from each segment [34, 35]. Class balancing techniques address class imbalance [18], and encoding categorical variables [36, 37]. This comprehensive pre-processing pipeline enhances data quality, ensures consistent scaling, captures temporal patterns [38, 39], increases diversity, addresses class imbalance, and enables effective numerical representation, leading to accurate and reliable driver identification models [40, 41]. A comprehensive discussion of these pre-processing methods can be found in Section 3, where they are examined in detail.

To create a useful collection of features, feature extraction and feature selection approaches are used, with the goals of improving accuracy, speeding up computation, and reducing classification mistakes. There are shallow features and deep features in the process of obtaining driving features. Shallow features include the extraction of traditional handmade characteristics such statistical time- and frequency-domain features. However, shallow features heavily rely on domain expertise, necessitate a substantial amount of labeled driving data, and employment of dimensionality reduction techniques that lack generalizability. Deep learning is a successful solution to address the aforementioned challenges due to its capability to construct features effectively. Deep learning algorithms are end-to-end machine learning algorithms that can be trained on raw data to automatically learn the representations needed for tasks like detection or classification. In various fields, deep learning methods outperform traditional approaches, particularly in constructing high-dimensional representations for complex data. The most significant advantage of deep learning is its ability to automatically map low-dimensional, shallow features to high-dimensional, deep features based on the objective function, eliminating the need for manual feature engineering or domain expertise. Since 2016, there has been a shift towards deep learning algorithms replacing conventional machine learning methods.
A Comprehensive Review: Analysis of Machine Learning, Deep Learning, and Large Language Model Techniques for Revolutionizing Driver Identification

2. Related work

Several surveys and reviews studies on driver identification have been published recently. [1, 27, 26]. However, these articles primarily focus on different machine learning and deep learning methods and techniques used for driver identification. In contrast, this study not only emphasizes machine learning and deep learning methods but also delves into datasets, data sources, pre-processing methods, feature selection techniques and potential use of LLM for driver identification. For instance, [27] presented a review that specifically discussed traditional machine learning algorithms used for driver identification or fingerprinting. Their study concentrated solely on traditional machine learning algorithms used for driver identification or fingerprinting. Their study concentrated solely on traditional machine learning algorithms used for driver identification or fingerprinting. Deep learning techniques are particularly popular in pattern recognition, natural language processing, and now driver identification. Examples include deep autoencoder, CNN, and RNN. These techniques and their respective studies are discussed in detail in Section 6.

Driving features combined with machine learning (ML) algorithms to build a driver identification model. Some of the ML algorithms employed in this context include Support Vector Machine [43, 44], Decision Tree [45, 29] and Random Forest [16, 17]. The specifics of these classification algorithms are discussed in Section 5. In the case of Deep Learning, both feature extraction and classification are trained as part of the model construction process. In the end, the driver identification system undergoes evaluation using diverse performance metrics, including accuracy, precision, and recall. These metrics help assess the effectiveness and reliability of the system in accurately identifying drivers.

The goal of this work is to undertake a thorough analysis of driver identification systems that make use of various deep learning and machine learning classifiers. In this paper, we seek to provide a comprehensive summary of current developments in driver identification. We go through the datasets and data sources with their characteristics, several driving pre-processing strategies, and feature selection techniques in particular. We also look at several machine learning classifiers and deep learning techniques used in building driver identification models, as well as assessment metrics to rate the effectiveness of classification methods. Furthermore, this study discusses potential use of LLM and future research directions for further enhancements and areas of focus based on the findings of the reviewed studies. Figure 2. illustrates the taxonomy used in this review, while Table 2 presents the abbreviations used in this study.

2.1. Motivation and Contribution

The present study differs from previous reviews in several aspects. Firstly, it encompasses the most up-to-date techniques and methods employed in driver identification, providing a comprehensive analysis of different machine learning and deep learning techniques utilized for this purpose. It also examines the various data sources employed in driver identification. Secondly, this review goes beyond the scope of previous studies by including datasets, pre-processing techniques, feature selection methods, and LLM specific to driver identification. There is no existing comprehensive review available, which addresses all the components necessary for implementing a driver identification system to the best of our knowledge. A comparison of recently released survey papers is shown in Table 1, highlighting the areas they cover for reference. The paper makes the following contributions:

- Summarizing recent advancements in the construction of datasets, pre-processing, feature engineering, and the utilization of variety of machine learning classifiers and deep learning methods for driver identification.
- Providing an analysis of driver identification techniques, outlining their strengths and weaknesses.
- Proposing a future framework for driver identification with LLM and highlighting applications of the advanced LLM framework, its potential to revolutionize this field.
- Emphasizing future challenges in the creation of datasets, feature engineering, and classification models for driver identification; and identifying research gaps in these areas.

The remainder of this research is structured as follows: A review of data sources and datasets commonly used for driver identification is presented in Section 3. Data pre-processing techniques are discussed in Section 4. Various feature selection methods are explored in Section 5. Machine learning and deep learning methods for driver identification are covered in Sections 6 and 7, respectively, highlighting their respective strengths and weaknesses. Section 8 and 9 describes the framework and utilization of LLM in the context of driver identification respectively. Section 10 details the performance assessment of machine learning and deep learning models for driver identification. Section 11 addresses open issues, observations, and future research directions for further improvements and focus. Finally, this comprehensive revised article is concluded in Section 12.

---

Table 1. Table shows the comparison between recent reviews on driver Identification and the topics covered in this study. The comparison is based on the inclusion of driving preprocessing methods (Prep), data sources (DS), datasets (DST), feature selection (FSel), machine learning, deep learning methods, LLM (ML/DL/LLM), and evaluation metrics (EMetr).

<table>
<thead>
<tr>
<th>Publication</th>
<th>Year</th>
<th>Prep</th>
<th>DS</th>
<th>DST</th>
<th>FSel</th>
<th>ML</th>
<th>DL</th>
<th>LLM</th>
<th>EMet</th>
</tr>
</thead>
<tbody>
<tr>
<td>This Study</td>
<td>2023</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

---

learning techniques for feature extraction and their use in driver identification tasks. [42] introduced various deep neural network architectures, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to learn representations of driver’s driving styles. After that, several other studies utilized deep learning methods, and indicates that this approach outperformed traditional machine learning algorithms. Deep learning techniques are particularly popular in pattern recognition, natural language processing, and now driver identification. Examples include deep autoencoder, CNN, and RNN. These techniques and their respective studies are discussed in detail in Section 6.

---

For instance, [27] presented a review that specifically discussed traditional machine learning algorithms used for driver identification or fingerprinting. Similarly, [1] compared and analyzed different approaches for driver identification based solely on driving behavior, without considering physical or physiological features. Another closely related review was conducted by [26], which examined various data sources, ML, and DL algorithms for driver identification. However, our work offers a thorough justification of significant new planned investigations that concentrate on cutting-edge deep learning algorithms for driver detection.
3. Data sources for Driver Identification

In the domain of driver identification, researchers and practitioners have extensively explored a wide array of data sources (Table 3 ) to develop robust and accurate identification techniques [46]. This section delves into the exploration of numerous data sources used for driver identification, aiming to provide insights into the diverse range of information that can be harnessed for this purpose. Recognizing the significance of driver identification in various applications, including vehicle security, personalized driver assistance, and insurance risk assessment, the selection and analysis of appropriate data sources are crucial steps in the development of effective identification methods. Additionally, the strengths and weaknesses of each data source will be examined, shedding light on their respective advantages and limitations in the context of driver identification. By comprehensively exploring and understanding these data sources, researchers and practitioners can leverage their unique characteristics to develop more accurate and reliable driver identification systems, contributing to the advancement of driver-centric technologies and systems. Additionally, the datasets used in the literature is also provided in the Table 4 with their detail. Fig. 4 . and Fig. 5 shows the frequency of datasets and data sources used in the literature respectively.

Some researchers utilize driver biometric data for driver identification purposes [47]. This includes data such as facial features [48, 20], actions [49], fingerprints [50], voice characteristics [51], physiological states [52], gestures [53], and grip patterns [54]. These kinds of data provide significant relevance and excellent identification accuracy since they directly represent the distinctive biological traits of the driver. However, challenges arise in acquiring such data as it often requires new devices, leading to issues like space requirements and increased costs. Additionally, continuous real-time recognition poses difficulties, and there are concerns regarding the collection of information that may infringe on personal privacy. Furthermore, limitations in application arise, including the potential for data falsification. Therefore, we reviewed all the data sources which are recently emerging for driver identification, despite one study which is based on facial features [20].

3.1. Smartphone sensors

Smartphone sensors have emerged as a valuable and accessible data source for driver identification. With their increasing ubiquity, smart-
phones offer a range of built-in sensors, including accelerometers, gyroscopes, GPS, and magnetometers, which can capture various aspects of driver behavior and vehicle dynamics. These sensors enable researchers to collect large-scale and real-world driving data for identification purposes. The strengths of smartphone sensors lie in their convenience, as most individuals carry smartphones with them while driving, allowing for continuous data collection without the need for additional devices. Additionally, they make it possible to acquire data from vehicles that do not support CAN bus data. Moreover, smartphones offer a diverse set of sensors that can capture different dimensions of driving behavior. However, smartphone sensors also have certain limitations. One key weakness is the potential variability in sensor quality and accuracy across different smartphone models, leading to potential inconsistencies in data collection and reliability. Additionally, issues related to sensor calibration, battery life, and data privacy need to be addressed when utilizing smartphone sensors for driver identification. Despite these challenges, the use of smartphone sensors holds great potential for driver identification, offering a scalable and cost-effective approach to gather valuable data on driver behavior and patterns.

3.1.1. Accelerometer and Gyroscope

Several studies have been conducted to explore accelerometers and gyroscope sensors for driver identification [55, 56]. Measured driving behavior signals from a smartphone and accurately identified three drivers using only two acceleration signals. In [15], a driver identification system relying on histograms of acceleration data acquired from the accelerometer sensor achieved an accuracy of up to 99% and demonstrated effective impostor detection. However, it is important to note that the dataset utilized in their research was confined to campus shuttle buses, limiting the generalizability of their findings. [57] also investigated driver identification using a 3-axis accelerometer and found it performed best. However, when there were more drivers, the algorithm’s performance suffered. A driver recognition system that made use of drivers’ maneuvers [58, 59, 60], such as right and left turns, was reported in [44], which utilized acceleration and orientation data. More recently, [35] proposed a driver identification system based on smartphone accelerometer and gyroscope signals, while [13] demonstrated competitive results using features extracted from smartphone accelerometers. Accelerometers and gyroscopes serve as valuable alternatives in environments with tall buildings and tunnels, where GPS receivers frequently encounter difficulties. These sensors prove especially useful in such scenarios, providing reliable data even when GPS signals are unavailable. Additionally, using accelerometers and gyroscopes mitigates concerns regarding privacy invasions that are frequently connected to GPS data.

3.1.2. Global Positioning System (GPS)

GPS is widely used for driver identification, providing valuable insights into driver behavior and patterns [61, 62]. It accurately tracks vehicle location, movement, and spatial behaviors. The strengths of GPS include precise trajectory information, real-time data availability, and analysis of driving patterns. However, GPS has limitations such as signal interference, satellite availability, and limited granularity for fine-grained identification. Nonetheless, GPS remains a valuable tool for studying driver behavior on a larger scale and can be combined with other data sources to enhance identification accuracy and reliability. Several researchers have used GPS for driver identification, showcasing its feasibility and effectiveness. [63] (Chowdhury, Chakravarty et al. 2018) demonstrated the potential of smartphone GPS data for distinguishing drivers and achieved an accuracy of 82%. [43, 64] (Rahim, Liu et al. 2020, Rahim, Zhu et al. 2020) created a method for identifying drivers that was based on GPS-derived driving patterns. This system achieved a remarkable accuracy of 96%.

3.1.3. CAN-BUS/OBD-II

CAN-BUS (Controller Area Network)/On-board Diagnostic (OBD-II) data has been extensively utilized for driver identification purposes in the context of vehicle identification. The vehicle’s OBD-II system is connected to various sensors and systems, such as the engine, brakes, and transmission, which provide a wealth of information about the vehicle's operation. This data can be useful for identifying the driver by analyzing patterns and symptoms unique to individual drivers.

### Table 2: Abbreviation List with Definition

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLM</td>
<td>Large language model</td>
</tr>
<tr>
<td>GPT-4</td>
<td>Generative Pre-trained Transformer-4</td>
</tr>
<tr>
<td>ADAS</td>
<td>Advance Driving Assistance System</td>
</tr>
<tr>
<td>UBI</td>
<td>Usage-Based Insurance</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>CAN</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>OBD</td>
<td>On-board Diagnostic</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>EMG</td>
<td>Electromyogram</td>
</tr>
<tr>
<td>CQT</td>
<td>Constant Q Transform</td>
</tr>
<tr>
<td>SMOTE</td>
<td>Synthetic Minority Over-sampling Technique</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>CMIM</td>
<td>Conditional Mutual Information Maximization</td>
</tr>
<tr>
<td>FCD</td>
<td>Floating Car Data</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>LR</td>
<td>Linear Regression</td>
</tr>
<tr>
<td>KNN</td>
<td>K-nearest Neighbor</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>RF</td>
<td>Random Forest</td>
</tr>
<tr>
<td>GBRF</td>
<td>Gradient Boosting with Random Forest</td>
</tr>
<tr>
<td>SVDD</td>
<td>Support Vector Domain Description</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>DNN</td>
<td>Deep Neural Network</td>
</tr>
<tr>
<td>CNN</td>
<td>Convolutional Neural Network</td>
</tr>
<tr>
<td>RNN</td>
<td>Recurrent Neural Network</td>
</tr>
<tr>
<td>TS-CNN</td>
<td>Three-stream Convolution Neural Network</td>
</tr>
<tr>
<td>1-D CNN</td>
<td>One-dimensional Convolutional Neural Network</td>
</tr>
<tr>
<td>RCN</td>
<td>Residual Convolution Network</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long Short-Term Memory</td>
</tr>
<tr>
<td>ResNet</td>
<td>Residual Neural Network</td>
</tr>
<tr>
<td>DAE</td>
<td>Deep Autoencoder</td>
</tr>
<tr>
<td>NCAE</td>
<td>Nonnegativity Constrained Autoencoder</td>
</tr>
<tr>
<td>GAN</td>
<td>Generative Adversarial Network</td>
</tr>
<tr>
<td>CLGAN</td>
<td>Convolutional Long short-term GAN</td>
</tr>
<tr>
<td>SGM</td>
<td>Stacked Generalization Method</td>
</tr>
<tr>
<td>SGRUs</td>
<td>Stacked Gated Recurrent Units</td>
</tr>
<tr>
<td>MTL</td>
<td>Multi-task Learning Network</td>
</tr>
<tr>
<td>BLSTM</td>
<td>Bidirectional Long short-term Memory</td>
</tr>
<tr>
<td>GPT</td>
<td>Generative Pre-trained Transformer</td>
</tr>
<tr>
<td>CSA</td>
<td>Crow Search Algorithm</td>
</tr>
</tbody>
</table>
various research studies [65]. The utilization of CAN-BUS data enables the extraction of valuable information related to vehicle parameters and behavior, offering insights into driver identification [66]. One of the notable strengths of CAN-BUS data is its availability in most modern vehicles, making it a convenient and accessible source for driver identification. Additionally, the use of CAN-BUS data allows for cost-effective solutions as it eliminates the need for additional sensors or devices. However, it is essential to acknowledge some limitations of CAN-BUS data for driver identification. One key weakness is the potential lack of standardization across different vehicle models, which may affect the consistency and compatibility of the data. The interpretation and analysis of the significant amount of complicated data produced by the CAN-BUS presents another difficulty, requiring sophisticated algorithms and techniques for accurate driver identification.

Several researchers have utilized CAN bus data for driver identification in various studies. In the context of infotainment, CAN-bus was employed by [25] to tackle driver identification. [16, 17] assessed the potential of using CAN-BUS data in digital forensics for hit-and-run car accident scenarios. A machine learning-based solution for driver profiling utilizing in-vehicle CAN-BUS sensors was developed by [45]. [1] developed a deep learning-based system using only steering behavior from CAN-BUS data. [18] demonstrated the utilization of CAN bus data to predict driver identity.

3.1.4. Biometric data

Biometric data has emerged as a promising approach for driver identification, leveraging individuals’ unique physiological or behavioral characteristics. Biometric data sources such as fingerprint, voice, iris, and facial recognition have been employed in driver identification systems. Biometric data’s strengths lie in their uniqueness, making them highly reliable for individual identification. They provide a non-intrusive and convenient means of identification, requiring minimal effort from the driver. Additionally, biometric data can be difficult to forge or manipulate, enhancing security. However, biometric data also has its limitations. Environmental influences, sensorlimitations, and human variability can all impact the reliability and accuracy of biometric data. Biometric systems may also face privacy concerns and potential biases. Using facial recognition, [20] provides a real-time and reliable driver identification approach. Introducing a lightweight CNN model, they achieved an average verification accuracy of 87.44%.

3.1.5. Physiological sensors

Physiological sensors, such as ECG (Electrocardiogram) and EMG (Electromyogram), have been explored for driver identification due to their ability to capture unique physiological signals. The strengths of physiological sensors lie in their high specificity and reliability, as they directly reflect the individual’s physiological responses. They can offer real-time monitoring of the driver’s physiological state, allowing for immediate identification of anomalies or changes in driver behavior. However, physiological sensors also have some limitations. They require direct contact with the driver’s body, which can be uncomfortable or inconvenient in certain situations. The accuracy and effectiveness of physiological sensors can be influenced by sensor placement, signal interference, and individual variations in physiological responses. Additionally, using physiological sensors for driver identification may raise privacy concerns related to collecting and storing sensitive health data. Therefore, while physiological sensors offer the potential for driver identification, carefully considering user comfort, signal quality, and privacy safeguards is necessary for their successful implementation. [23, 30] proposed a driver identification system by employing ECG morphological features. [67] demonstrate that including driver-related information like heart rate and driving pattern-related data can significantly improve the effectiveness of driver identification.

3.1.6. Smartwatch

Smartwatches have been utilized for driver identification, leveraging their built-in sensors such as accelerometers and heart rate monitors. The strengths of smartwatches include portability, convenience, and continuous monitoring. They provide valuable data for driver behavior analysis. However, limitations include limited sensor accuracy, dependency on user compliance, and potential privacy concerns due to continuous data collection. In their study, [44] introduced the concept of driver identification using smartwatches. Their system concentrated on analyzing drivers’ maneuvers, specifically right and left turns, utilizing data gathered from smartwatches. The system underwent evaluation in both real-world and simulated environments, yielding an error rate of
and training drivers using 13-second-long testing data, irrespective of the routes. DNN (Deep Neural Network), achieved an accuracy of 81% in identifying drivers, irrespective of the routes. Their algorithm, called the GPS device and CAN bus. [72] utilized accelerometer and GPS data to address the driver's identification with signals obtained from both the GPS device and CAN bus. Many studies combine action, combining data from multiple sensors to enhance accuracy and behavior by integrating information from various sources such as accelerometers, gyroscopes, GPS, and cameras. This approach allows for a more robust identification system that considers multiple modalities simultaneously. The strengths of sensor fusion include improved accuracy through data integration, better contextual understanding of driver actions, and the ability to capture a wide range of driving features. However, challenges in sensor calibration, synchronization, and data fusion algorithms can pose weaknesses, requiring careful design and implementation to ensure optimal performance. Many studies combine multiple data sources to improve driver identification accuracy. [25] addressed the driver's identification with signals obtained from both the GPS device and CAN bus. [72] utilized accelerometer and GPS signals in their driver identification approach. Their algorithm, called DNN (Deep Neural Network), achieved an accuracy of 81% in identifying drivers using 13-second-long testing data, irrespective of the routes and traffic conditions. [36] have introduced a method to optimize driver detection using data extracted from GPS and in-car sensors. To improve identification accuracy, [30] suggest a driver recognition system that concurrently uses both electrocardiogram (ECG) and electromyogram (EMG) data in various driving states, accounting for the actual driving environment. Compared to utilizing either a single EMG or ECG signal alone, the results show that combining both EMG and ECG data increases accuracy. Their optimized deep learning model's performance on merging Smartphone data and CAN-BUS data was evaluated by [37]. Suggested a driver identification system based on smartphone accelerometer and gyroscope signals [35].

4. Data pre-processing for driver Identification

In this section, we will discuss the data pre-processing techniques utilized in a review paper focused on driver identification. We will highlight the strengths and weaknesses of these techniques as outlined in the paper. These pre-processing methods have been carefully selected and evaluated to enhance the effectiveness and accuracy of driver identification models. By understanding their strengths and weaknesses, we can gain insights into their suitability for different datasets and contexts, ultimately informing the development of robust driver identification systems.

4.1. Data Cleaning

Data cleaning encompasses the process of addressing missing values, outliers, irrelevant features, noise, and instances of 0 km/hr speed that occur when vehicles come to a stop at traffic signals or encounter traffic congestion in the dataset. Data cleaning ensures that the subsequent analysis is based on reliable and accurate information. [15] removed inactive periods (0 km/hr) where bus stops from the dataset. They demonstrate that the application of an inactive-period filtering module leads to a significant increase in accuracy, with improvements of up to 18%.

4.2. Feature Scaling

Feature scaling is crucial to ensure that the input features are standardized to a similar scale. It prevents certain features from dominating others and helps in achieving better convergence during model training. Common scaling methods for feature scaling include normalization techniques such as min-max scaling and standardization techniques such as z-score scaling.

4.3. Data augmentation

Data augmentation is a method used to artificially increase the quantity and variety of the training sample. In the realm of driver identification, data augmentation can involve the application of random transformations or perturbations to the driving data, such as rotations, translations, or the addition of noise. By implementing data augmentation, the model's generalization capabilities are enhanced, and the risk of overfitting is mitigated. In this regard, [31] used up-sampling, introducing noise, data reversal, and random drifting as four methods of time-series data augmentation. These techniques were used to enrich the initial training data and improve the performance of an ensemble deep model for driver identification. Remarkably, the authors achieved an accuracy exceeding 50% even with only one minute of driving data. Their findings highlight the substantial enhancement in model performance that can be attained through augmentation methods, particularly when the training data quantity is limited. In another work by [35], a hybrid model called GAN-SGM was introduced for driver identification. The Generative Adversarial Network (GAN) was utilized for data augmentation, while the Stacked Generalization Method (SGM) employing KNN, MLP, SVM, RF, and LR classifiers was used for classification. The proposed approach demonstrated exceptional performance, achieving 97% accuracy.

4.4. Time window segmentation

Time window segmentation for driver identification involves dividing the continuous time series data, such as sensor readings or driving behavior observations, into smaller, non-overlapping or overlapping segments or windows of fixed duration. This segmentation allows for the analysis and modeling of temporal patterns or behaviors within those specific time intervals. By breaking the data into manageable segments, it becomes easier to capture and analyze the variations and characteristics of driver-specific features over time. Time window segmentation is particularly useful for driver identification as it helps to extract relevant information from the temporal patterns exhibited by drivers during different time intervals. The choice of window size and overlap percentage can impact the accuracy and granularity of the identification process, and it is typically determined through experimentation and optimization based on the specific dataset and identification requirements.

Multiple studies have investigated the application of time window segmentation for driver identification, starting with the work of [25], who obtained a 95% accuracy after normalizing OBD-II data and applying a sliding window technique. The Extra Tree technique with sliding window partitioning of CAN-Bus signals was used by [67]. They improved classification accuracy from 96.20% to 99.92% by adjusting the processing window size. Then, in the study by [81], a sliding window approach was employed to create a driving fingerprint map using selected features. The parameters of the time window were optimized by considering the outcomes of driver identification classification using
a deep convolutional neural network. By setting the length and shift of the time window to 3 seconds and 1 second, respectively, driver identification accuracy reached 93.5%. The following year, [15] investigated the effect of overlapping sliding time windows, achieving 94% accuracy when the window size was set to 15 minutes with a 75% overlap. [80] introduced a nonnegativity constrained autoencoder (NCAE) network to identify the optimal size of the sliding time window. [75] proposed a driver identification system by implementing an overlapping sliding window segmentation. This approach resulted in identification accuracies ranging from 97.72% to 98.36%. [12] also analyzed their system by manipulating the time window length. [35] achieved the highest accuracy using a window length of 15 minutes and a 75% overlap, while [1] found that longer window lengths improved accuracy, selecting a 30-second window length. [10] determined that a 30-second window size was optimal, achieving an 89.9% macro F1 score. Then, [36] achieved an accuracy of up to 99.7% with only a 10-second driving window using random forest. In summary, the utilization of (overlapping) windowing preprocessing on raw sensor data has demonstrated the cutting-edge identification capabilities of driver classification techniques based on deep learning.

4.5. Spectrogram

A spectrogram is a widely used visualization technique in signal processing that allows the analysis of the frequency content of a signal as it evolves over time [13]. In the context of driver identification, spectrograms can be employed to analyze and represent the frequency characteristics of sensor data collected from a vehicle, such as accelerometer or microphone signals. The spectrogram presents a two-dimensional representation, where the x-axis corresponds to time, the y-axis corresponds to frequency, and the color intensity or shading represents the magnitude or power of the frequency components at each specific time point. By generating spectrograms from the sensor data, it becomes possible to extract patterns and distinct features associated with the driver’s behavior. These extracted characteristics can then be employed for identification purposes. These spectrogram-based features can then be fed into ML or DL models for driver identification tasks. Spectrograms offer insights into the temporal and spectral characteristics of the signals, allowing for effective analysis and discrimination between different driving behaviors and individuals.

[13] implemented a transformation of 1D accelerometer signals into a 2D format, which were subsequently inputted into a discriminative end-to-end deep learning model. [30] put forward a driver identification system that employs a 2D spectrogram. The proposed method involves converting a segmented ECG cycle into a spectrogram as part of the process. The experimental findings show that, in contrast to the current multidimensional characteristics, the proposed approach improves the identification performance. The same authors, [30], utilize two biological signals of the driver for driver identification, specifically ECG and EMG. The system utilizes a conversion process to transform these signals into 2D constant Q transform (CQT) images, which are then input to a multistream CNN for classification purposes. Based on the experimental results, it has been observed that there is a higher accuracy of 98.9% when using 2D CQT representations.

4.6. Class balancing

Class balancing techniques address the issue of class imbalance in the dataset [18], where certain driver classes may have significantly more samples than others. This may result in biased models that are more accurate on the majority class but perform poorly on minority classes. Balancing techniques such as oversampling, like SMOTE (Synthetic Minority Over-sampling Technique), or undersampling can be used to ensure a more balanced representation of different driver classes [18].

4.7. Encoding Categorical Variables

The process of encoding categorical variables entails the conversion of these variables into numerical representations. This can be achieved through various methods such as label encoding or one-hot encoding [37]. Label encoding gives each category a distinct number value whereas one-hot encoding generates binary columns for each distinct category. These encoding techniques enable categorical variables to be used as input in machine learning algorithms that require numerical data. It is required for machine learning algorithms to process them effectively [36].

4.8. Wavelet transform

Wavelet transform is a signal processing method used in driver identification to analyze the frequency content of signals over different
Table 3  
Data sources used for driver identification

<table>
<thead>
<tr>
<th>Data source</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerometers</td>
<td>Provides information on vehicle dynamics and motion. Captures acceleration, deceleration, and lateral movement.</td>
<td>Limited to vehicle-centric information. Requires accurate calibration and placement for reliable measurements. May not capture nuanced driver behavior or contextual information.</td>
<td>[35], [15], [57], [35], [72], [13], [35], [44]</td>
</tr>
<tr>
<td>Gyroscopes</td>
<td>Measures the rotational motion and orientation of the vehicle. Provides insights into steering control and maneuvering.</td>
<td>Limited to vehicle-centric information. Requires accurate calibration and placement for reliable measurements. May not capture driver-specific behavior or characteristics.</td>
<td>[35]</td>
</tr>
<tr>
<td>GPS</td>
<td>Provides accurate vehicle location and trajectory information. Captures driving patterns, routes, and speed.</td>
<td>May have limitations in accuracy and precision, especially in dense urban areas or areas with poor GPS reception. Requires a clear line of sight to satellites for reliable data collection.</td>
<td>[67], [43], [64], [63], [9], [36], [72], [25], [29]</td>
</tr>
<tr>
<td>Smartphone</td>
<td>Wide availability and high adoption rate of smartphones. Captures driving-related data (speed, location) and audio cues. Potential for integrating multiple sensors (accelerometers, GPS, microphones). Non-intrusive data collection method.</td>
<td>Limited to capturing specific aspects of driving behavior. Challenges in accuracy and consistency of sensor measurements. Privacy concerns related to collecting personal smartphone data. May raise concerns of driver distraction if not used appropriately.</td>
<td>[67], [43], [35], [25], [13], [9], [28], [37], [20]</td>
</tr>
<tr>
<td>CAN-BUS/OBD-II</td>
<td>Access to real-time vehicle performance data. Standardized data format across different vehicles. Wide availability in modern vehicles. Provides insights into driving behavior (speed, fuel consumption, braking).</td>
<td>Limited driver-specific information. Data may not capture nuanced driving behavior or contextual information. Limited to vehicles equipped with OBD systems (older vehicles may not have this capability).</td>
<td>[5], [1], [32], [31], [24], [1], [73], [37], [74], [12], [16], [17], [10], [75], [67], [25], [45], [7], [9], [18], [36], [19], [37], [45], [29], [76], [77], [19], [78]</td>
</tr>
<tr>
<td>Biometric Data</td>
<td>Unique to individuals, enabling reliable identification. High visual data resolution. Comprehensive view of the driver. Ability to capture facial expressions. Can capture driver actions and gestures. Difficult to manipulate or impersonate. Non-intrusive data collection method.</td>
<td>Requires specialized sensors or systems for data collection. Privacy concerns and ethical considerations. Challenges in data processing and analysis due to variability in lighting conditions, occlusions, and facial recognition algorithms. Limited in capturing dynamic changes in biometric characteristics (e.g., appearance changes over time).</td>
<td>[20]</td>
</tr>
</tbody>
</table>

scales [35]. It decomposes a signal into a series of wavelet coefficients, which represent the signal’s energy at various frequencies and time intervals. In driver identification, the wavelet transform is utilized to process sensor data, such as accelerometer or gyroscope signals. This transform helps extract features that effectively capture the distinctive patterns and characteristics specific to individual drivers. The wavelet coefficients provide information about both low-frequency and high-frequency components of the signal, allowing for a detailed analysis of driver behavior. By analyzing the wavelet coefficients and their variations, machine learning algorithms can effectively classify and identify
### Feature Selection Methods for Driver Identification

Feature selection plays a critical role in driver identification since it entails carefully selecting a subset of relevant features from a wider set. This process aims to determine the most illuminating and discriminative traits that contribute significantly to the accurate identification of drivers. It plays a vital role by improving model performance, reducing dimensionality, enhancing interpretability, and increasing computational efficiency. By selecting the most informative and discriminative features, feature selection improves classification accuracy and generalization, eliminates redundancy, and reduces overfitting. It enables faster training and inference times, particularly important for real-time applications, while providing insights into the driving behaviors that contribute to driver identification. The study conducted by [16] (Dolos, Meyer et al. 2020) emphasized the importance of evaluating the contribution of available features to the classification process in driver identification. It was emphasized that without a comprehensive understanding of how driving behavior influences each feature, it would be challenging to construct reliable and trustworthy models. Various feature selection techniques (Table 7) have been employed for driver identification in the reviewed studies.

### 5.1. Univariate Feature Selection

This method involves evaluating each feature individually to determine its connection to the desired variable. To rank the features depending on their relevance, statistical methods such as the chi-squared test, t-test, or ANOVA might be used. Features with high statistical scores are considered more relevant and selected for the model. [37] conducted experimentation with simple feature selection methods, including univariate feature selection, and ultimately selected 40 features based on their findings.

### 5.2. Principal Component Analysis (PCA)

PCA is a dimensionality reduction method that converts the original features into a new set of uncorrelated variables called principal components. The greatest amount of data variance is captured by these components. One can successfully minimize the feature space while maintaining the most crucial data by choosing a subset of the top principal components. In a recent study, PCA has been utilized for feature pre-processing in driver identification tasks, leading to significant improvements in accuracy. For instance, [5] reported an approximate 20% increase in accuracy. [56] developed an algorithm that incorporates PCA, specifically focusing on lateral and longitudinal accelerations. Their approach determines driver identification based on the angle between the principal components of test data and those of all training data. Similarly, [45] employed PCA to select the most discriminative features associated with drivers from the dataset. Their findings demonstrate that utilizing only relevant features can enhance accuracy while reducing learning time. The quintessential procedure for Principal Component Analysis (PCA), as delineated by [82], entails a sequence of steps to distill the essence of the feature space. The process commences with the computation of the mean $\bar{x}$ of a given feature vector $\{x_1, x_2, x_3, \ldots, x_n\}$, articulated as

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

(1)

Subsequently, the covariance matrix $\text{Cov}$ is constructed through

$$\text{Cov} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})^T$$

(2)

<table>
<thead>
<tr>
<th>Data source</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physiological Data</td>
<td>Provides physiological responses related to stress, arousal, and cognitive states. Provides insights into driver conditions.</td>
<td>Requires specialized sensors and equipment for data collection. Challenges in calibration and accuracy of physiological measurements. Privacy concerns regarding collection of personal physiological data. Intrusive.</td>
<td>[67], [23], [30], [79]</td>
</tr>
<tr>
<td>(ECG &amp; EMG)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Driving Simulators</td>
<td>Controlled environment for data collection. Ability to recreate various driving scenarios. Simultaneous recording of multiple data types (vehicle dynamics, driver inputs, eye movements, physiological responses).</td>
<td>Lack of real-world driving context. Possible influence of simulator-induced effects on driver behavior. Limited external validity (findings may not generalize to real-world driving).</td>
<td>[67], [9], [44], [34]</td>
</tr>
<tr>
<td>Sensor Fusion</td>
<td>Improved accuracy through data integration. Robustness against individual sensor failures. Enhanced discrimination of driver characteristics. Comprehensive understanding of driver behavior.</td>
<td>Increased complexity in system design. Reliance on multiple sensor calibration. Potential privacy concerns with multi-modal data. Cost and maintenance of multiple sensors.</td>
<td>[67], [9], [72], [35], [30], [37], [36], [25]</td>
</tr>
</tbody>
</table>

### Table 7: Data Source Comparison

<table>
<thead>
<tr>
<th>Data source</th>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driving Simulators</td>
<td>Controlled environment for data collection. Ability to recreate various driving scenarios. Simultaneous recording of multiple data types (vehicle dynamics, driver inputs, eye movements, physiological responses).</td>
<td>Lack of real-world driving context. Possible influence of simulator-induced effects on driver behavior. Limited external validity (findings may not generalize to real-world driving).</td>
</tr>
</tbody>
</table>
Table 4
Datasets for Driver Identification

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Available features/sources</th>
<th>Detail of Dataset</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ocslab Driving dataset/HCRL Dataset (Kwak, Woo et al. 2016)</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 10, Avg trip: 46km, 23hr, 2 trips/driver. Road type: City ways, motorways, and parking spaces, Time: 8 pm - 11 pm on weekdays, Record/size: 94,401, No of Features: 51, Sampling rate of 1 Hz</td>
<td>[10], [45], [24], [11], [73], [17], [78], [5], [73], [11], [37], [32], [74], [10], [9], [80], [75], [67]</td>
</tr>
<tr>
<td>OpernXC (Ford Bug Labs 2020)</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 3, No of Features: 7, Road type: Freeway and highway</td>
<td>[5]</td>
</tr>
<tr>
<td>(Romera, Bergasa et al. 2015)</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 6, No of Features: 7, Road type: Motorway and secondary road</td>
<td>[5]</td>
</tr>
<tr>
<td>Uyanik dataset (Takeda, Hansen et al. 2011)</td>
<td>CAN-BUS/OBD-II Data, GPS, gyroscope, video camera, infrared range sensors</td>
<td>No of Drivers: 105, Gender: 17 female and 88 male, Avg trip: 25-km route of Istanbul, Record/Size: 9TB</td>
<td>[18]</td>
</tr>
<tr>
<td>VPLAB Dataset (Abut, Erdoğan et al. 2007)</td>
<td>CAN-bus, IMU, GPS, cameras, and laser scanners</td>
<td>No of Drivers: 95, Avg trip: 25km/47:68 m, Road Type: City ways, motorways, and highways, Record/Size: 9TB</td>
<td>[13], [1]</td>
</tr>
<tr>
<td>UAH-DriveSet Dataset (Romera, Bergasa et al. 2016)</td>
<td>Smartphone sensors (GPS, camera, internet, internal sensors)</td>
<td>No of Drivers: 6, Age: 20-50 years, Avg trip: 500 minutes, 25km, with each trip: 7, Road type: Motorway, secondary roads, Freq: 10hz, Speed: 90-120km/hr</td>
<td>[9], [67]</td>
</tr>
<tr>
<td>Hci-Lab Dataset (Schneegass, Pfleging et al. 2013)</td>
<td>GPS + physiological + camera + smartphone</td>
<td>No of Drivers: 10, Gender: 7 male, 3 females, Ages 23 – 57, average of 35 years old, Road Type: Highways and tunnels, Record/Size: 2.5 million, 450 MB</td>
<td>[1], [43], [9], [67]</td>
</tr>
<tr>
<td>drivedb from Physionet’s databank</td>
<td>ECG + EMG (Physiological)</td>
<td>Link: PhysioBank ATM (physionet.org)</td>
<td>[30], [79]</td>
</tr>
<tr>
<td>CASIA-WebFace data set (Yi, Lei et al. 2014)</td>
<td>Images</td>
<td>No of Drivers: 10575, Record/Size: 494 414 images</td>
<td>[20]</td>
</tr>
<tr>
<td>Beijing Taxis Dataset (Yu, Zhao et al. 2010)</td>
<td>GPS</td>
<td>No of Drivers: 8500+, Avg trip: 1 month, Road Type: 42% of all taxis in the urban area of Beijing</td>
<td>[43], [64]</td>
</tr>
<tr>
<td>Datasets</td>
<td>Available features/sources</td>
<td>Detail of Dataset</td>
<td>References</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------</td>
<td>-------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Beijing Metrobuses Dataset (Rahim, Zhu et al. 2020)</td>
<td>GPS</td>
<td>No of Drivers: 10 city bus drivers, Avg trip: 1.5 to 7.2 hours, Record/Size: 3600 - 27000 per driver</td>
<td>[43]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 20, Age: 20 and 40 years, Avg trip: 10KM, Road type: urban area, which included arterial roads, express lanes, continuous curved roads, a roundabout, and signalized intersections, Time: 9:00–10:30 am and 3:00–4:30 pm Record/size: 87,120 samples</td>
<td>[31], [10]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 20, Gender: 14 males, 6 females, Age: 20-54 years Avg trip: 12–17 minutes, 5.7km, Time: 3 pm to 6 pm, 45 days, sunny days, Traffic Type: light traffic, heavy traffic, and other complex scenarios</td>
<td>[? ], [?]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 45, Gender: 30 males and 15 females Avg trip: 6.5 km in total and takes about 12–15 min per lap Time: sunny days</td>
<td>[19]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II Data</td>
<td>No of Drivers: 8, with 5 different cars, Avg trip: 30 Km, Road type: city streets and highways</td>
<td>[78]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>ECG (Physiological)</td>
<td>No of Drivers: 100 Gender: 64 males and 36 females, Age: 23 - 34 years Affiliation: Chosun University (CU-DB)</td>
<td>[30], [79], [23]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>Smartwatch (3-axis accelerometer and 2-axis orientation sensor signals)</td>
<td>Simulator: No of Drivers: 90, Real vehicle: No of Drivers: 20, Avg trip: 1.77 km, Roat type: route included five turns, clockwise and counterclockwise</td>
<td>[44]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>Electronic Stability Control (ESC), Global Positioning System (GPS) and On-Board Diagnostics (OBD2)</td>
<td>No of Drivers: 57, Avg trip: 153.9 miles, 14 hrs, with each trip: 45 min, Road type: Urban, roundabouts, traffic lights, and streets roads, Time: 10am-12:30pm, 1–5:30pm or 7–10:30pm, London time Link: <a href="https://doi.org/10.13140/RG.2.2.14505.49765">https://doi.org/10.13140/RG.2.2.14505.49765</a></td>
<td>[36]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>Tri-axial accelerometer</td>
<td>Features: (lateral, longitudinal and vertical acceleration) No of Drivers: 10, Time: 9:00 to 11:00 or 15:00 to 17:00 for 8 days, Record: 5,500,000 each driver, Freq: 100 Hz</td>
<td>[57]</td>
</tr>
</tbody>
</table>
where $\mathbf{x}$ denotes the mean-adjusted data vector. The eigenvalues and eigenvectors are then extracted from $\text{Cov}$ via the relation

$$\text{Cov} \times \mathbf{e}_k = \lambda_k \times \mathbf{e}_k \quad \text{for} \quad k = 0, 1, \ldots, n - 1 \quad (3)$$

where $\mathbf{e}_k$ signifies the $k$th eigenvector and $\lambda_k$ the corresponding eigenvalue. To curtail the dimensionality of the feature vectors, one selects the eigenvectors associated with the most substantial eigenvalues (Shlens, 2014). The matrix of selected eigenvectors is represented as

$$M_{p} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \ldots, \mathbf{e}_p) \quad (4)$$

where $p$ corresponds to the number of dimensions retained in the PCA matrix. The transformed feature vector is then given by

$$N_p = M_{p}^T \times \mathbf{x} \quad (5)$$

with $T$ being the transpose of the mean-adjusted original input vector.

In a recent study, [83] successfully applied PCA to condense the dimensionality of audio scripts by linearly projecting the frequency components into a lower-dimensional space, retaining only 80 of the original 256 dimensions post-PCA application.

### 5.5. Mutual Information

Mutual information measures the dependency between variables. Relationship between each attribute and the target variable can be used to quantify it. Features with higher mutual information scores are considered more relevant and can be selected. Conditional mutual information maximization (CMIM) algorithm is utilized by [81] to perform feature selection for the driving fingerprint map in driver identification. CMIM is a widely recognized algorithm capable of evaluating both the relevance to the class and the intercorrelation among features. According to the CMIM results, the most important variables for correctly identifying drivers were found to be the distance to the lane, acceleration (X axis), and yaw rate.

### 6. Machine learning methods for driver Identification

Machine learning is a field within artificial intelligence that concentrates on the creation of algorithms and models capable of enabling computers to learn and make predictions or decisions autonomously, without the need for explicit programming instructions. The process involves training these models using vast quantities of data to uncover patterns, relationships, and underlying structures. Once the models are trained, they possess the ability to generalize from the training data and apply their learned knowledge to make predictions or classifications on new, unseen data. Driver identification has been addressed using different machine learning algorithms (Table 8), such as the K-nearest neighbor (KNN) [52, 15], Hidden Markov Model (HMM) [84, 68], Gaussian Mixture Model (GMM), Support Vector Machine (SVM), Decision Tree (DT), and Random Forest (RF). ANN [85, 86], extra trees [87]. We have included the most recently applied machine learning algorithm in this study.

#### 6.1. GMM

GMM, which stands for Gaussian Mixture Model, is a probabilistic model employed to analyze and represent intricate data distributions. The GMM assumes that the observed data is generated from a combination of Gaussian distributions, where each Gaussian component represents a distinct subpopulation within the data. A Gaussian Mixture Model (GMM), as introduced by [88], is conceptualized as a multivariate probability distribution approach. It is adept at modeling arbitrary distributions and has become a prevalent method in Speaker Identification (SI) systems [89]. The GMM for a feature vector distribution pertaining to a speaker $s$ is represented as a weighted linear combination of $M$ unimodal Gaussian densities $b_s(x)$, parameterized by mean vectors

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Available features/sources</th>
<th>Detail of Dataset</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-collected</td>
<td>Tri-axial accelerometer</td>
<td>No of Drivers: 25 Trips; 20,025, Each trip: 801 real trips per driver</td>
<td>[13]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>IMU (accelerometer, gyroscope, magnetometer)</td>
<td>No of Drivers: 10, Trip: 2 hours and 10 minutes, Features: Only accelerometer and gyroscope used</td>
<td>[35]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>Audio, video, CAN-bus, gas pedal, brake pedal sensor recordings, and IMU</td>
<td>No of Drivers: 25, Age: 22-57 years, Road type: urban roads, ring roads, high speed roads</td>
<td>[81]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II + GPS Record/size: 292 observations</td>
<td>Link: Home martinacimitile/Car-Data-Mining Wiki · GitHub</td>
<td>[78], [25]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II + Lidar</td>
<td>No of Drivers: 8, Road type: 4th Ring Road of Beijing</td>
<td>[33]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>OBD + GPS</td>
<td>NA</td>
<td>[29]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>CAN-BUS/OBD-II Data</td>
<td>NA</td>
<td>[76]</td>
</tr>
<tr>
<td>Self-collected</td>
<td>Driving simulator</td>
<td>NA</td>
<td>[? ]</td>
</tr>
</tbody>
</table>
Table 5
Pre-processing techniques applied in reviewed studies.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Cleaning</td>
<td>Removes noise and outliers, improving data quality. Remove 0-speed data.</td>
<td>May remove useful information along with noise if not carefully applied.</td>
<td>[63], [72], [67], [32], [9], [12], [28], [15], [31], [10], [30], [79], [44], [35], [17], [33], [25], [78], [5], [1], [36]</td>
</tr>
<tr>
<td>Feature Scaling</td>
<td>Brings features to a common scale, preventing dominance by a single feature. Helps improve convergence speed and model performance.</td>
<td>Sensitive to outliers if not robustly implemented. Scaling may not be necessary for all algorithms (e.g., tree-based models).</td>
<td>[5], [75], [72], [32], [9], [72], [77], [12], [25], [74], [28], [31], [10], [23], [30], [37], [24], [35], [73], [33], [57], [25], [11], [19]</td>
</tr>
<tr>
<td>Data Augmentation</td>
<td>Helps improve model generalization. Reduces the risk of overfitting. Increases the size and diversity of the training dataset.</td>
<td>Augmented data may not accurately represent real-world cases. Augmentation techniques must be carefully chosen to avoid distorting the original data distribution.</td>
<td>[9], [31], [35]</td>
</tr>
<tr>
<td>Time Window Segmentation</td>
<td>Captures temporal patterns, extracts relevant features. Identifies driver-specific patterns</td>
<td>Choice of window size and overlap affects results. Determining optimal window size and overlap is challenging.</td>
<td>[5], [75], [1], [67], [13], [9], [72], [77], [12], [25], [74], [15], [1], [31], [35], [73], [17], [33], [81], [57], [25], [19], [34]</td>
</tr>
<tr>
<td>Spectrogram</td>
<td>Captures important frequency patterns. Allows for the identification of unique frequency signatures</td>
<td>Spectrogram size and resolution can be sensitive to the chosen windowing parameters. Spectrogram features alone may not capture all driver-specific characteristics.</td>
<td>[13], [79]</td>
</tr>
<tr>
<td>Wavelet Transform</td>
<td>Suitable for analyzing non-stationary signals. Can capture both time and frequency information.</td>
<td>Selection of appropriate wavelet function and decomposition levels can be challenging. Computationally more complex compared to other methods.</td>
<td>[12], [35]</td>
</tr>
</tbody>
</table>

\[ m \] and a covariance matrix \( \Sigma_i \). These parameters collectively form a speaker’s model, denoted as \( \{ p_s, \mu_s, \Sigma_s \} \). The mixture weights \( p_s \) comply with the stochastic constraint \( \sum_{i=1}^{M} p_s = 1 \). For a feature vector \( x \), the density mixture for speakers is computed as:

\[
p(x|\lambda_s) = \sum_{i=1}^{M} p_s b_s(x)
\]
where each Gaussian component $b_i(x)$ is defined as:

$$b_i(x) = \frac{1}{(2\pi)^{D/2}\Sigma_i^{1/2}} \exp \left( -\frac{1}{2} (x - \mu_i)^T (\Sigma_i)^{-1} (x - \mu_i) \right)$$

(7)

and $D$ represents the dimension of the feature vector. Given a set of feature vectors $X = \{x_1, x_2, \ldots, x_T\}$ corresponding to a driving behavior signal spanning $T$ frames, the log-probability for a specific driver model $s$ is computed as:

$$L_s(X) = \log p(X|t_s) = \sum_{t=1}^{T} \log p(x_t|t_s)$$

(8)

In the context of Driver Identification (DI), the value of $L_s(X)$ is determined for each enrolled driver model $t_s$, and the model yielding the maximum likelihood is identified as the corresponding driver. During the training phase, the feature vectors are refined using the Expectation-Maximization (EM) algorithm [90, 91], a method involving iterative updates to each parameter in $\lambda$, thereby enhancing the log-probability at each iteration.

GMM is utilized in driver identification by modeling the driving patterns of different individuals as separate Gaussian components. GMM can effectively differentiate between different drivers by capturing their distinct driving behaviors and patterns by estimating the parameters of these components, such as means and covariances. This approach facilitates the development of driver identification systems capable of identifying and authenticating individuals based on their unique driving characteristics.

Some researchers have employed GMM [44, 33] for driver identification. This algorithm has shown promise in capturing drivers’ commonalities and behaviors based on various input data sources. By modeling the data as a mixture of Gaussian distributions, GMMs can effectively cluster and identify different driver profiles. Studies have leveraged GMMs to analyze features such as driving behavior, vehicle telemetry, or sensor data, providing valuable insights into driver identification. The utilization of GMM for driver identification demonstrates its effectiveness and adaptability in this specific domain. In their study, [44] present a driver identification system that specifically targets the analysis of drivers’ maneuvers, with a specific emphasis on right and left turns. The authors employ GMM-based approaches and report an error rate of approximately 18% for driver identification. [33] present a driver identification approach that considers the heterogeneity within drivers and between different drivers in car-following sequences. They utilize a GMM to model these variations. By conducting an experiment with eight drivers, collecting car-following samples of 15 seconds per sample and 10 samples per driver, the proposed method demonstrates a driver identification accuracy of 82.3%.

### 6.2. Support Vector Machine (SVM)

SVM is a widely employed machine learning algorithm utilized for classification and regression tasks [92, 93]. It operates by identifying an optimal hyperplane that efficiently separates data points from different classes. This is achieved by maximizing the margin, which represents the distance between the hyperplane and the closest data points. SVMs are powerful in dealing with both linearly separable and non-linearly separable data by employing kernel functions. These kernel functions enable SVMs to transform the input space, effectively separating complex, non-linear patterns in the data. They are renowned for their capability to handle high-dimensional feature spaces and their resistance to overfitting.

Support Vector Machines (SVM), originally introduced by [94], hinge upon the construction of an optimal hyperplane that maximizes the margin between two distinct classes [95]. For tasks involving multinomial classification, SVM adopts a structural risk minimization framework [96], which can be formalized as:

$$\min_{\w, \xi} \frac{1}{2} \w^T \w + C \sum_{i=1}^{N} \xi_i$$

(9)

The linear SVM methodology, as described by [97], entails the projection of an input feature vector $x$ into a functional space, $f(x)$, delineated as:

$$f(x) = \text{sign} \left( \sum_{i=1}^{N} \alpha_i k(x; x_i) + \beta \right)$$

(10)

In these equations, $x_i$ represent the support vectors, $\beta$ is the bias term, $\alpha_i$ are the Lagrange multipliers or weights, and $N$ denotes the total number of support vectors.

SVMs have been extensively utilized as a primary algorithm for classification tasks due to their robustness and effectiveness in handling high-dimensional data.

SVMs have been utilized by researchers to conduct driver identification studies [98, 43, 64, 12]. In [43], an event-driven framework is introduced for driver identification and impostor detection, utilizing features extracted from GPS data. The framework employs SVM as the supervised classification method. To achieve high accuracy in scenarios with many drivers, the researchers propose a modified radial basis function (RBF) kernel. The empirical evaluation demonstrates promising results, with an average recall of 94% for ten drivers and 90% for 24 drivers. Furthermore, the framework achieves impostor driver detection with over 90% accuracy when dealing with three or more drivers. [12] utilized CAN bus data and employed an SVM classifier to perform driver classification. The binary classification obtained up to 95.07% accuracy, while the multiclass classification achieved 89.06% accuracy.
### Table 7
Feature selection techniques for driver identification

<table>
<thead>
<tr>
<th>Technique</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Univariate Feature Selection</td>
<td>Simple and computationally efficient. Can handle a large number of features. Can uncover individual feature relevance. Statistical significance of features can be determined.</td>
<td>Ignores feature interactions and dependencies. Does not consider the combined effect of features. May overlook useful features if their relevance is dependent on interactions with other features. Assumes features are independent of each other.</td>
<td>[37]</td>
</tr>
<tr>
<td>PCA</td>
<td>Reduces dimensionality while retaining most of the information. Allows for visualization of data. Captures major patterns in the data. Reduces computational complexity. Works well with linear models.</td>
<td>May not be suitable for capturing complex non-linear relationships. Interpretability of principal components may be challenging. Does not provide feature rankings or insights into individual feature relevance. Requires standardization of features. May lose some discriminative power if the important features are not the major components.</td>
<td>[5], [56], [13], [77], [74], [37], [45], [24], [11], [19]</td>
</tr>
<tr>
<td>Tree-based Feature Importance</td>
<td>Considers feature interactions and non-linear relationships. Handles missing values and outliers. Provides feature rankings based on importance. Works well with categorical and numerical features. Robust to feature scaling. Can handle high-dimensional data.</td>
<td>May prioritize high cardinality categorical variables over continuous variables. Prone to overfitting if the model is excessively complex. Can be biased towards features with high cardinality. Tree-based models can be computationally expensive for large datasets. Feature importance can vary between different tree ensemble methods.</td>
<td>[67], [32], [74], [28], [37], [24], [57]</td>
</tr>
<tr>
<td>Correlation-based Feature Selection</td>
<td>Identifies linear relationships between features and the target. Quick and straightforward to compute. Can identify highly correlated features for removal. Provides insights into feature dependencies.</td>
<td>Only captures linear relationships, may miss non-linear dependencies. Ignores feature interactions. May not capture the true relevance of features if there are non-linear relationships. Does not provide feature rankings based on importance. May overlook informative features that are not strongly correlated with the target variable. Requires continuous numerical features for correlation analysis.</td>
<td>[5], [67], [32], [9], [28], [36], [35], [1], [16, 17], [25], [77]</td>
</tr>
<tr>
<td>Mutual Information</td>
<td>Considers the statistical dependence between features and the target variable. Captures nonlinear relationships.</td>
<td>May overlook higher-order feature interactions. Sensitive to the discretization or binning of continuous variables.</td>
<td>[81]</td>
</tr>
</tbody>
</table>

### 6.3. Decision Tree (DT)

A Decision Tree is a widely adopted machine learning algorithm that employs a tree-like model for making decisions or predictions. It organizes data into a hierarchical structure consisting of nodes and branches. Each internal node within the tree represents a feature or attribute, whereas each branch corresponds to a decision made based on that particular attribute. The tree structure is constructed by iteratively dividing the data using various features in a recursive manner, aiming to create subsets that are as pure as possible in terms of the target variable. Decision Trees can handle both classification and regression tasks and are interpretable, allowing users to understand the decision-making process. They are known for their versatility, ease of implementation, and capability to handle both categorical and numerical data. [45] present a machine learning approach by analyzing vehicle sensors. They utilize a DT algorithm to assess the effectiveness of their proposed method in driver identification. The dataset [14] used consists of 51 distinct features extracted from 10 different drivers, and the evaluation demonstrates a precision and recall of 99%.
6.4. Random Forest (RF)

RF is a widely used machine learning algorithm that constructs an ensemble of decision trees [99]. By combining the predictions of multiple individual decision trees, it enhances the accuracy and robustness of the overall predictions [100]. Each decision tree is trained on a random subset of the training data, and during the prediction phase, the final result is determined by aggregating the predictions of all the trees. RF is known for its effectiveness in handling high-dimensional data, as well as capturing intricate relationships within the data. Additionally, it serves as a useful tool for mitigating overfitting, a common challenge in machine learning. It is commonly employed algorithm for both classification and regression tasks, and its versatility and performance have made it widely adopted in various domains.

7. Deep Learning methods for driver identification

Deep learning is a branch of machine learning that utilizes artificial neural networks to learn patterns and make predictions from data [101]. Inspired by the human brain, these neural networks are designed to mimic the structure and functionality of the brain’s interconnected neurons, excelling at detecting intricate patterns within the data [102]. Deep learning has been successfully employed to address various challenges, including object recognition, speech recognition, face recognition, sentiment analysis, and numerous classification tasks [103]. In recent years, deep learning approaches have shown promising performance across different domains, including network communication failure recovery [104], Cyber security [105, 106, 107], intelligent transportation systems and smartcity [108, 109, 1]. Different types of deep neural networks (Table 9) include Convolutional Neural Networks, Recurrent Neural Networks, and deep autoencoders, commonly employed for driver identification tasks. Deep autoencoders, specifically, are neural networks designed to reconstruct the input at the output, primarily utilized for reducing the dimensionality of data.

7.1. Deep Neural Network (DNN)

DNN is an artificial neural network that comprises multiple hidden layers positioned between the input and output layers, as depicted in Fig. 6. It is called “deep” because of its depth, meaning it has a large number of layers compared to traditional neural networks [110]. Each layer in a DNN contains multiple interconnected neurons that perform computations on the input data. The architecture of a DNN typically adopts a feedforward structure, where information propagates in a unidirectional manner, starting from the input layer and passing through the hidden layers before reaching the output layer. In each hidden layer, every neuron receives inputs from the preceding layer, calculates a weighted sum of these inputs, applies an activation function to the sum, and then transmits the result to the next layer in the network. This iterative process continues for each layer until the output layer generates the final prediction or classification based on the transformed inputs from the preceding layers.

Training a DNN consists of two primary stages: forward propagation and backpropagation. During the forward propagation phase, the input data is fed into the network, and predictions are generated as the data flows through the network. The disparity between the predicted outputs and the actual outputs is assessed by employing a loss function. Subsequently, backpropagation is employed to compute the gradients of the loss function in relation to the weights and biases of the network. The gradients calculated in the previous step are utilized to update the weights and biases of the network through optimization algorithms such as gradient descent.

\[
\Delta \omega_i(j + 1) = \Delta \omega_i(j) + \eta \nabla C / (\sigma \omega_i(j)) \tag{11}
\]

In the given equation, the learning rate is denoted by \( \eta \), the weight is represented by \( \omega_i(j) \), and the cost function associated with the weights is denoted by \( C \).

[25] identified the driver within a group of 10 drivers by employing DNN, achieving an accuracy of 95%. [24] utilized feature reduction techniques, specifically principal component analysis and random forest, in combination with the DNN model. The proposed RF-DNN model achieved an accuracy of 97.05%, while the PCA-DNN model achieved an accuracy of 95.55% in driver identification. The proposed DNN model by [72] successfully achieved driver identification with a testing duration of 13 seconds and an accuracy rate of 81%. Their data source consists of two smartphone sensors, accelerometer, and GPS. The study conducted by [15] reported an impressive accuracy rate of 99% in their driver identification system by utilizing DNN for histograms of acceleration data from the accelerometer sensor.

7.2. Convolutional Neural Network (CNN)

CNN [111] is a specialized deep learning architecture designed specifically to process and analyze structured grid-like data, such as images or sequential data [112]. While CNNs are commonly utilized for computer vision applications, their applicability extends beyond that domain. CNNs can be employed in various fields, including natural language processing and speech recognition [20]. The fundamental concept underlying CNNs involves utilizing convolutions, a mathematical operation that enables the integration of input data with learnable filters or kernels [113]. This process allows for the extraction of pertinent features that are essential for the network’s learning and decision-making. The process of convolution entails sliding the filters across the input data, performing element-wise multiplications, and subsequently summing the results to generate a feature map. This process is repeated for different filters, creating multiple feature maps that capture various patterns and spatial hierarchies within the input.

A CNN is commonly structured with multiple layers organized hierarchically [114]. The fundamental components of a CNN comprise convolutional layers, activation functions, pooling layers, and fully connected layers. Convolutional layers play a crucial role in a CNN as they apply filters to the input data, extracting relevant features. Activation functions are utilized to introduce non-linearity to the network, enabling it to capture complex relationships within the data. Pooling layers perform downsampling of the feature maps, decreasing their spatial dimensions while preserving essential features. Fully connected layers establish connections between every neuron in a given layer and every neuron in the subsequent layer, facilitating the representation of high-level features and final classification. Fig. 7 depicts the CNN structure.

[115] introduced the initial CNN architecture in 1998, primarily for recognizing handwritten digital images. When it comes to driver identification, the key challenge lies in extracting and capturing the distinctive features of each driver’s driving behavior. CNNs offer a viable solution as they have the ability to automatically extract relevant features from driving data, thereby facilitating effective driver identification. In their study, [11] directed their attention towards utilizing a CNN model to differentiate between drivers. Their architecture comprises a convolutional layer, subsequently followed by max-pooling and a fully connected layer. Through the extraction of more than 20 features from CAN-bus signals, encompassing accelerator and brake pedal information, as well as engine speed, they achieved an impressive accuracy exceeding 95% for two distinct vehicle types. In [20] presented a real-time CNN model called the three-stream convolution network (TS-CNN) for driver identity verification. Their model, based on the lightweight SqueezeNet architecture, effectively captured features from low-quality face images. Through their experiments, they attained an impressive average verification accuracy of 87.44%.

[10] developed a driver identification system employing a one-dimensional convolutional neural network (1-D CNN) architecture. The 1-D CNN structure encompasses two convolutional-pooling layers, a fully connected layer, and a SoftMax layer. Their approach outperformed SVM, MLP, and LSTM models in terms of macro F1 score, achieving an impressive 99.1% accuracy in identifying 20 drivers. The
### Table 8
Machine learning algorithms for Driver Identification

<table>
<thead>
<tr>
<th>Method</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM</td>
<td>Can model complex data distributions. Can handle overlapping or multimodal clusters. Probabilistic framework allows uncertainty modeling.</td>
<td>Can be computationally expensive for large datasets. Initialization can impact the quality of results. Can struggle with high-dimensional data.</td>
<td>[44], [33]</td>
</tr>
<tr>
<td>DT</td>
<td>Easy to understand and interpret. Can handle both numerical and categorical features. Non-parametric and distribution-agnostic. Provides feature importance.</td>
<td>Prone to overfitting, especially with deep trees. Sensitive to small variations in the training data. Lack of robustness when dealing with noisy or irrelevant features. Limited generalization ability.</td>
<td>[45], [29]</td>
</tr>
<tr>
<td>RF</td>
<td>Effective in handling complex interactions. Robust against overfitting. Provides feature importance measures.</td>
<td>Computationally expensive during training. Lack of interpretability for individual trees. Requires careful hyperparameter tuning. May struggle to generalize well with unrepresentative training data.</td>
<td>[67], [43], [17], [16], [18], [36], [32], [57], [63]</td>
</tr>
</tbody>
</table>

1-D CNN demonstrated stable performance and robustness, making it a promising choice for driver identification tasks. [74] introduced an improved version of CNN called residual convolution network (RCN) based on a CNN for real-time driver identification. The RCN architecture is comprised of four layers of conventional CNN, succeeded by max-pooling and dropout layers. Through their method, they achieved remarkable performance with a training time of less than two hours, achieving an accuracy of 99.3%. Comparative analysis demonstrated that the RCN approach outperformed traditional machine learning algorithms and exhibited slightly superior performance compared to an RNN and a combined CNN-RNN model. [30] introduced a driver identification system based on a single CNN architecture that utilizes 2D spectrogram multidimensional features. The proposed method involves converting segmented ECG cycles into spectrograms, optimizing time-frequency resolution, resizing the resulting 2D image, and performing driver identification using a single CNN. Additionally [30], proposed a multi-stream CNN approach that leverages ECG and EMG signals, achieving an impressive accuracy rate of 98.9% for driver recognition. [81] utilized a CNN and conditional user information maximization to derive driving fingerprint heat maps that serve as representations of distinct driving characteristics. Through this approach, they attained a remarkable accuracy of 93.5% in driver identification.

### 7.3. Recurrent Neural Network (RNN)

RNN is a specialized form of artificial neural network created to handle sequential data or data with temporal dependencies [116]. In contrast to feedforward neural networks, RNNs incorporate internal loops within their architecture, enabling them to retain information and maintain temporal dependencies over time. This enables them to capture and analyze patterns in sequential data by incorporating information from both the current input and previous time steps. At every time step, an RNN receives an input, generates an output, and updates its hidden state. The hidden state acts as the memory of the network, enabling it to store information from previous inputs. This hidden state is fed back into the network at the next time step, making RNNs capable of modeling sequences of arbitrary length. Fig. 8. shows the architecture of RNN. The key component of an RNN is the recurrent connection, which enables the flow of information from one time step to the next. This connection allows the network to learn long-term dependencies and contextual information from the input sequence. By considering the context and history of the data, RNNs have shown great efficacy in tasks like language modeling, speech recognition, machine translation, and time series prediction.
7.3.1. Long Short-Term Memory (LSTM)

A widely used form of RNN is the LSTM network, which tackles the issue of vanishing gradients in conventional RNNs through the use of memory cells and gates that control the flow of information [117]. These mechanisms enable LSTMs to selectively retain and retrieve information from previous steps, making them highly effective in capturing long-term dependencies in sequential data.

[19] utilized the LSTM network to capture driving behavior patterns of drivers, enabling their designed network to possess memory capabilities and learn driving behavior characteristics that are dependent on time. This approach allows the network to learn the relationships among multiple features and effectively extract valuable insights from the data. Further, [37] introduced an optimized deep learning model for driver identification, leveraging LSTM networks. The architecture of the proposed model comprised an input layer, two LSTM layers with hidden layer dropout regularization, and a dense layer with a softmax activation function serving as the output layer. Hyperparameter tuning was performed using the Crow Search Algorithm (CSA), an optimization algorithm inspired by nature. Through this optimization process, an impressive accuracy of 99% was achieved.

Research conducted by [23] utilized LSTM networks for driver identification. By analyzing the normalized Electrocardiogram (ECG) signal, they achieved an impressive accuracy of 96.5%. In a separate study on driver identification, [76] utilized an LSTM network architecture consisting of four hidden layers, comprising two LSTM layers and two dense layers. The LSTM layers were comprised of 64 hidden units, while the first dense layers contained 32 hidden units, and the second dense layers comprised of 1 hidden unit. By leveraging real-world driving data, the researchers achieved an impressive overall accuracy of 81% ± 7.5%. Furthermore, [99] utilized LSTM, results indicated that bidirectional LSTM networks outperformed forward LSTM networks, demonstrating an average accuracy improvement of 7% in driver identification performance. This highlights the effectiveness of bidirectional LSTM architectures in capturing temporal dependencies and enhancing the accuracy of driver identification tasks.

7.4. Deep Autoencoder (DAE)

A deep autoencoder is a sophisticated neural network architecture that incorporates multiple hidden layers between the encoder and decoder components of an autoencoder [118]. It extends the traditional autoencoder by allowing the network to learn hierarchical representations of the input data. The encoder maps the input to a lower-dimensional representation. By analyzing the normalized Electrocardiogram (ECG) signal, they achieved an impressive accuracy of 96.5%. In a separate study on driver identification, [76] utilized an LSTM network architecture consisting of four hidden layers, comprising two LSTM layers and two dense layers. The LSTM layers were comprised of 64 hidden units, while the first dense layers contained 32 hidden units, and the second dense layers comprised of 1 hidden unit. By leveraging real-world driving data, the researchers achieved an impressive overall accuracy of 81% ± 7.5%. Furthermore, [99] utilized LSTM, results indicated that bidirectional LSTM networks outperformed forward LSTM networks, demonstrating an average accuracy improvement of 7% in driver identification performance. This highlights the effectiveness of bidirectional LSTM architectures in capturing temporal dependencies and enhancing the accuracy of driver identification tasks.
latent space through successive hidden layers, capturing different levels of abstraction. The decoder then reconstructs the original input from the latent space, gradually increasing the dimensionality. The deep autoencoder learns to reconstruct the input by minimizing a reconstruction loss, enabling it to capture complex patterns and extract meaningful features.

Several approaches aim to extract hidden features from driving data for the specific objective of driver identification. [80] introduced using deep nonnegativity constrained autoencoder for feature extraction purposes. Their proposed method included window size optimization for data segmenting and extracting hidden features through a layer-by-layer greedy approach. The extracted features were passed through a SoftMax classifier, which generated probabilities for each driver, enabling driver identification. While this method requires substantial computational resources for data calculation and training, it exhibits high accuracy in performance. Overcoming the challenge of graphics processing unit (GPU) usage could potentially broaden the applicability of these deep learning algorithms to diverse types of data.

7.5. Hybrid Model

To maximize the advantages of different base models, researchers have explored using hybrid models for multiple applications. [75] proposed a hybrid deep learning architecture based on an attention-based combination of CNN, Gated Recurrent Unit (GRU), and LSTM. This approach allows the models to be trained using raw data without manual feature engineering. The neural network layers extract the hidden temporal dependencies within the data. The advantage of this approach is that it leverages CNNs to automatically extract relevant features without the need for explicit feature selection. However, a potential drawback is that the models require many training epochs. A hybrid model called GAN-SGM is presented for driver identification in the study conducted by [35]. The Generative Adversarial Network GAN is utilized for data augmentation, while the Stacked Generalization Method (SGM) employing KNN, MLP, SVM, RF, and LR is used for classification. The hybrid GAN-SGM approach demonstrates exceptional performance with an accuracy of 97%, precision of 98%, recall of 97%, and F1-measure of 97%, surpassing conventional machine learning techniques. [13] have directed their attention towards the utilization of DL techniques for driver identification and verification. They have introduced a hybrid model architecture called CNN-RNN for driver identification. This architecture incorporates ResNet-50 as the initial component, followed by two Stacked Gated Recurrent Units (SGRUs). A Dense layer and a final SoftMax layer are included in the model as well. For driver verification, the authors have proposed the utilization of Siamese Neural Networks and Triplet Loss Training. In their implementation, ResNet-50 is employed as the CNN feature extractor, and GRUs are used for modeling the recurrent dynamics. Driver verification received an F1 score of 74.09%, while driver identification produced top-1 and top-5 accuracy results of 71.89% and 92.02%, respectively.

Given that the sensors used to collect data for driver identification are not entirely immune to malfunctions or failures [9], it is common to encounter anomalies within the collected sensor data. Therefore, it is crucial to have a well-defined strategy for detecting and correcting these anomalies to ensure stable and reliable results. [9] investigated the importance of detecting and correcting data anomalies in driver identification schemes by integrating CNN and LSTM. They evaluated the performance of their method by comparing the results with and without data anomaly detection techniques. The findings demonstrated that incorporating anomaly correction, even with varying anomaly rates, led to improved results. However, a notable limitation is the requirement for a substantial amount of driving data (typically training samples) to achieve such high-performance outcomes using these approaches.

Obtaining accurately labeled data from multiple drivers, including both vehicle owners and thieves, for supervised learning is practically impossible. Therefore, anomaly detection has emerged as a preferred approach for vehicle theft detection. In their work, [5] propose a redesigned GAN model called Convolutional Long short-term GAN (CLGAN) for theft detection. The CLGAN model combines LSTM and a CNN and only requires accessible data from the owner of the vehicle. The average accuracy achieved by the proposed CLGAN model is an impressive 98.5%, with the highest accuracy of 92.1% observed across various road types. However, it should be noted that the CLGAN model incurs increased time and memory costs. To address the need for immediate driver behavioral analysis in real time, the focus should be on enhancing the system’s ability to identify the thief as early as possible.

Various approaches have been explored to address the identification of illegal or unauthorized drivers. [77] proposed a scheme for automobile driver fingerprinting using a combined model of CNN and Support Vector Domain Description (SVDD). Their approach incorporates an unsupervised anomaly detection method, SVDD, to identify imposters. The authors recommended employing the output of a CNN model as input for the anomaly detection procedure. The findings indicated an accuracy exceeding 95% for two different vehicle categories. [1] proposed a methodology called the Siamese Temporal Convolutional Network, which combines the concept of Siamese networks with a 1D temporal convolution architecture for driver verification with a limited duration of...
driving data. They extended the driver verification approach to include the detection of impostors in an open-set scenario, eliminating the requirement for model re-training. However, a limitation of their method is that the decrease in accuracy as driver increases. [19] introduced a multi-task learning network (MTL) that addresses the tasks of detecting illegal drivers, identifying legal drivers, and evaluating driver behavior. Their MTL network consisted of LSTM, SVDD, and feedforward neural network components. The LSTM module was used to identify driving behaviour traits, and its output was used to input the SVDD and feedforward neural networks to detect unlawful drivers and identify legitimate drivers. The outcomes showed that the proposed MTL network achieved high accuracy, broad application, and strong resilience across the three tasks, enabling parallel learning and lowering time and space expenses. [1] proposed design of a deep learning architecture, particularly a stacked encoder model for driver identification. This architecture utilizes dilated causal convolutions, deep residual blocks, and triplet loss to effectively extract driver embeddings from data without supervised learning. These embeddings are then used to differentiate among drivers belonging to a specific group. The proposed solution achieves an average accuracy of 94.7% for two-way identification and 96% for three-way identification. Notably, their approach outperforms supervised methods when utilized with data that has limited labelling. Furthermore, [73] propose a deep meta-learning approach called "MetaARNet" to solve the issues with deep model adaptation with varying numbers of drivers (output classes) when there are just a small number of driving samples available. Their method, referred to as MetaFSDI (Meta Few-Shot Driver Identification), leverages deep meta-learning to enable fast adaptation in such scenarios. The authors adopt a meta-learning framework due to its advantage in reusing features for domains with similar tasks. To ease the development of generalised driving style representations, they introduce an autoencoder regularised convolutional neural feature extractor network (ARNet). This network aids in capturing essential characteristics of driving behaviors. To evaluate the performance of their approach, experiments are conducted on 3-way and 5-way problems using 5-shot and 1-shot approaches, respectively, using a publicly available driving dataset. The results demonstrate that their method outperforms traditional meta-learning algorithms, ensuring better adaptation performance in few-shot driver identification scenarios. An ensemble deep learning framework proposed be [31], combines a Modified M 1-D CNN and bidirectional long short-term memory (BLSTM) to improve robustness, prevent overfitting, and enhance generalization. After employing four different data augmentation techniques, the proposed model attained an accuracy exceeding 50% when limited to one minute of driving data. In another study, [28] developed an ensemble approach known as stacked generalization, which combines multiple models. They utilized stacked generalization with various machine learning methods, including KNN, SVM, DT, and RF, using Floating Car Data (FCD) collected from smartphones. The system achieved an identification accuracy of 88%. However, it should be noted that their approach was not scalable for all drivers, suggesting potential limitations in its application.

8. Large Language model for driver Identification

Large Language Models (LLMs) like OpenAI’s Chat-GPT are revolutionizing Natural Language Processing (NLP). These models, based on advanced deep learning architectures such as Transformers, excel in understanding and generating language, rivaling human-like conversation and content creation abilities. Their sophistication is rooted in processing vast datasets, where larger data inputs enhance their output’s nuance and accuracy. A key aspect of LLM development is the labor-intensive text labeling process. Human annotators play a vital role in this phase, organizing and tagging enormous data volumes to provide structured context. This meticulous task significantly enhances LLMs’ language processing capabilities.

Recently, LLMs have made a significant leap, gaining the ability to autonomously generate labeled datasets. This breakthrough has transformed their development, shifting away from the previously manual-intensive labeling process. This automation not only streamlines but also accelerates LLM development, marking a pivotal change in their evolution. Our previously paper [119] has already given overview to reader for better understanding the architecture and performance comparison of different large language models such Chat GPT, Google Bert etc.

8.1. Framework for driver Identification using LLM

Applying LLM in driver identification involves integrating the model’s natural language processing capabilities with driver-related data to enhance accuracy and personalization. This section explain the framework components used for driver identification.

8.1.1. Data Collection and Preprocessing:

Gather a diverse and comprehensive dataset of driving-related information, including driving behavior data, vehicle telemetry, sensor readings, driver profiles, and any other relevant data points. Ensure that the data is labeled with driver identification information.

- Gather multimodal data:

  - Numerical data: Collecting numerical data is an important aspect of understanding driving patterns and behaviors. The data collected may include GPS coordinates, vehicle speed, acceleration, and various sensor readings. Such quantitative insights can be used to identify patterns and trends in driver behavior, which can be essential for improving driving safety and efficiency. By analyzing this data, we can develop better strategies for fleet management, route optimization, and fuel consumption. It can also help in identifying areas where driver training and coaching may be required. Overall, collecting and analyzing numerical data can provide valuable insights into various aspects of driving, which can lead to a safer and more efficient driving experience.

  - Voice data: One crucial type of data that can be collected is voice data, which includes audio recordings of the driver’s interactions such as commands, conversations, or phone calls made while driving. This type of data can offer valuable insights into the driver’s communication style, verbal cues, and behavior behind the wheel. By analyzing this data, it is possible to gain a deeper understanding of how drivers interact with their vehicles and the environment around them. This information can be used to help improve driver safety, optimize vehicle performance, and develop more effective communication tools for drivers.

  - Text data: Gathering and analyzing various sources of data such as driving logs, navigation history, social media posts, and text messages can provide valuable insights into the habits and preferences of the driver. For instance, compiling driving logs can help identify frequent destinations, routes, and driving patterns. Navigation history can reveal the places the driver often visits, while social media posts and text messages can offer additional context about their interests, activities, and social connections. By analyzing this text data, it is possible to gain a better understanding of the driver’s behavior, preferences, and tendencies, which can be useful for various purposes such as improving safety, personalizing services, or detecting anomalies.

  - Image data: One of the key components of advanced driver assistance systems (ADAS) and autonomous vehicle technology is the collection and analysis of visual data. This includes gathering various types of images such as facial images, in-vehicle camera captures, and road scene images.
These visual data inputs can be used to recognize and analyze the driver’s physical features, as well as their interactions within the vehicle. By gathering this information, ADAS and autonomous vehicle systems can make more informed decisions and take appropriate actions to ensure the safety of the driver and passengers.

- Accurately labeling each data point with the corresponding driver ID is a crucial step in building a robust driver behavior model. When we label each data point with its respective driver ID, we enable the model to learn and identify patterns that are unique to each driver. This process is essential because it helps the model to differentiate between different driving styles and behaviors. In the absence of accurate labeling, the model may not be able to associate the patterns it identifies with specific drivers, which can lead to inaccurate predictions and results. Therefore, ensuring that each data point is accurately labeled with its corresponding driver ID is critical to building a reliable and effective driver behavior model.

- Preprocess data: In order to prepare the collected data for LLM (Language Model) input, it is important to preprocess it to ensure that it is in a suitable format. This involves several steps, including tokenization and conversion of data into sequences or batches. Tokenization refers to the process of breaking down the data into smaller units, such as words or phrases, which can then be analyzed by the LLM. This is particularly important for textual data, but can also be applied to numerical data, which may need to be segmented into appropriate units. Once the data has been tokenized, it can be converted into sequences or batches, which are essentially groups of data points that can be processed by the LLM in parallel. This can help to improve the speed and efficiency of the model, as well as ensure that the input data is properly formatted and aligned with the LLM’s requirements. Overall, preprocessing the collected data is an essential step in preparing it for LLM input, and can help to ensure that the model is able to analyze the data accurately and effectively.

  - Numerical data: When working with numerical data, it is crucial to ensure that the data is clean and free of any outliers or anomalies that may lead to inaccurate results. Therefore, it is recommended to remove any such data points before proceeding with analysis. Additionally, to ensure consistency, it is helpful to normalize the values within the data set. This step can be followed by extracting meaningful features or creating time-series representations for better model interpretation. By doing so, one can obtain a better understanding of the data and make more informed decisions.

  - Voice data: One of the most common tasks in modern speech technology is converting voice recordings into text. This process is carried out through Automatic Speech Recognition (ASR) technology, which uses complex algorithms to transcribe spoken language into written words. Additionally, it is possible to extract specific audio features from the recordings that may be relevant to the task at hand, such as tone or pitch. This information can be used to further analyze the speech data and gain insights into the speaker’s emotions or intentions.
Table 9
Deep Learning Techniques for Driver Identification

<table>
<thead>
<tr>
<th>Method</th>
<th>Strengths</th>
<th>Weaknesses</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>Capable of learning complex non-linear relationships between input features and driver identification. Versatile and widely applicable to various domains. Handles high-dimensional data effectively.</td>
<td>Requires careful hyperparameter tuning and architecture design. Susceptible to overfitting, especially with limited training data. Prone to getting stuck in local optima during training. Computational complexity increases with the number of layers and neurons.</td>
<td>[24], [25], [72], [15]</td>
</tr>
<tr>
<td>CNN</td>
<td>Effective in capturing spatial features. Robust to input variations. Hierarchical feature extraction. Parameter sharing and regularization.</td>
<td>Limited temporal modeling. Lack of interpretability. Computationally expensive.</td>
<td>[74], [80], [81], [30], [10], [20]</td>
</tr>
<tr>
<td>LSTM</td>
<td>Effective in capturing long-term dependencies. Mitigates the vanishing gradient problem. Handles sequential data effectively.</td>
<td>Computationally expensive. Difficulty in parallelization. May suffer from overfitting if not properly regularized.</td>
<td>[34], [37], [23], [76]</td>
</tr>
<tr>
<td>DAE</td>
<td>Efficiently learns low-dimensional representations of data. Reduces dimensionality and extracts important features. Preprocesses high-dimensional sensor data effectively.</td>
<td>Unsupervised learning requires labeled data for driver identification. Reconstruction loss may not directly correlate with driver identification accuracy. Sensitivity to hyperparameter tuning and network architecture.</td>
<td>[80]</td>
</tr>
<tr>
<td>Hybrid Model</td>
<td>Captures both spatial and temporal features effectively. Hierarchical feature extraction. Memory of past information. Variable-length inputs. Handles sequential and spatial data.</td>
<td>Computationally expensive due to the combination of CNN and RNN. Requires careful architecture design and hyperparameter tuning. Potential difficulty in interpretability of combined CNN and RNN components.</td>
<td>[9], [75], [13], [5], [77], [35], [1], [31], [28]</td>
</tr>
</tbody>
</table>

– Text data: To prepare text data for natural language processing (NLP), it’s essential to apply several pre-processing techniques. These techniques include removing any irrelevant or noisy information from the text, correcting any spelling or grammatical errors, and tokenizing the text to split it into individual words. Additionally, normalization techniques such as stemming or lemmatization can be applied to transform words to their base form and reduce the number of unique words in the text. By performing these pre-processing steps, the text data is more suitable for further analysis using NLP techniques.

– Image data: When working with image data, it is often necessary to ensure uniformity in terms of size and aspect ratio. This can be achieved through techniques such as resizing and cropping. In addition, it is important to normalize pixel values to ensure that they fall within a specific range. This can be done by subtracting the mean pixel value from each pixel and dividing by the standard deviation. To extract relevant features from images, techniques such as facial recognition or object detection can be employed. Facial recognition can be used to identify specific individuals within an image or to detect facial expressions. Object detection can be used to identify specific objects within an image, such as cars or trees. These techniques can be used to extract features that can be used in further analysis or to train machine learning models.

8.1.2. Data Encoding:
– Represent all modalities as text:
  * Numerical data: When dealing with data analysis, one can use a technique called feature vector encoding. This method involves transforming data points into a set of numerical values that can be easily processed by algorithms. Alternatively, one can generate a statistical description of the data that can be interpreted as text. This method involves summarizing the data using statistical measures such as mean, median, and standard deviation, and then presenting the results in a textual form. Both of these techniques can be useful in different scenarios, depending on the nature of the data and the specific goals of the analysis.
    - Feature vector encoding.
    - Statistical description generation.
  * Voice data: Automated Speech Recognition (ASR) transcripts are computer-generated transcriptions of spoken language. They are widely used in various scenarios, such as generating subtitles for videos, transcribing interviews or meetings, and enabling voice-to-text features in smartphones and other devices. ASR technology works by analyzing the audio input and converting it into written text. However, the accuracy of ASR transcripts can vary depending on several factors, such as the clarity of speech, background noise,
the quality of the ASR technology, and the complexity of the language used. For example, ASR technology may struggle to accurately recognize accents or dialects that are different from the standard language model. Similarly, the presence of background noise or overlapping speech can cause errors in the transcription. The quality of ASR technology can also vary depending on the vendor and the type of algorithm used. Some algorithms may be better suited for certain languages or accents than others. Therefore, it’s important to carefully evaluate the accuracy of ASR transcripts before using them in critical applications.

* The text data provided is already preprocessed, which means that it has undergone a series of cleaning and tokenization processes. Cleaning refers to the removal of unwanted characters, such as special symbols, punctuation marks, and stop words, while tokenization involves breaking down the text into smaller units, such as words or phrases, to facilitate analysis or machine learning tasks. Hence, the given text is ready for further processing or analysis without any additional cleaning or preprocessing steps.

* Image data: Image captioning is a computer vision task that involves generating a textual description of an image. It utilizes machine learning models to analyze visual features of an image and then generates a sentence or a paragraph that describes the contents of the image in a natural language format. By using outputs from an image captioning model, it is possible to transform visual data into textual descriptions. This can be especially useful in applications such as image search, robotics, and assistive technologies for individuals with visual impairments.

8.1.3. Large Language Model (LLM) as Driver Identification Model:

* Choose a suitable LLM: When deciding on an LLM model, it’s important to choose one capable of processing diverse textual inputs and encoding complex patterns. This is particularly important when dealing with large amounts of data or complex language structures. To ensure that you’re selecting the most appropriate model for your needs, consider looking for models that are either inherently multimodal or have strong text classification capabilities. Multimodal models can process multiple input forms, such as text, images, and audio, while text classification models are specifically designed to categorize and analyze text data accurately. By evaluating these options carefully, you can feel confident that you’re selecting an LLM model that will best meet your needs and produce the most accurate and valuable results.
  - Capable of processing diverse textual inputs and encoding complex patterns.
  - Consider multimodal LLMs or strong text classification models.

* Fine-tune the LLM: To achieve optimal performance of the LLM, you can fine-tune it to adapt to your specific use case. This can be done by training the LLM on the encoded multimodal data while adjusting various hyperparameters, training strategies, and data sampling techniques. By doing so, you can customize the LLM to better suit your specific needs and improve its accuracy and efficiency. Keep in mind that the process of fine-tuning may require some experimentation and iteration to find the best combination of settings for your particular use case.
  - Train on the encoded multimodal data for driver identification.
  - Adjust hyperparameters, training strategies, and data sampling.

8.1.4. Evaluation and Refinement:

* Measure performance: When it comes to measuring the performance of a model, it’s important to use metrics that provide a comprehensive evaluation. These metrics should include accuracy, precision, recall, and F1-score. By evaluating the performance of the model across different data modalities and among various driver groups, we can ensure that the model is both robust and accurate. This allows us to identify any areas where the model may be struggling and make improvements to increase its overall effectiveness. Additionally, by measuring performance across different data modalities and driver groups, we can ensure that the model is capable of handling a wide range of scenarios and that it is not biased towards any particular group or situation. Overall, using a variety of metrics to evaluate model performance is crucial for ensuring that the model is effective and reliable.
  - Accuracy, precision, recall, F1-score, etc.
  - Assess performance across modalities and driver groups.

9. LLM Application’s for Driver Identification

LLM like GPT-4, developed by OpenAI, is an advanced version of the Generative Pre-trained Transformer (GPT) series [120], exhibit advancements in natural language processing capabilities. With a potentially larger number of parameters, GPT-4 further enhances its contextual understanding, generation abilities, and application to various tasks. It could be utilized for driver identification in various applications. Leveraging its advanced language understanding and multimodal analysis capabilities, LLM can analyze driver identification data, including text, voice, and image inputs, to verify and authenticate the identity of drivers. By processing diverse data sources, LLM could contribute to enhance accuracy and reliability in driver identification processes, aiding in access control, personalized settings, driving behavior analysis, and security measures. This section explains the use of LLM’s in driver identification.

9.1. Law Enforcement

9.1.1. License Plate Recognition

LLM can analyze images or video feeds from surveillance cameras or police vehicles to recognize and interpret license plates. By leveraging its language understanding capabilities, the model could extract relevant information from the license plates, such as vehicle registration details or owner information, aiding in driver identification.

9.1.2. Voice Analysis

LLM has the capability to analyze and identify distinct voice patterns or characteristics present in audio recordings. This feature can be utilized for the purpose of matching a driver’s voice with those of already known individuals or to provide supplementary evidence for identification purposes.
9.1.3. Integration with Law Enforcement Databases
The integration of LLM with law enforcement databases, such as criminal records or watchlists, enables the model to access and analyze this information. This analysis can help the model identify drivers who have previous records or are associated with ongoing investigations.

9.2. Usage-Based Insurance
9.2.1. Personalized Insurance Plans
Insurance companies could benefit from LLM’s ability to create custom insurance plans for drivers based on their driving behavior, historical data, and other relevant factors. The model has the potential to assist in matching coverage and premiums to each driver’s specific needs and risks.

9.2.2. Risk Scoring and Premium Calculation
LLM has the ability to generate personalized risk scores for drivers based on their driving behavior analysis. These scores can be utilized to establish insurance premiums that are more precise, reflecting the driver’s real conduct on the road.

9.2.3. Claims Assessment
LLM is capable of providing assistance in evaluating the circumstances and variables which have resulted in an insurance claim. With the help of analyzing telematics data and other pertinent information, the system can assist in determining the responsibility, evaluating the extent of the incident, and simplifying the process of filing a claim.

9.2.4. Driving Insights and Feedback
Drivers can receive feedback and insights in real-time through LLM based on their behavior. Based on positive behavior changes, LLM can make personalized recommendations for improving driving skills, reducing risk, and even lowering insurance premiums.

9.2.5. Fraud Detection
The detection of fraudulent activities in UBI programs is something that LLM can assist with. Through analysis of the telematics data, the model is able to identify patterns, discrepancies, and anomalies that may point to instances of fraudulent behavior. For example, the tampering with or falsification of driving data could be detected using this approach.

9.3. Ride Sharing/E-hailing Industry
9.3.1. Driver Verification
LLM has the capability to provide more advanced verification mechanisms which would only allow verified and authorized drivers to participate in the ride-sharing platform. The platform can examine and authenticate the credentials, licenses, and background checks of drivers, thereby improving the verification process.

9.3.2. Biometric Authentication
It is possible to integrate biometric authentication techniques, such as voice recognition, into LLM for driver identification purposes. This would involve comparing the driver’s biometric information with previously registered profiles, thus adding an extra layer of security to prevent unauthorized access to the platform.

9.3.3. Anti-Fraud Measures
The ride-sharing platform can leverage LLM to detect and prevent fraudulent activities. LLM can analyze driver behavior or transaction data to identify patterns, discrepancies, and anomalies, which can signal potential cases of fraudulent behavior like identity theft or account manipulation.

9.3.4. Ride Verification
LLM has the ability to help verify that rides are completed and the correct driver is assigned to each trip. The model uses trip data, geolocation information, and driver identity to cross-check the information for preventing unauthorized ride pickups or fraudulent claims.

9.3.5. Driver Ratings and Feedback
Analyzing feedback from passengers can be a valuable way for LLM to contribute to the driver rating system. By analyzing text data, the model can extract sentiments, identify patterns, and provide insights that can help drivers improve their service quality.

9.4. Advanced Driver Assistance Systems
Drivers can receive real-time feedback on driving information, vehicle status, and other pertinent details through text and voice communication from their vehicles. This increased interaction between drivers and their vehicles enhances the overall driving experience. LLM can also contribute to this by analyzing driver behavior data, allowing drivers to gain insights into their driving behavior and receive personalized driving advice. As a result, driving safety is improved, and a more informed and responsible approach to driving is promoted.

9.4.1. Driver Monitoring
The utilization of LLM is possible for monitoring and identifying the driver’s state and attentiveness while driving. Through the analysis of voice patterns, this model can determine whether the driver is engaged, fatigued, or distracted, thus contributing to safer driving.

9.4.2. Personalized Settings
ADAS settings can be personalized by LLM based on driver identification. The system can automatically adjust seat position, mirrors, climate control, and other preferences to match the identified driver’s profile once the driver is recognized.

9.4.3. Driver-Specific Training and Guidance
LLM has the capability to offer customized training and instruction to drivers by analyzing their identification and driving habits. By examining driving data, the system can detect areas that require improvement and suggest real-time feedback or recommendations to improve driving abilities and ensure safety.

9.5. Fleet Management
9.5.1. Driver Authentication
LLM has the capability to provide advanced authentication mechanisms that guarantee that only authorized drivers can gain access to and operate fleet vehicles. The model can examine driver identification data, including biometric information or credentials, to authenticate the identity of drivers prior to permitting them access to vehicles.
9.5.2. Driving Behavior Analysis
Fleet vehicles’ driving behavior data can be analyzed by LLM to assess driver performance and safety. By analyzing telematics data that includes speed, acceleration, braking, and adherence to traffic rules, the model can evaluate individual driver behavior and give suggestions for improving performance or reducing risks.

9.5.3. Safety Monitoring
Real-time monitoring of driver safety within fleet vehicles can be enhanced by LLM. The model has the ability to analyze sensor data, driver behavior, and environmental factors to detect potential risks or hazards. In case of any issues, the model promptly sends alerts or interventions to prevent accidents and enhance overall safety.

9.5.4. Compliance Monitoring
Fleet management can benefit from LLM’s assistance in ensuring compliance with regulations and company policies. LLM’s analysis of driver behavior and operational data can detect instances of non-compliance, such as extended idling or speeding, and generate notifications or reports to ensure adherence to regulations and guidelines.

9.5.5. Incident Investigation
LLM can help investigate and analyze any accidents or incidents involving fleet vehicles. The model can analyze telematics data, driver behavior, and external factors to provide insights into the cause and contributing factors of the accident. These insights can be helpful in processing insurance claims and taking preventive measures to avoid future incidents.

9.6. Automotive Security

9.6.1. User Authentication
LLM has the capability to aid in the implementation of strong user authentication systems in automobiles. To make certain that only authorized individuals are able to access and operate the vehicle’s systems, it can utilize various biometric authentication methods such as voice recognition, facial recognition, and others.

9.6.2. Anomaly Detection
During the process of driver identification, LLM has the capability to recognize and alert about possible anomalies or suspicious activity. In order to do so, the model analyzes the present driver’s traits and actions, comparing them with previous data to identify any unusual patterns that may suggest security threats or unauthorized entry.

9.6.3. Integration with Security Systems
It is possible to combine LLM with various other automotive security systems, including intrusion detection systems, sensors, and surveillance cameras. The model’s language comprehension and analysis capabilities can be utilized to link driver identification data with other security events, resulting in a comprehensive security solution.

9.7. Strengths of LLM

9.7.1. Language Understanding
LLM has the ability to understand language at an advanced level, which enables it to analyze and comprehend driver identification information in different formats, including text, voice, and images. This feature can facilitate more precise and all-encompassing driver identification.

9.7.2. Multimodal Analysis
LLM has the ability to combine various data sources such as biometric information, driving behavior data, and identification documents. This integration enables the model to improve the precision and dependability of driver identification by comprehensively analyzing the different types of data available.

9.7.3. Continuous Learning
The LLM model can consistently learn and adjust to novel driver identification patterns and approaches. This feature may facilitate the model to enhance its performance with time and remain up-to-date with evolving identification techniques or security measures.

9.8. Weaknesses of LLM

9.8.1. Dependency on Data Quality
The effectiveness of LLM, similar to other AI models, is dependent on the quality and variety of the data used in its training. In case the training data for driver identification is inadequate, prejudiced, or restricted, it may result in incorrect outcomes or partiality in the identification procedure.

9.8.2. Vulnerability to Adversarial Attacks
LLM is vulnerable to adversarial attacks because of intentional input manipulation by malicious individuals who aim to trick the model and bypass driver identification systems. To prevent such attacks, it’s crucial to have strong security measures and adequate safeguards in place.

9.8.3. Ethical and Privacy Concerns
There are ethical and privacy concerns associated with using LLM or any AI model for driver identification. To ensure responsible and legal use of driver identification technologies, it is important to implement proper measures for consent, transparency, and data protection.

9.8.4. Hardware and Processing Requirements
Deploying LLM, which is a more advanced model, can be challenging in resource-constrained environments or devices that have limited capabilities due to the significant computational resources and processing power it requires.

10. Performance evaluation
It is essential to choose appropriate performance measures for classification systems based on their specific domain [121]. One common approach for evaluating classification performance is to use a confusion matrix, as shown in Table 9, which assesses a classification problem using test data. The confusion matrix is useful in determining the prediction accuracy for both negative and positive class instances. A True Positive (TP) is when both the actual and predicted classes are positive, indicating accurate prediction of positives. A True Negative (TN) is when both the predicted and actual classes are negative. False Positive (FP) cases are when the actual class is negative, but the expected class is positive, whereas False Negative (FN) cases are when the actual category is positive, but the projected category is negative. To optimize the performance of a driver identification system, it is crucial to minimize both FN and FP. Accuracy, recall, precision, and F-measure are some of the performance metrics frequently used in driver identification to assess classifier performance. Table 10 shows all the evaluation metrics used by the researchers. Descriptions of these performance indicators are provided below.
Table 10
Frequency of performance evaluation used in review studies.

<table>
<thead>
<tr>
<th>References</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1 Score</th>
<th>ERR</th>
<th>ROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>[67, 9, 80, 28, 23, 30, 79, 33, 76, 81, 57]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>[5, 63, 32, 15, 35, 29, 78]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>[18, 19, 1, 13]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[24]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>[45]</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[10]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>[43]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[74, 37]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[75, 73]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[64]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[77]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[44, 36]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[20]</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[1]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[16, 17]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[57]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[25]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>←</td>
<td>←</td>
</tr>
<tr>
<td>[77]</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>←</td>
<td>←</td>
</tr>
</tbody>
</table>

10.1. Accuracy

Driver identification is often evaluated using accuracy as a metric to determine correctness. This metric measures the proportion of correctly identified drivers to the total sample size. However, accuracy alone might not accurately identify imbalances in the dataset, especially if the classes are not evenly distributed.

\[
\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (12)
\]

10.2. Precision

Precision evaluates the model’s capability to accurately identify drivers among all the positive identifications made. It focuses on minimizing false positives.

\[
\text{Precision} = \frac{(TP)}{(TP + FP)} \quad (13)
\]

10.3. Recall

Recall is a measure used to evaluate the accuracy of a model in identifying positive cases (drivers) correctly. This measure aims to minimize the number of false negatives.

\[
\text{Recall} = \frac{(TP)}{(TP + FN)} \quad (14)
\]

10.4. F1 Score

The F1 score is a metric that considers both precision and recall, providing a comprehensive evaluation of a model’s performance in terms of false negatives and false positives.

\[
F1\text{Score} = \frac{2((\text{Precision}\text{Recall}))}{(\text{Precision} + \text{Recall})} \quad (15)
\]

10.5. Equal Error Rate (EER)

Biometric systems, such as driver identification, commonly use EER as a performance evaluation metric. EER represents the point of balance between false acceptance (FAR) and false rejection (FRR) rates in the system’s performance. For driver identification specifically, EER indicates the threshold where the system’s identification errors for different drivers are in equilibrium. To calculate FAR, FRR, and EER, we can use the following formulas:

\[
\text{FAR} = \frac{FP}{(FP + TN)} \quad (16)
\]

\[
\text{FRR} = \frac{FN}{(FN + TP)} \quad (17)
\]

\[
\text{EER} = \frac{((\text{FAR} + \text{FRR}))}{2} \quad (18)
\]

10.6. ROC Curve and AUC

The graphical representation of the performance of a model across different discrimination thresholds is provided by
the Receiver Operating Characteristic (ROC) curve. Meanwhile, the scalar value that quantifies this performance is known as the Area Under the Curve (AUC). The ROC curve and AUC provide a quantification of the balance between the sensitivity (true positive rate) and specificity (false positive rate) of the model.

11. Future research directions

Several research challenges were identified in the driver identification sector through a review study. To enhance the effectiveness of driver identification systems, substantial research efforts are required to address the highlighted research topic. Below, we delve deeper into the open research issues that need to be tackled.

1. Robustness and Generalization: Improving the resilience of driver identification models to cope with varying driving conditions, road types, and environmental elements. Advancing the overall adaptability of models to precisely recognize drivers in different situations and datasets.

2. Real-Time and Online Systems: Real-time and online driver identification systems are being developed to process data instantly and detect drivers and possible anomalies immediately. This necessitates the optimization of algorithms and architectures to achieve high processing speed with minimal latency.

3. Multimodal Data Fusion: Utilizing the fusion of multiple data modalities, including visual data, audio data, and physiological signals such as heart rate and eye movement, can improve the precision and dependability of driver identification. By merging data from different sensors and sources, it is possible to obtain a more thorough comprehension of driver behavior.

4. Privacy and Security: Ensuring the security of driver identification systems and addressing privacy concerns. Creating strategies to safeguard sensitive personal data and prohibit unauthorized access or misuse of driver information.

5. Unsupervised and Semi-supervised Learning: Exploring unsupervised and semi-supervised learning methods for identifying drivers, even when there is a lack of labeled data due to cost or limited availability. These techniques can help reduce the reliance on large, labeled datasets and make driver identification more feasible in practical situations.

6. Transfer Learning and Domain Adaptation: Investigating transfer learning and domain adaptation techniques to allow the transfer of knowledge from one driving situation to another. This could aid in the creation of driver identification models that are effective in a variety of driving scenarios, regardless of the type of vehicle, demographic, or environment.

7. Ethical and Legal Considerations: Developing ethical and legal strategies to handle driver identification issues, including data privacy, consent, and avoiding any potential biases. Establishing frameworks and guidelines to ensure using driver identification technologies in a fair and responsible manner.

8. Real-World Deployment and Validation: We are in the process of carrying out comprehensive field trials and validation studies to evaluate the effectiveness and practicality of driver identification systems. Our aim is to assess these systems’ real-world applicability and determine their scalability, reliability, and usability in practical settings. This evaluation will help us integrate these systems into current transportation systems and applications.

12. Conclusion:

In this paper, we have conducted an exhaustive examination of the multifaceted components integral to the architecture of driver identification systems. This exploration encompassed a diverse array of data sources pertinent to driver identification, including CAN-BUS/OBD-II interfaces, smartphones, Global Positioning Systems (GPS), Inertial Measurement Units (IMU), and wearable technologies, each accompanied by an analysis of their respective strengths and limitations. Critical to developing robust and efficient driver identification systems is the preprocessing of the data harvested from these sources. Multiple datasets used in the existing studies are examined based on multiple characteristics of the dataset, such as data source used, average trip, road type, and time of data collection. The strategic application of feature engineering methodologies is paramount in optimizing accuracy, expediting computational processes, and minimizing classification errors. Moreover, a classification algorithm is selected for driver identification from a wide variety of machine learning and deep learning algorithms. The process of driver identification requires a careful selection of algorithms from a vast range of machine learning and deep learning options. In this case, a classification algorithm has been chosen to ensure the most accurate and efficient results. Furthermore, our analysis illuminated the efficacy of hybrid models in the intelligent selection of algorithms tailored to specific objectives and data availabilities, thereby harnessing the synergistic advantages of multiple models to bolster generalization capabilities and fortify recognition resilience. In addition, this study has delved into the prospective applications of the advanced LLM framework in driver identification, highlighting its potential to revolutionize this field. Evaluation matrices are important for maintaining the generalization. Various matrices used in existing studies are identified. In conclusion, we have delineated prospective research trajectories aimed at continuously enhancing and refining driver identification systems, thereby contributing to the evolution of this pivotal technology in the context of automotive safety and personalization.

References


A Comprehensive Review: Analysis of Machine Learning, Deep Learning, and Large Language Model Techniques for Revolutionizing Driver Identification


URL <GotoISI>://WOS:000894963300007


URL <GotoISI>://WOS:000619167800007


[17] M. L. Bernardi, M. Cimitile, F. Martinelli, F. Mercaldo, Driver and path detection through time-series classification...
A Comprehensive Review: Analysis of Machine Learning, Deep Learning, and Large Language Model Techniques for Revolutionizing Driver Identification

S. Fang, H. Chen, Z. Khan, P. Fan, On the content delivery


