Journal of King Saud University – Computer and Information Sciences 34 (2022) 5921–5942

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Intelligent transportation systems: A survey on modern hardware devices for the era of machine learning

Issam Damaj [a,*], Salwa K. Al Khatib [a], Tarek Naous [b], Wafic Lawand [a], Zainab Z. Abdelrazzak [a,c], Hussein T. Mouftah [d]

[a] Electrical and Computer Engineering Department, Beirut Arab University, Debbieh, Lebanon
[b] Electrical and Computer Engineering Department, American University of Beirut, Beirut, Lebanon
[c] Mechanical Engineering Department, Beirut Arab University, Debbieh, Lebanon
[d] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada

ARTICLE INFO

Article history:
Received 13 April 2021
Revised 3 July 2021
Accepted 25 July 2021
Available online 30 July 2021

Keywords:
Intelligent transportation systems
Machine learning
Hardware devices
Performance evaluation
Taxonomy

ABSTRACT

The increasing complexity of Intelligent Transportation Systems (ITS), that comprise a wide variety of applications and services, has imposed a necessity for high-performance Modern Hardware Devices (MHDs). The performance challenge has become more noticeable with the integration of Machine Learning (ML) techniques deployed in large-scale settings. ML has effectively supported the field of ITS by providing efficient and optimized solutions to problems that were otherwise tackled using traditional statistical and analytical approaches. Addressing the hardware deployment needs of ITS in the era of ML is a challenging problem that involves temporal, spatial, environmental, and economical factors. This survey reviews the recent literature of ML-driven ITS, in which MHDs were utilized, with a focus on performance indicators. A taxonomy is then synthesized, giving a complete representation of what the current capabilities of the surveyed ITS rely on in terms of ML techniques and technological infrastructure. To alleviate the difficulties faced in the non-trivial task of selecting suitable ML techniques and MHDs for an ITS with a specific complexity level, a performance evaluation framework is proposed. The presented survey sets the basis for developing suitable hardware, facilitating the integration of ML within ITS, and bridging the gap between research and real-world deployments.

Contents

* Corresponding author at: Electrical and Computer Engineering Department, Faculty of Engineering, Debbieh Campus, Beirut Arab University, P.O. Box 11-50-20, Riad El Solh 11072809, Lebanon.
E-mail address: i.damaj@bau.edu.lb (I. Damaj).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2021.07.020
1319-1578/

1. Introduction

Over the past few decades, ITS have spiked an increasing research interest as a promising discipline for revolutionizing the transportation sector and solving common traffic- and vehicle-related problems. ITS comprise a multitude of interconnected engineering feats that function as an entity for optimizing network-scale travel experiences from a technical, social, economic, and environmental aspect. Such optimizations necessitate the advancement of information and communication technologies, electronic sensors, control systems, and computers, which highlights the data-driven nature of modern ITS (Zhang et al., 2011; An et al., 2011). By acquiring and analyzing relevant data, ITS can efficiently manage computing hardware resources through several control and coordination algorithms, resulting in better traveler convenience, reduced fuel consumption, and enhanced traffic flow (Meneguette et al., 2018). ITS is a broad topic that encompasses many subjects, where each one of these subtopics is a research challenge of its own. A plethora of studies have addressed these subjects, but they can be classified into a few areas including Advanced Traffic Management Systems, Advanced Traveler Information Systems, Advanced Vehicle Control Systems, Advanced Public Transport Systems, Commercial Vehicle Operations, and Rural Transportation Systems (Sussman, 2005).

The complexity of ITS applications necessitates computationally powerful algorithms capable of analyzing tremendous amounts of data, especially in the era of Big Data, as well as competent implementation devices to obtain accurate results in real time. Conveniently, ML comprises a wide range of such algorithms designed for classification, regression, ranking, clustering, and dimensionality reduction in different learning scenarios such as supervised, unsupervised, semi-supervised, online, reinforcement, or active learning (Mohri et al., 2018) as well as federated learning. As such, ML, including Deep Learning (DL), proves crucial for diverse applications including, but not limited to, text or document classification, natural language processing, speech processing, computer vision, among others (Mohri et al., 2018). From there stems its importance for ITS applications in handling immense data concerned with different features of vehicles, roads, traffic, passengers, and many others to perform detection, recognition, and prediction of the different parameters involved for optimal assessment and management of transportation systems. These may include vehicle type recognition, lane detection, vehicle tracking, traffic speed estimation, and more.

Consequently, hardware devices capable of satisfying the different computational requirements of ML algorithms are needed in Onboard Units (OBUs) and Roadside Units (RSUs). An ensemble of MHDs such as Graphical Processing Units (GPUs), Central Processing Units (CPUs), Microcontroller Units (MCUs), Field Programmable Gate Arrays (FPGAs), Application-Specific Integrated Circuits (ASICs), and others offer promising potential for ITS applications owing to their accessibility, reliability, cost-effectiveness, and performance specifications. Depending on the computational complexity of a required ML technique for a specific ITS application, MHDs are chosen based on specific performance indicators including computing performance, flexibility, power consumption, and computational resources (Velez et al., 2015).

ITS has emerged as a topic of vigorous research loaded with a multitude of challenges, which motivates conducting a thorough literature survey on MHDs utilized in ML-Driven ITS applications. Some of these applications, along with commonly utilized MHDs and ML techniques, can be seen in Fig. 1. This field comprises a complex network of data sources, hardware units, sensors, infrastructure, communication technologies, and intelligent control (Meneguette et al., 2018). Applications such as vehicle routing, intersection management, coordinated parking, and many others, necessitate the utilization of a system capable of real-time computation and optimization, tasks conveniently performed by ML models. Such tasks elicit the need for advanced ML techniques including online learning (Ser et al., 2019), or more recently federated learning (MManias and Shamianias and Shami, 2021), which are adopted in ITS systems owing to their ability to perform real-time computations on tremendous amounts of data. These demanding computations vitally require advanced collective resources for training and testing of models and pose the challenge of selecting suitable hardware units. To that end, hardware resources have to efficiently take in and process huge amounts of data, satisfying the needs of large-scale learning models. For such large settings, such as in federated learning, the device is not only used to serve the ML model at inference but would also be used for re-training the model on updated data that is gathered collectively by ITS units. Thus, it is important to carefully consider the type of hardware device used in deployment, especially with the recent advances of large-scale applications such as smart cities and connected and autonomous vehicles.

When it comes to a multitude of hardware units in transportation systems, numerous challenging temporal, spatial, environmental, economical, and performance factors are involved. Thus, hardware units must be carefully selected to satisfy the needs of learning models while alleviating the impact of these challenges. A thorough review of existing surveys shows the lack of a general review on MHDs used to run ML techniques in ITS applications. Since common embedded hardware units in connected vehicles, trains, or any mode of transportation usually have limited capabilities compared to other computationally fit processors, there exists a need to explore other processor technologies. Such units must be capable of taking in and analyzing huge amounts of data while attaining appealing performance characteristics. Examples of such units are Automotive Grade (AG) (Yalla et al., 2015; Sistu et al., 2019) and off-the-shelf hardware, on which learning models are easily deployed, facilitating the process of testing different scenarios in ITS. To that end, our research objectives are as follows:

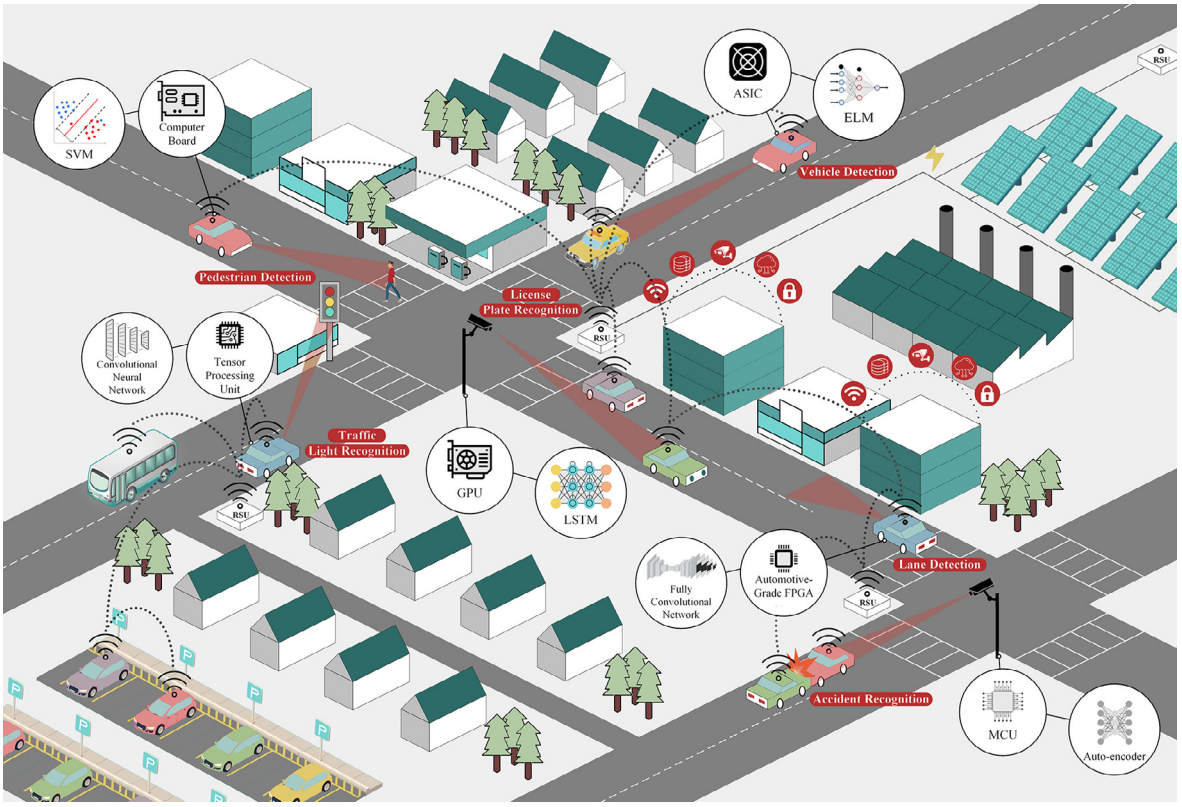

Fig. 1. An ensemble of ITS applications in vehicle control and traffic management with emphasis on the diverse adoption of MHDs and ML techniques.

- Present the different applications in the literature within ITS where ML can be of benefit and introduce what learning models have been used to tackle each application.
- Identify hardware devices that are used to satisfy the needs of ML-driven ITS applications. Specifically, our survey focuses on the processing technology rather than sensors or communication interfaces.
- Identify the choice factors of ML techniques and MHDs.
- Synthesize different basic taxonomies confined within the specified article context. Our proposed taxonomy considers only the ITS applications within the areas of Advanced Vehicle Control and Traffic Management Systems, the ML techniques used to address various problems within these two areas, the processing devices relied on for testing, validation, or real deployment, as well as performance characteristics that give complexity or competency indications.
- Develop a performance evaluation framework for ITS applications, ML techniques, and hardware device characteristics that enable cross-matching among the three areas reflecting how they interweave.
- Validate the developed framework through its application on a diverse set of carefully selected investigations.
- Analyze common and good practices within the paper context and identify promising patterns of use versus achievements.
- Determine important limitations and gaps and refine a set of pointers to future directions within the specified paper context.

A systematic protocol was abided by during the search of the articles to be surveyed, and it is as follows. The academic search engines used were *IEEE Xplore digital library*, *ScienceDirect*, and *SpringerLink*. Several combinations of the following keywords were used in the search process: *intelligent transportation systems*, *traffic management*, *vehicle control*, *artificial intelligence*, *machine learning*, *deep learning*, *computer vision*, *hardware devices*, *onboard units*, and *roadside units*. Finally, to keep the scope of this survey modern, only publications as of 2014 were regarded.

In this paper, The adopted methodology is as follows:

- Develop the search protocol
- Survey the literature
- Develop classifications of surveyed investigations
- Develop the taxonomies
- Develop and validate the evaluation framework
- Analyze the survey findings and provide recommendations
- Conclude with open research questions

Although there are a plethora of surveys on various areas within ITS, no survey has yet addressed the suitability of potential MHDs that can be deployed for ML-driven ITS applications. Several surveys in the literature present thorough investigations on ITS applications where data-driven solutions have a great impact, thus enabling the usage of machine and deep learning techniques (Veres and Moussa, 2019; Zhang et al., 2011; Zhu et al., 2018). Among such surveys are some that fully focus on specific areas in advanced Vehicle Control (Arnold et al., 2019; Kuutti et al., 2021; Martinez et al., 2017a,b) or advanced Traffic Management (Chen et al., 2015; Datondji et al., 2016; Jensen et al., 2016). Hardware devices used for visual perception systems in autonomous vehicles were presented in Shi et al. (2017). Additionally, the different sensor technologies that were considered for ITS were surveyed in Guerrero-Ibáñez et al. (2018). However, none of these surveys consider the MHDs needed to satisfy the potentially complex requirements of the application considered and ML technique used. In the literature, there are available surveys that tackle the development of hardware devices for efficient operation of ML models (Sze et al., 2017; Talib et al., XXXX), but these

investigations remain general and do not consider the requirements of ITS applications and their implications. Therefore, to the best of our knowledge, our work is the first in the literature that explores MHDs needed to comply with ITS and ML requirements for high-performance operation.

This paper is structured so that Section 2 thoroughly surveys the literature on MHDs for ML-driven ITS applications. In Section 3, the synthesized taxonomies that pertain to ITS, ML, and MHDs are presented along with their developed performance models. In Section 4 the cross-matching evaluation chart and selection model of the proposed framework are presented. Section 5 provides a deep discussion that comprises validating the developed model, analyzing common and good practices, identifying gaps and limitations, and refining a rich set of future work recommendations. Concluding remarks, open research questions, and additional future directions follow in Section 6.

2. MHDs for ML-Driven ITS Applications

A thorough review of the recent literature brings to attention the extensive research focus on implementing ML techniques on MHDs to solve common ITS problems concerned with Vehicle Control and Traffic Management. These problems involve acquiring data for further processing and analysis while achieving real-time requirements. Thus, training and testing of suitable ML algorithms are performed and compared with similar work through performance evaluation. Authors have selected different hardware devices that fall under three main categories: Multipurpose Devices (MDs), Specialized Devices (SDs), and Hybrid Devices (HDs). MDs are defined as hardware devices capable of performing diverse computational tasks due to their reprogrammability. In contrast, SDs are those specifically designed and optimized for the desired application. HDs, however, are those combining the two aforementioned types to achieve unique results by bringing Hardware/Software co-design, pre-processing, and co-processing features to the table. The literature shows that implemented models perform better on certain MHDs than on others when it comes to the time and accuracy requirements of an ITS application. This section surveys the most recent work in this context and touches on their performance, first from an ITS perspective, then according to the three aforementioned hardware categories.

Table 1 shows a classification of the surveyed articles by hardware category in chronological order, where the used hardware device, ML algorithm, and ITS application were presented. The hardware devices identified are the primary hardware devices used in the investigations as several investigations utilized secondary hardware devices for purposes such as training and/or pre-processing. The different factors influencing the choice of the hardware device and ML algorithm per article are also highlighted for each reference. These choice factors were identified in the papers by either inferring from the text or finding these factors explicitly mentioned. For ML algorithms, the choice factors are as follows: performance reflects training and inference time, accuracy reflects the ability of the model to produce correct results, robustness reflects the ability of the algorithm to handle changes to the input data, and complexity reflects how computationally light the model is. For MHDs, the choice factors are defined as follows: performance reflects the processing power of the device, flexibility reflects the extent of reconfigurability of each device, power consumption reflects that the author aimed to develop an energy-efficient implementation, and resources reflect how many computational resources the device can offer. In Table 1, L-CNN stands for Lookup-based Convolutional Neural Network, SE-ResNet for Squeeze-and-Excitation-Residual Network, and GCGA for Graph Convolutional Generative Autoencoder. The superscript in the Hardware Device column corresponds to the ML purpose the device was used for, where 'train' stands for Training, 'inf' for Inference, 'on' for Online Learning, 'pre' for Pre-processing, and 'data' for Data Acquisition. Similarly, the superscript in the ITS application column corresponds to the scope of the ITS application, where 'sim' stands for Simulation, 'pro' for Physical ITS Prototype, and 'dep' for Physical ITS Real Deployment.

The surveyed literature depicts not only ML techniques used for theoretical testing and training purposes (Simulations), but also those with the potential to be implemented in real ITS scenarios (Physical ITS Prototypes and Physical ITS Real Deployment), as shown in Table 1. While a major portion of the surveyed articles were merely Simulations, where the proposed approaches were simulated on CPUs or GPUs, namely (Dairi et al., 2018; Hassannejad et al., 2015; Königshof et al., 2019; Wont et al., 2018; Chen et al., 2017; Kosaka and Ohashi, 2015; Kuang et al., 2016; Nguyen et al., 2016; Chen and Huang, 2016; Gudigar et al., 2019; Chen et al., 2014; Tian et al., 2019; Li et al., 2018; Wang and Zhou, 2018; Sanz-Madoz et al., 2019; Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Jin et al., 2014; Lu et al., 2016; Yuan et al., 2016; Shi et al., 2015; Deng et al., 2020; Outay et al., 2019; Li et al., 2019; Wang et al., 2017; de Paula and Jung, 2015; Abughalieh and Alawneh, 2020; Gkolias and Vlahogianni, 2019; Chen et al., 2017; Soon et al., 2018; Shvai et al., 2020; Wang et al., 2017; Xie et al., 2018; Hu et al., 2015; Fang et al., 2016; Bulan et al., 2017; Yu and Gu, 2019; Ke et al., 2018; Chung and Sohn, 2017; Cecilia et al., 2018; Shi et al., 2016; Vu et al., 2017; Ke et al., 2018; Singh and Mohan, 2018; Lee and Min, 2018; Du et al., 2020; Zang et al., 2018; Zheng et al., 2019; Zhan et al., 2018), others implemented or even deployed the proposed ML techniques. On the other hand, authors in (Chen et al., 2013; Hsiao et al., 2014; Khan et al., 2016; Sangeetha and Deepa, 2017; Hsiao et al., 2016; Cabanes et al., 2019; Lee et al., 2016; Giesemann et al., 2014; Weng and Chiu, 2018; Lyu et al., 2018; Jing et al., 2017; Mhalla et al., 2018; Khan et al., 2016; Dey et al., 2018) developed Physical ITS Prototypes. That is to say that they developed hardware implementations on devices such as FPGAs, MCUs, and computer boards but only tested their approaches using already existing datasets. However, in (Hsiao et al., 2015; Chen et al., 2019; Heredia and Barros-Gavilanes, 2019; Han et al., 2017; Wu et al., 2019; Alonso et al., 2015; Vasconcelos et al., 2017; Panahi and Gholampour, 2016; Khan et al., 2016; Wang et al., 2019; Campoverde and Barros, 2019; Hu et al., 2014; Magrini et al., 2015; Hu et al., 2014; Vinh, 2015; Zhou et al., 2016; Asmara et al., 2020), hardware implementations were developed in addition to testing using real data for the target ITS applications. Thus, this kind of work represents actual Physical ITS Real Deployment of the proposed techniques as it runs in real scenarios that involve data collection using cameras and sensors as input to the ML techniques deployed on MHDs.

2.1. ITS Applications

As shown in Table 1, there exists a considerable body of the literature tackling common ITS problems, one of which is Object Detection. In Hsiao et al. (2014), edge feature extraction was combined with ML to detect vehicles, while DL was deployed in Chen et al. (2019) for panoramic video detection. Nighttime vehicle detection was the focus of Kosaka and Ohashi (2015), in which ML was utilized for blob detection of vehicle lights, and of Kuang et al. (2016), in which bio-inspired image enhancement was combined with DL and ML for detection in challenging scenarios. DL was presented in Nguyen et al. (2016) with local patterns and depth information to detect, recognize, and track vehicles and pedestrians. Pedestrian detection was also addressed in (Chen

Table 1
MHDs, ML techniques, and ITS applications used in the surveyed papers sorted by hardware class with emphasis on choice factors and chronological deployment trends.

Class	Ref.	Year	Hardware Device[Purpose of Use]	Hardware Device Choice Factors				ML Algorithm	ML Algorithm Choice Factors				ITS Application[Scope]
				Performance	Flexibility	Power Con.	Resources		Performance	Accuracy	Robustness	Complexity	
Multipurpose Devices	Hu et al. (2014)		ARM Cortex A9[pre]	✓				Gaussian Mixture Model			✓	✓	Object Detection[dep]
	Hu et al. (2014)		ARM Cortex A9[train, inf]	✓				Adaboost		✓	✓		Traffic Monitoring[dep]
	Hsiao et al. (2014)		Xilinx Virtex-5[pre]	✓			✓	Support Vector Machine (SVM)		✓			Object Detection[dep]
	Chen et al. (2013)	2014	Intel Altera Stratix II[inf]	✓				SVM				✓	Object Detection[pro]
	Alonso et al. (2015)		ARM Cortex M4F[data]			✓		SVM		✓			Vehicle Analysis[dep]
	Hsiao et al. (2015)		Xilinx Spartan-6[train, inf, pre]	✓				SVM		✓			Object Detection [dep]
	Magrini et al. (2015)		Custom-made ARM-based MCU[train, inf]		✓	✓		Gaussian Mixture Model			✓		Traffic Monitoring[dep]
	Khan et al. (2016)		Intel Altera Cyclone IV[inf]			✓		SVM				✓	Object Detection[pro]
	Hsiao et al. (2016)		Xilinx Spartan-6[pre]	✓		✓		AdaBoost	✓				Object Detection[pro]
	Lee et al. (2016)	2016	Xilinx Virtex-7[pre]	✓				SVM				✓	Instruction Recognition[pro]
	Vasconcelos et al. (2017)		ARM Cortex A9[train, inf, on			✓		K-means	✓	✓			Vehicle Analysis[dep]
	Sangeetha and Deepa (2017)		Xilinx Virtex-5[pre]	✓	✓			SVM		✓			Object Detection[pro]
	Jing et al. (2017)	2017	Intel Altera Cyclone IV[inf]	✓			✓	Artificial Neural Network (ANN)		✓			Traffic Detection[pro]
	Giesemann et al. (2014)		Xilinx Virtex-6[inf]	✓	✓			SVM			✓		Instruction Recognition[pro]
	Lyu et al. (2018)		Xilinx UltraScale[inf]	✓				Convolutional Neural Network (CNN)		✓			Road Detection[pro]
	Wu et al. (2019)		Xilinx Artix-7[inf]		✓	✓		AdaBoost	✓			✓	Instruction Recognition[dep]
	Heredia and Barros-Gavilanes (2019)		Raspberry Pi 4[inf]	✓				SVM & CNN		✓		✓	Object Detection[dep]
	Chen et al. (2019)	2019	Xilinx Spartan-3[inf]	✓		✓		EZ-Net (CNN)	✓	✓			Object Detection[dep]
	Asmara et al. (2020)		Raspberry Pi 3B+[inf]			✓		CNN	✓	✓			Traffic Prediction[dep]
	Jin et al. (2014)		NVIDIA Tesla C2075[train, inf]	✓			✓	CNN		✓			Instruction Recognition[sim]
	Chen et al. (2014)	2014	NVIDIA Tesla K20[train, inf]	✓			✓	HOG-SVM		✓			Instruction Recognition[sim]
	Nguyen et al. (2016)		NVIDIA Tesla K40[train, inf]	✓				TCNN & R-TCNN		✓	✓		Object detection[sim]
	Chen et al. (2017)		NVIDIA GeForce GTX 960[train, inf]	✓				CNN		✓	✓		Instruction Recognition[sim]
	Chen et al. (2017)		NVIDIA GeForce GTX 960[train, inf]	✓				K-means				✓	Object Detection[sim]
	Bulan et al. (2017)		NVIDIA GeForce GTX 570[train, inf]	✓			✓	CNN	✓	✓	✓		Traffic Detection[sim]
	Kuang et al. (2016)		NVIDIA GeForce GTX 970[train, inf]	✓				CNN		✓			Object Detection[sim]
	Lu et al. (2016)		NVIDIA Tesla C2075[train, inf]				✓	M2-tMTL (ANN)		✓		✓	Instruction Recognition[sim]
	Yuan et al. (2016)		NVIDIA GeForce GTX Titan X[train, inf]	✓			✓	SVM			✓	✓	Instruction Recognition[sim]
	Fang et al. (2016)	2017	NVIDIA GeForce GTX Titan X[train, inf]	✓				CNN		✓			Traffic Detection[sim]
	Lee and Min (2018)		NVIDIA GeForce GTX 1080 Ti[train, inf]	✓				Long Short-term Memory			✓		Traffic Prediction[sim]
	Cecilia et al. (2018)		NVIDIA Tesla Kepler K40C[train, inf]	✓			✓	Fuzzy Minimals				✓	Optimization[sim]
	Wang et al. (2017)		NVIDIA Tesla K80[train, inf]				✓	Fully-Convolutional Siamese Networks			✓		Road Detection[sim]
	Xie et al. (2018)		NVIDIA GeForce GTX 980[train, inf]	✓				CNN		✓		✓	Traffic Detection[sim]
	Zhu et al. (2017)		NVIDIA GeForce GTX Titan X[train, inf]	✓				Fully Convolutional Networks			✓		Instruction Recognition[sim]

(continued on next page)

Table 1 (*continued*)

Class	Ref.	Year	Hardware Device[Purpose of Use]	Hardware Device Choice Factors: Performance	Flexibility	Power Con.	Resources	ML Algorithm	ML Algorithm Choice Factors: Performance	Accuracy	Robustness	Complexity	ITS Application[Scope]
	Wont et al. (2018)		NVIDIA GeForce GTX Titan X[train, inf]	✓				AggNet (CNN)			✓		Object Detection[sim]
	Weng and Chiu (2018)		TSMC 90 nm[pre]	✓		✓	✓	SVM		✓		✓	Instruction Recognition[pro]
	Vu et al. (2017)		NVIDIA GeForce GTX 1050 Ti[train, inf]				✓	CNN	✓	✓			Incident Recognition[sim]
	Chung and Sohn (2017)		NVIDIA Quadro K5200[train, inf]				✓	CNN		✓			Traffic Monitoring[sim]
	Lee and Kim (2018)		Snapdragon TM 820A (GPU)[train, inf]	✓				CNN		✓			Instruction Recognition[sim]
	Luo et al. (2017)		NVIDIA Titan Black[train, inf]	✓				CNN	✓	✓			Instruction Recognition[sim]
	Soon et al. (2018)		NVIDIA GeForce GT 730[train, inf]	✓				CNN		✓	✓	✓	Traffic Detection[sim]
	Wang et al. (2017)	2018	NVIDIA GeForce GTX 1080[train, inf]	✓			✓	CNN		✓			Traffic Detection[sim]
	Singh and Mohan (2018)		NVIDIA Tesla K20c[train, inf]	✓			✓	Stacked Autoencoder		✓			Incident Recognition[sim]
	Tian et al. (2019)		NVIDIA GeForce GTX Titan X [train, inf]	✓				MSRA-NN (RNN)	✓		✓		Instruction Recognition[sim]
	Königshof et al. (2019)		NVIDIA GeForce GTX Titan X[train, inf]	✓				CNN	✓				Object Detection[sim]
	Li et al. (2019)		NVIDIA GeForce GTX Titan X[train, inf]	✓			✓	L-CNN		✓			Road Detection[sim]
	Yu and Gu (2019)		NVIDIA GeForce GTX 1080 Ti[train, inf]	✓				GCGA	✓				Traffic Monitoring[sim]
	Li et al. (2018)		NVIDIA Titan Xp[train, inf]				✓	SE-ResNet	✓				Instruction Recognition[sim]
	Wang and Zhou (2018)		NVIDIA GeForce GTX 760[train, inf]	✓				CNN	✓				Instruction Recognition[sim]
	Sanz-Madoz et al. (2019)		NVIDIA Geforce GTX 980[train, inf]				✓	Extreme Learning Machine	✓			✓	Instruction Recognition [sim]
	Ke et al. (2018)		NVIDIA GeForce GTX 1080[train, inf]	✓				CNN		✓			Traffic Monitoring[sim]
	Zang et al. (2018)		NVIDIA GeForce GTX 1080[train, inf]	✓			✓	CNN & Long Short-term Memory	✓		✓		Traffic Prediction[sim]
	Zheng et al. (2019)		NVIDIA GeForce GTX 1080 Ti[train, inf]	✓				CNN & Long Short-term Memory	✓	✓			Traffic Prediction[sim]
	Shvai et al. (2020)		NVIDIA Tesla K80[train, inf]				✓	CNN	✓				Traffic Detection[sim]
	Du et al. (2020)		NVIDIA Tesla P100-PICE[train, inf]				✓	Residual Long Short-term Memory	✓	✓	✓		Traffic Prediction[sim]
	Deng et al. (2020)		NVIDIA Titan Xp[train, inf]	✓				CNN	✓				Vehicle Analysis[sim]
	Campoverde and Barros (2019)		NVIDIA GeForce GTX 1060[train, inf]				✓	Deep Neural Network (SSD)	✓	✓			Traffic Detection[dep]
Specialized Devices	Abughalieh and Alawneh (2020)	2020	NVIDIA GeForce GTX 1060[train, inf]				✓	CNN		✓			Driver Assistance[sim]
	Vinh (2015)		FriendlyARM Tiny4412 Board[inf]	✓		✓		SVM	✓	✓			Instruction Recognition[dep]
	Khan et al. (2016)		Xilinx Zynq-7000 SoC[train, inf, pre]	✓		✓		SVM		✓	✓		Traffic Monitoring[dep]
	Zhou et al. (2016)		Xilinx Zynq-7000 SoC[inf, pre]	✓			✓	SVM		✓	✓		Instruction Recognition[dep]
	Han et al. (2017)		NVIDIA Jetson TX1[inf]	✓			✓	CNN		✓			Instruction Recognition[dep]
	Dey et al. (2018)	2018	Xilinx Zynq-7000 SoC[train, inf, on	✓				CNN		✓			Traffic Monitoring[pro]
	Wang et al. (2019)		NVIDIA Jetson TK1[train, inf]	✓			✓	CNN		✓			Traffic Detection[dep]
	Cabanes et al. (2019)		Xilinx Zynq-7000 SoC[train, inf, pre]		✓			SVM				✓	Object Detection[pro]
Hybrid Devices	Mhalla et al. (2018)	2019	NVIDIA Tegra TX1[train, inf]				✓	Region-based CNN			✓		Traffic Detection[pro]

et al., 2013; Khan et al., 2016; Sangeetha and Deepa, 2017; Hsiao et al., 2016; Hsiao et al., 2015; Heredia and Barros-Gavilanes, 2019). Both Hsiao et al. (2016) and Hsiao et al. (2015) employed background image segmentation with ML, while (Chen et al., 2013) used ML with feature extraction. ML was also utilized in Khan et al. (2016) for a highly accurate, memory-efficient accelerator, and in Sangeetha and Deepa (2017) for multi-scale, low-cost, hardware-efficient detection. Multiple classification algorithms were implemented in Heredia and Barros-Gavilanes (2019) to detect pedestrians, bicycles, and autos. Multiple object detection was also the focus of Dairi et al. (2018) using DL-based detection and Cabanes et al. (2019) by automating prototyping steps to reduce execution time. Furthermore, a real-time stereo-based 3-dimensional DL approach was introduced in (Königshof et al., 2019). However, for small objects, an aggregated DL-based approach was proposed in (Wont et al., 2018). Finally, to detect moving objects, real-time detection was achieved using motion extraction and visual features with ensemble learning in Hassannejad et al. (2015), and detection and motion estimation were realized using stereo super-pixel boundary classification in (Chen et al., 2017).

A commonly addressed ITS problem is the recognition of traffic signs and traffic lights. Traffic sign recognition was investigated in (Lee et al., 2016; Gudigar et al., 2019; Han et al., 2017; Weng and Chiu, 2018; Chen et al., 2014; Sanz-Madoz et al., 2019; Luo et al., 2017; Jin et al., 2014; Lu et al., 2016; Yuan et al., 2016; Vinh, 2015; Zhou et al., 2016). ML was implemented with feature extraction in (Lee et al., 2016; Gudigar et al., 2019; Chen et al., 2014) and with computer vision in (Weng and Chiu, 2018). Additionally, decreased computational and memory cost performance was achieved in Lu et al. (2016) using multitask learning. DL was utilized with color and shape segmentation in Han et al. (2017) and feature extraction in (Sanz-Madoz et al., 2019). Moreover, using DL, symbol and text-based classification was realized in Luo et al. (2017) and state-of-the-art recognition was achieved in (Jin et al., 2014). Notably, the ML approach in Yuan et al. (2016) achieved performance comparable with DL with less computational power. Traffic sign detection was discussed in (Giesemann et al., 2014; Tian et al., 2019; Li et al., 2018; Lee and Kim, 2018; Zhu et al., 2017). In Giesemann et al. (2014), an ML approach with image segmentation was presented, while DL was employed in Tian et al. (2019) for high-accuracy multi-scale recurrent attention analysis, Li et al. (2018) for end-to-end saliency model capable of dealing with challenging visual conditions, Lee and Kim (2018) for robust low-power detection, and Zhu et al. (2017) for an efficient framework able to detect different languages. Furthermore, Vinh (2015) and Zhou et al. (2016) both deployed computer vision and ML techniques to detect and recognize traffic signs, respectively. Finally, (Chen and Huang, 2016; Wu et al., 2019; Wang and Zhou, 2018; Shi et al., 2015) studied traffic light detection, of which Chen and Huang (2016) and Shi et al. (2015) deployed a vision-based ML approach, while Wu et al. (2019) used ensemble learning for efficient performance, and Wang and Zhou (2018) proposed DL with high dynamic range imaging.

Several other vehicle-related challenges have also attracted attention in recent years. For instance, Alonso et al. (2015) utilized computer vision and ML to detect weather status using road surface conditions. Other work investigated driver behavior prediction by applying ML techniques to speed and acceleration data Vasconcelos et al. (2017) and driver attention assessment by tracking eye movement using DL (Deng et al., 2020). Several other studies focused on road-related problems. For example, Lyu et al. (2018) implemented a DL-based road segmentation algorithm to process LiDAR data, Li et al. (2019) deployed an end-to-end DL system to perform road traffic line detection, Wang et al. (2017) implemented a DL model to detect roads, while de Paula and Jung (2015) applied computer vision and ML to detect and classify road lane markings. Additionally, to enhance visibility in foggy scenarios, Outay et al. (2019) utilized feature extraction with ML for excellent visibility range estimation. Other works presenting less-addressed driving issues include detection of pedestrians' intention to cross using DL with depth-sensing in Abughalieh and Alawneh (2020), detection of on-street parking spaces using DL with in-vehicle camera images in Gkolias and Vlahogianni (2019), and robust nighttime turn signal detection and tracking using DL in Chen et al. (2017).

Multiple investigations in the literature have focused on traffic surveillance. Namely, traffic detection and classification were tackled in (Khan et al., 2016; Wang et al., 2019; Campoverde and Barros, 2019; Shvai et al., 2020; Mhalla et al., 2018; Wang et al., 2017; Hu et al., 2015; Fang et al., 2016). ML was utilized to recognize vehicle type and color in (Khan et al., 2016) and car brand using multiple-instance learning in (Hu et al., 2015). Furthermore, DL was implemented in several investigations for vehicle type recognition. Namely, it was used for classification in Wang et al. (2019), challenging scenario detection in Mhalla et al. (2018), and recognition by transfer learning in (Wang et al., 2017). Moreover, DL models were deployed on recycled smartphones in Campoverde and Barros (2019) for the detection of urban actors and automatic toll collection in Shvai et al. (2020). Finally, DL and ML were employed for vehicle model recognition in Fang et al. (2016). Other issues included the recognition of vehicle logos and license plates. For instance, vehicle logo recognition was achieved in Soon et al. (2018) by DL hyper-parameter optimization. License plate recognition was addressed by (Panahi and Gholampour, 2016; Jing et al., 2017; Bulan et al., 2017), where ML was used in Panahi and Gholampour (2016) for recognition of unclear plates, DL based optical character recognition was implemented in Jing et al. (2017), and DL based segmentation- and annotation- free recognition was attained in Bulan et al. (2017) for challenging scenarios. Lastly, Xie et al. (2018) deployed a DL-based approach for multi-direction license plate detection with limited computational resources.

Researchers in (Hu et al., 2014; Magrini et al., 2015; Khan et al., 2016; Dey et al., 2018; Yu and Gu, 2019; Ke et al., 2018; Chung and Sohn, 2017) focused on monitoring traffic and passenger flow parameters in real time. For instance, Hu et al. (2014) used foreground detection and ML to count the number of passengers. In Magrini et al. (2015) and Ke et al. (2018), traffic flow estimation was investigated, where Magrini et al. (2015) utilized monitoring cameras and ML with constrained memory and computational power, while Ke et al. (2018) used ensemble learning with unmanned aerial vehicle data. ML was also employed in Khan et al. (2016) for queue length, speed, and count estimation, and in Chung and Sohn (2017) for traffic density measurement. Additionally, traffic speed maps were generated in Yu and Gu (2019) using DL architecture. Finally, traffic load characterization was done in Dey et al. (2018) using a DL model that proved effective even with little training data.

Traffic parameter prediction is another application occasionally encountered in the literature. Lee and Min (2018) deployed a long short-term memory DL model to accurately predict urban traffic. Du et al. (2020) utilized a spatio-temporal DL model for urban traffic passenger flow prediction, which outperformed traditional and DL-based methods. In addition, Zang et al. (2018) applied a multi-scale spatio-temporal DL approach for long-term traffic speed prediction. Traffic flow prediction was also tackled in Zheng et al. (2019) and Zhan et al. (2018), where a hybrid deep and embedding learning approach was presented in Zheng et al. (2019), and ensemble learning was used in Zhan et al. (2018). Finally, Asmara et al. (2020) addressed traffic density prediction using the You Only See Once deep learning technique for object detection.

Other ITS problems studied include monitoring and identifying miscellaneous traffic variables. For example, Vu et al. (2017) used DL with highway live streaming videos to identify road violations. To detect abandoned items robustly and accurately, ML was employed with foreground information extraction in Hu et al. (2014). Moreover, to detect road congestion, surveillance videos were used with multidimensional feature space DL in Ke et al. (2018). Accident recognition and localization were achieved through automatic deep feature representation of spatio-temporal video volumes in Singh and Mohan (2018) even in harsh visual conditions. Another challenge was air pollution monitoring, which was tackled using vehicle position and status data with ML in Cecilia et al. (2018) even with high loads. Finally, automatic road crack detection was addressed in Shi et al. (2016) by deploying ML using learned visual crack features in a fast, precise, and easy to parallel approach.

2.2. Multipurpose Devices

An abundance of work in the literature has used CPUs to run ML algorithms in the context of various ITS application, as was highlighted in the aforementioned discussion. Among these are Intel processors such as the Intel Xeon E5 used in Kosaka and Ohashi (2015) and Outay et al. (2019), the Intel i5 used in (Gkolias and Vlahogianni, 2019; Zhan et al., 2018; Panahi and Gholampour, 2016; Gudigar et al., 2019), and the Intel i7 used in (de Paula and Jung, 2015; Hu et al., 2015; Dairi et al., 2018). However, those processors that fall under the umbrella of Complex Instruction Set Computers (CISC) processors are known for their bulk instructions that mandate high latency per cycle. Thus, such processors are not necessarily compact enough to cater to the needs of OBUs and RSUs when real-time ITS applications are considered. Hence, they are just used for simulation and testing.

Other works in the literature have adopted stand-alone Reduced-instruction-set Computing (RISC) processors to run various ML algorithms in different ITS scenarios. Among these are Advanced RISC Machines (ARM) processors. For instance, the ARM Cortex A9 Processor was used in Hu et al. (2014) to run the Gaussian Mixture Model algorithm for real-time detection of abandoned objects, and in Hu et al. (2014) where the AdaBoost classifier was trained to detect passengers in public areas. The ARM Cortex A9 was also used in Vasconcelos et al. (2017), where the Samsung Galaxy SIII smartphone was used to run the K-means algorithm for driver behavior prediction, achieving a high accuracy of 95.45%. Another ARM processor, the ARM Cortex-M4F, was used in Alonso et al. (2015) to run a SVM classifier for detecting road weather status in real time, while keeping power consumption low with a CPU usage of only 5.3%.

A different line of research uses FPGAs as the hardware device instead of CPUs. Most notably, FPGAs developed by Xilinx have been widely used in several investigations for ML applications in the context of ITS. The Xilinx Virtex-5 FPGA was used in Hsiao et al. (2014) to run an SVM classifier for multiple-vehicle detection where a detection rate above 90% was reached, and in Sangeetha and Deepa (2017) for pedestrian detection where the histogram of edge oriented gradients feature extracture was used and followed by SVM. The Xilinx Virtex-6 ML605 and Xilinx Virtex-7 V2000T FPGAs were used in Giesemann et al. (2014) and Lee et al. (2016) for traffic sign detection with SVM. Another type of Xilinx FPGA is the Spartan-6 that was used to run SVM and AdaBoost classifiers in Hsiao et al. (2015) and Hsiao et al. (2016) respectively, for real-time pedestrian detection. Other Xilinx FPGA boards are the UltraScale FPGA used in Lyu et al. (2018) to train a CNN for real-time road segmentation, the Artix-7 FPGA used in Wu et al. (2019) to train an AdaBoost classifier for traffic light detection, and the Xilinx FPGA used in Chen et al. (2019) to train the EZNet, a CNN proposed for surrounding vehicle detection in autonomous vehicles, which achieved a mean average precision of 73.6% that is slightly higher than the identified similar work.

Another important family of FPGAs are the ones manufactured by Altera Corporation, now acquired by Intel. These FPGAs have been widely used in various ITS applications. For instance, the Altera Stratix II FPGA was used in Chen et al. (2013) to run an SVM classifier and a Histogram of Oriented Gradients feature extractor for human pedestrian detection, where a high detection accuracy above 95% was achieved. Another type of Altera FPGA is the Cyclone IV FPGA that was used in Khan et al. (2016) to run an SVM classifier for pedestrian detection, helping reduce the memory requirements to 0.34 Mbits. The Cyclone IV FPGA was also used in Jing et al. (2017) to run a feed-forward Neural Network (NN) for license plate recognition, achieving a processing time of 30.52 μs per license plate which is much lower than similar work that had processing times in the *ms* range.

Different types of MDs such as Computer Boards and MCUs have also been used in the literature. A Raspberry Pi 3B + and a Raspberry Pi 4 were utilized in Asmara et al. (2020) and Heredia and Barros-Gavilanes (2019) respectively to detect moving vehicles in real-time, where ML algorithms such as SVM and CNN were used. Also, a custom-made ARM-based MCU was used in Magrini et al. (2015) to run a Gaussian Mixture Model algorithm to estimate traffic flow in urban environments.

2.3. Specialized Devices

Extensive usage of GPUs to run various ML algorithms has been observed in ITS literature. The main motivation behind using GPUs is their specialized computational units that play a major role in accelerating computationally demanding tasks in computer vision applications. In this respect, there has been a very wide usage of the different GPUs manufactured by NVIDIA, which offers several lines of product. Among these GPUs is the NVIDIA Tesla series. The Tesla K20 was used in Chen et al. (2014) to support Histogram of Oriented Gradients SVM computations for real-time traffic sign detection and recognition, achieving a processing rate of 27.9 frames per second (fps). The Tesla K20C was used in Singh and Mohan (2018) to train a stacked auto-encoder for road accident detection with an accuracy of 77.5%. In Nguyen et al. (2016), the Tesla K40 was used to run a Tile Convolutional Neural Network (TCNN) and a Recursive TCNN (R-TCNN) for road obstacle detection, achieving a processing time of 70 *ms*. Similarly, the Tesla K40C was employed in Cecilia et al. (2018) where highly polluting drivers and traffic areas were identified with Fuzzy Minimals clustering which took 145.57 *sec* to execute. The Tesla K80 was used to run the Siamesed Fully Convolutional Networks for road detection in Wang et al. (2017) and a CNN for vehicle type classification in Shvai et al. (2020), where high accuracy levels of 98.1% and 99.03% were obtained, respectively.

Another Tesla model is the Tesla C2075 used for traffic sign recognition in both (Jin et al., 2014 and Lu et al., 2016). A CNN-based approach in Jin et al. (2014) achieved a high recognition rate of 99.65%, while a multi-modal tree-structure embedded multitask learning algorithm (M^2-tMTL) in Lu et al. (2016) took 0.5 *hours* in training compared to a CNN-based approach of the literature that took 37 *hours*. The Tesla P100 PICE was employed in Du et al. (2020) to train a deep Irregular Convolutional Residual Long Short-term Memory (ICRL), which accelerated the training process by 6 times.

Another line of research employed the NVIDIA Gefore GTX Titan X to run their selected ML algorithms. The Titan X was used in (Yuan et al., 2016; Tian et al., 2019; Zhu et al., 2017) for traffic sign detection. In Yuan et al. (2016), multi-class SVM achieved a high classification rate of 99.14%. A Multi-Scale Recurrent Attention

Neural Network was used in Tian et al. (2019) where a processing time of 600 *ms* was observed. Another type of NNs, namely the Fully-Convolutional Network was used in Zhu et al. (2017) where a lower processing time of 150 *ms* was achieved. The Titan X was also used for object detection in Wont et al. (2018) and Königshof et al. (2019). An Aggregated CNN was proposed in Wont et al. (2018) for real-time detection in road-driving scenarios, where high resolution images of 1024×320 were processed at a rate of 22 fps. Similar work was done in Königshof et al. (2019) using a CNN, where total latency remained below 100 *ms*. The Titan X was also used for vehicle model recognition in Fang et al. (2016) where a CNN model achieved a competitive 90% test accuracy. Line-CNN was introduced in Li et al. (2019) for traffic line detection, where a rate of 30 fps was achieved.

Other versions of the NVIDIA Titan series have also been explored, such as the Titan XP used in Deng et al. (2020) to run a Convolutional-Deconvolutional Neural Network that models drivers' eye movements and predicts driving fixation. Titan XP was also used for traffic sign detection in Li et al. (2018) where the proposed Squeeze-and-Excitation-ResNet was shown to work well in real-time with an average elapsed time of 292 *ms*. Another Titan model is the Titan Black used in Luo et al. (2017) for traffic sign recognition, where the developed CNN had a slow processing time of 6.44 *sec*.

Wide usage of the NVIDIA GTX Geforce 500, 700, and 900 series was found in the literature, despite the fact that they are considered to be relatively older than the devices found in the market or in the literature. The GTX 570 was used in Bulan et al. (2017) for license plate recognition with a CNN where an acceptable processing time of 2 *sec* was reached. The GTX 730 was used in Soon et al. (2018) for the recognition of 13 vehicle logos with a CNN that reached a high accuracy level of 99.1%. A CNN was also run on the GTX 760 in Wang and Zhou (2018) to recognize traffic lights and achieved a precision rate of 96.8% that was slightly higher than the identified similar work. The GTX 960 was employed in Chen et al. (2017), where an algorithm that uses K-means clustering to detect moving objects in driving scenarios was proposed and achieved a short capturing time of 10 *ms*. In Chen et al. (2017), the GTX 960 was used for turn signal detection with a CNN model that showed a high accuracy rate of 95%. The GTX 970 was used for nighttime vehicle detection, where the mean detection time of the CNN used was 0.42 *sec*. The GTX 980 was used in Xie et al. (2018) for CNN-based real-time car license plate detection, achieving a run time of 7.7 *ms*, and in Sanz-Madoz et al. (2019) for traffic sign recognition, where a reduced kernel Extreme Learning Machine method was proposed with the results showing a reduction of 10 times in execution time compared to a CPU-based simulation.

The GTX 10 series have also been widely used, as in Vu et al. (2017) where a CNN-based approach to detect traffic incidents was run on a GTX 1050 Ti, achieving a processing time of 83 fps. The GTX 1060 was used for object detection in Campoverde and Barros (2019) using a Deep Neural Network, and for predicting the intention of pedestrians crossing the road in Abughalieh and Alawneh (2020) where the developed CNN achieved a fair accuracy of 85%. The GTX 1080 was used in Wang et al. (2017) for vehicle type recognition, where a low accuracy of 55.13% was achieved for 20 vehicles types. Similarly in Ke et al. (2018), the GTX 1080 was used in traffic flow estimation using an ensemble classifier based on a CNN. The GTX 1080 was also used in Zang et al. (2018) for traffic speed estimation, using a Multiscale Spatio-Temporal Feature Learning network that achieved a mean relative error of 0.138. The GTX 1080 Ti was used in Lee and Min (2018) for traffic prediction, where the deployed Long Short-term Memory model achieved a root mean square error of 0.41. The GTX 1080 Ti was also used for a similar purpose in Zheng et al. (2019), where a model composed of CNN and Long Short-term Memory layers was proposed and achieved a mean absolute percentage error of 0.1721 which was the lowest among the various identified similar work. Similarly in Yu and Gu (2019), the GTX 1080 Ti was used for real-time traffic speed estimation, where the GCGA model used achieved a satisfactory mean absolute percentage error of 7.3%

Other considered GPUs include the NVIDIA Quadro K5200 that was used in Chung and Sohn (2017) to predict traffic density with a CNN model that achieved a low root mean square error of 4.150 after being trained at a rate of 13 *sec* per epoch. Another type of identified GPU usage is seen in Lee and Kim (2018) where Qualcomm's Snapdragon 820A processor with GPU utilization was deployed in traffic sign detection. The CNN model proposed in this work was able to run at a rate of 7 fps.

A Taiwan Semiconductor Manufacturing Company (TSMC) 90 nm Complementary Metal–Oxide–Semiconductor (CMOS) was considered in Weng and Chiu (2018) where it was used to run a Histogram of Oriented Gradients SVM classifier for traffic sign recognition. This ASIC device proved to be resource-efficient with a power consumption of 8 *mW* and achieved a real-time processing rate of 135 fps.

2.4. Hybrid Devices

Some investigations in the literature used System on Chip (SoC) platforms that are usually paired with traditional general purpose CPUs that enable Hardware/Software partitioning when it comes to implementing ITS applications. Specifically, Xilinx's Zynq-7000 SoC has been a desired platform due to its integration of the hardware programmability of FPGAs and software programmability of ARM processors, thus providing desirable co-design characteristics. In Khan et al. (2016), the Zynq-7000 was used to run an SVM classifier for vehicle type recognition, achieving a fast processing rate of 50 fps for 720×576 images. A similar approach is presented in Cabanes et al. (2019), where the Zynq-700 was used for SVM-based object detection and achieved a processing rate 300 times faster than software implementations. Similarly, the Zynq-7000 SoC was used in Dey et al. (2018) for traffic monitoring using CNN models. In Zhou et al. (2016), a novel Hardware/Software co-design architecture was implemented on a Zynq-7000 SoC to run an SVM classifier for real-time traffic signal recognition. Similar work was done in Vinh (2015), where a Zynq-7000 SoC, equipped with a quad-core ARM Cortex-A9, was used to run an SVM classifier for real-time traffic sign detection.

The NVIDIA Jetson TX1 supercomputer on module has been desirable for ITS applications where visual computing is of need. For instance, the Jetson TX1 was used in Han et al. (2017) for CNN-based traffic sign recognition, achieving an accuracy of 96% but a low processing rate of 1.6 fps. The Jetson TX1 was also used in Wang et al. (2019) for vehicle type recognition with a faster region-based CNN model, achieving a processing rate of 354 ms and accuracy levels around 90%. Similar work was done in Mhalla et al. (2018) where the Jetson TX1 module was used for object detection in traffic scenarios using a region-based CNN model.

3. Taxonomies and Performance Models

The richness of the survey in Section 2 inspired the synthesis of a taxonomy of ITS applications, ML techniques, and MHDs commonly used in the literature of Vehicle Control and Traffic Management. Similarly, the performance metrics used in the literature influenced the development of a standalone taxonomy of ITS, ML, and MHD indicators, as shown in Table 2. This table presents these indicators as either complexity or competency indicators, and highlights the articles which considered some of these indicators

Table 2
Taxonomy of Indicators Describing the Complexity or Competency of ITS Applications, MHDs, or ML Techniques.

Indicator	Indicator	Type		Ref.	Consideration in the Literature
		Complexity	Competency		
ITS1	Vehicle & Traffic Efficiency	✓		Kaparias et al. (2011)	
ITS2	Energy	✓		Kaparias et al. (2011)	
ITS3	Operational Load	✓		Asmara et al. (2019)	
ITS4	Infrastructure & Deployment Scale	✓		Kaparias et al. (2011)	
ITS5	Data	✓		Zhu et al. (2018)	
ITS6	Node Resources	✓		Rouabeh et al. (xxxx)	
ITS7	Safety Services	✓		Asmara et al. (2019)	
HW1	Frequency		✓	Damaj (2007)	(Khan et al., 2016; Weng and Chiu, 2018)
HW2	Area		✓	Zhu et al. (2018)	
HW3	Power Consumption		✓	Damaj and Kasbah (2018)	(Weng and Chiu, 2018; Alonso et al., 2015; Magrini et al., 2015; Khan et al., 2016; Vasconcelos et al., 2017; Wu et al., 2019; Chen et al., 2019)
HW4	Memory Capacity		✓	Greathouse and Loh (2018)	(Heredia and Barros-Gavilanes, 2019; Lee et al., 2016; Alonso et al., 2015; Vasconcelos et al., 2017)
HW5	Interfacing Capability		✓	Malone et al. (2011)	
HW6	Time Factor		✓	Damaj and Kasbah (2018)	
ML1	Complexity	✓		Klaine et al. (2017)	
ML2	Accuracy		✓	Goutte et al. (2005)	(Chen et al., 2013; Hsiao et al., 2014; Dairi et al., 2018; Sangeetha and Deepa, 2017; Hsiao et al., 2016; Hsiao et al., 2015; Chen et al., 2019; Heredia and Barros-Gavilanes, 2019; Hassannejad et al., 2015; Königshof et al., 2019; Wont et al., 2018; Chen et al., 2017; Kosaka and Ohashi, 2015: Kuang et al., 2016; Nguyen et al., 2016; Chen and Huang, 2016; Gudigar et al., 2019; Han et al., 2017; Weng and Chiu, 2018; Chen et al., 2014; Tian et al., 2019; Wang and Zhou, 2018; Sanz-Madoz et al., 2019;Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Jin et al., 2014; Lu et al., 2016; Yuan et al., 2016; Shi et al., 2015; Vasconcelos et al., 2017; Deng et al., 2020; Outay et al., 2019; Lyu et al., 2018; Li et al., 2019; Wang et al., 2017; de Paula and Jung, 2015; Abughalieh and Alawneh, 2020; Gkolias and Vlahogianni, 2019; Chen et al., 2017; Panahi and Gholampour, 2016; Wang et al., 2019; Campoverde and Barros, 2019; Jing et al., 2017; Soon et al., 2018; Shvai et al., 2020; Mhalla et al., 2018; Wang et al., 2017; Xie et al., 2018; Hu et al., 2015; Fang et al., 2016; Bulan et al., 2017; Magrini et al., 2015; Dey et al., 2018; Yu and Gu, 2019; Ke et al., 2018; Chung and Sohn, 2017; Shi et al., 2016; Vu et al., 2017; Ke et al., 2018; Singh and Mohan, 2018; Lee and Min, 2018; Du et al., 2020; Zang et al., 2018; Zheng et al., 2019; Zhan et al., 2018)
ML3	Training Data		✓	Klaine et al. (2017)	
ML4	Scalability		✓	Klaine et al. (2017)	
ML5	Additional Abilities		✓	Zappone et al. (2019)	
ML6	Time Factor	✓		Klaine et al. (2017)	(Cabanes et al., 2019; Heredia and Barros-Gavilanes, 2019; Hassannejad et al., 2015; Königshof et al., 2019; Wont et al., 2018; Kosaka and Ohashi, 2015; Gudigar et al., 2019; Han et al., 2017; Giesemann et al., 2014; Weng and Chiu, 2018; Tian et al., 2019; Wang and Zhou, 2018; Sanz-Madoz et al., 2019;Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Jin et al., 2014; Lu et al., 2016; Shi et al., 2015; Vasconcelos et al., 2017; Outay et al., 2019; Lyu et al., 2018; Wang et al., 2017; de Paula and Jung, 2015; Chen et al., 2017; Jing et al., 2017; Soon et al., 2018; Mhalla et al., 2018; Wang et al., 2017; Xie et al., 2018; Hu et al., 2015; Yu and Gu, 2019; Ke et al., 2018; Chung and Sohn, 2017; Cecilia et al., 2018; Vu et al., 2017; Zhan et al., 2018)
ML7	Additional Needs	✓		Zappone et al. (2019)	
ML8	Robustness		✓	Xu and Mannor (2011)	

as performance metrics. The proper assessment of ITS applications is integral in determining the suitable ML techniques and MHDs needed for implementation.

3.1. ITS Taxonomy and Performance Model

The thorough survey of ITS applications within the wide scopes of Vehicle Control and Traffic Management exhibits a set of discrete subcategories, presented in Fig. 2, that constitute the common problems tackled in the literature, grouped by their respective goals. In Vehicle Control, vehicle analysis refers to data acquisition and analysis for the driver or vehicle, instruction recognition relates to traffic light and sign recognition, while driver assistance includes vehicle tasks that do not fall under the other subcategories. For Traffic Management, traffic detection implies detecting vehicle type or number plate, while optimization includes studying the road condition, weather, air quality, among others. The rest of the subcategories are self-explanatory.

The literature also presents the most used performance indicators for ITS applications, listed in Table 2, which led to the development of the later performance evaluation model in Table 3. This model evaluates the complexity of a system based on its target vehicle and traffic efficiency, energy consumption, operational load, infrastructure and deployment scale, data and node resources, and safety services. The complexity is evaluated as low, moderate, or high, as shown in Table 4, using the sub-indicators inspired by previous work. This evaluation highlights the prospective system requirements and provides a unified approach when choosing the appropriate ML techniques and MHDs for ITS implementations.

3.2. ML Taxonomy and Performance Model

The various learning techniques used in the literature for ITS applications with MHD implementations are grouped into four categories, as illustrated in Fig. 2, which are regression, classification,

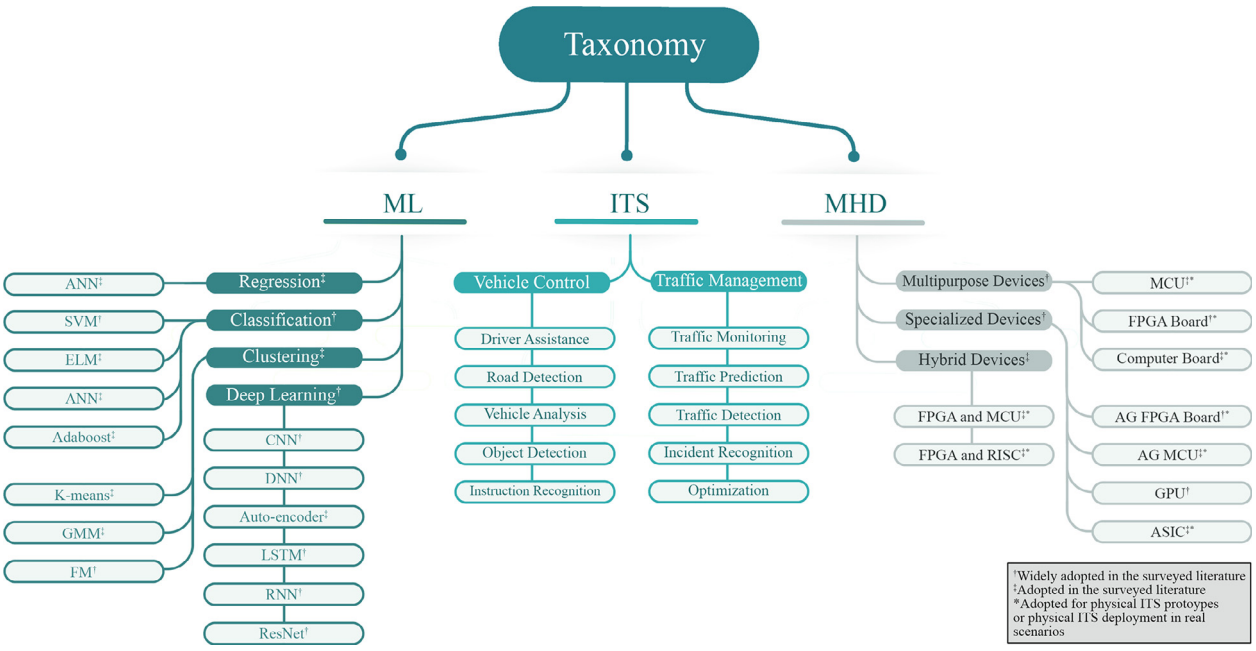

Fig. 2. A taxonomy for MHDs and ML techniques utilized in the surveyed ITS applications, where ELM stands for Extreme Learning Machine, GMM for Gaussian Mixture Model, FM for Fuzzy Minimals, DNN for Deep Neural Network, LSTM for Long Short-term Memory, and RNN for Recurrent Neural Network.

Table 3
The proposed levels of complexity of ITS indicators.

Indicator	Complexity Levels		
	Low	Moderate	High
Vehicle & Traffic Efficiency	• Low average travel time • Short queue length • Few congestion occurences	• Moderate average travel time • Moderate queue length • Some congestion occurences	• High average travel time • Large queue length • Many congestion occurences
Energy	• Low fuel consumption	• Moderate fuel consumption	• High fuel consumption
Operational Load	• Low peak hour traffic flow	• Moderate peak hour traffic flow	• High peak hour traffic flow
Infrastructure & Deployment Scale	• Limited road coverage by ITS units • Low number and width of tracks	• Moderate road coverage by ITS units • Moderate number and width of tracks	• Large road coverage by ITS units • High number and width of tracks
Data	• Low volume of data needed • Numeric data • Low availability of data	• Moderate volume of data needed • Numeric and/or acoustic and/or visual data • Moderate availability of data	• High volume of data needed • Audio-visual and/or time-series data • High availability of data
Node Resources	• Small number of computational units	• Moderate number of computational units	• Large number of computational units
Safety Services	• Low accident rate • Small number of violations • Limited coverage by safety services	• Moderate accident rate • Moderate number of violations • Moderate coverage by safety services	• High accident rate • Large number of violations • Large coverage by safety services

Table 4
The complexity scheme for the proposed ITS performance levels.

Evaluation	Indicator Levels
Complex: The problem is highly demanding in terms of resources and service requirements	Most indicators are in high level and none are low
Somewhat Complex: The problem is somewhat demanding in terms of resources and service requirements	Most indicators are in moderate level
Simple: The problem is undemanding in terms of resources and service requirements	Most indicators are in low level and none are high

clustering, and DL. This grouping is based on the input–output mapping relationship, where regression predicts real-valued labels of unseen data, classification labels unseen data given a discrete set of possible classes, and clustering identifies patterns or regions within a dataset. These categories incorporate classical ML algorithms in addition to simple ANN architectures. A fourth DL category is proposed to include more sophisticated and complex NN architectures, which may be used for any of the other three categories.

The surveyed papers utilized various performance indicators for the ML techniques deployed. These include, among others, the following:

- Accuracy (Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Jin et al., 2014; Lu et al., 2016; Yuan et al., 2016; Shi et al., 2015; Vasconcelos et al., 2017; Deng et al., 2020; Outay et al., 2019; Lyu et al., 2018; Li et al., 2019; Wang et al., 2017; de Paula and Jung, 2015; Abughalieh and Alawneh, 2020; Gkolias and Vlahogianni, 2019; Chen et al., 2017; Panahi and Gholampour, 2016; Wang et al., 2019; Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Jin et al., 2014; Lu et al., 2016; Yuan et al., 2016; Shi et al., 2015; Vasconcelos et al., 2017; Deng et al., 2020; Outay et al., 2019; Lyu et al., 2018; Li et al., 2019; Wang et al., 2017; de Paula and Jung, 2015; Abughalieh and Alawneh, 2020; Gkolias and Vlahogianni, 2019; Chen et al., 2017; Panahi and Gholampour, 2016; Wang et al., 2019; Campoverde and Barros, 2019; Jing et al., 2017; Soon et al., 2018; Shvai et al., 2020; Mhalla et al., 2018; Wang et al., 2017;Xie et al., 2018;Hu et al., 2015;Fang et al., 2016;Bulan et al., 2017;Magrini et al., 2015;Dey et al., 2018;Yu and Gu, 2019;Ke et al., 2018;Chung and Sohn, 2017;Shi et al., 2016;Vu et al., 2017;Ke et al., 2018;Singh and Mohan, 2018;Lee and Min, 2018;Du et al., 2020;Zang et al., 2018;Zheng et al., 2019; Zhan et al., 2018)
- Precision (Chen et al., 2019;Heredia and Barros-Gavilanes, 2019;Wont et al., 2018;Chen and Huang, 2016;Tian et al., 2019;Li et al., 2018;Wang and Zhou, 2018;Lee and Kim, 2018;Zhu et al., 2017;Yuan et al., 2016;Shi et al., 2015; Vasconcelos et al., 2017;Outay et al., 2019;Lyu et al., 2018;Li et al., 2019;Wang et al., 2017;Abughalieh and Alawneh, 2020; Chen et al., 2017;Wang et al., 2019;Campoverde and Barros, 2019;Xie et al., 2018;Ke et al., 2018;Shi et al., 2016;Ke et al., 2018)
- Recall (Chen et al., 2019;Heredia and Barros-Gavilanes, 2019; Chen and Huang, 2016; Tian et al., 2019; Li et al., 2018; Wang and Zhou, 2018; Lee and Kim, 2018; Luo et al., 2017; Zhu et al., 2017; Yuan et al., 2016; Shi et al., 2015; Vasconcelos et al., 2017; Outay et al., 2019; Li et al., 2019; Wang et al., 2017; Wang et al., 2019; Campoverde and Barros, 2019; Shvai et al., 2020; Xie et al., 2018; Ke et al., 2018; Shi et al., 2016; Ke et al., 2018)
- F1 score (Chen et al., 2019; Heredia and Barros-Gavilanes, 2019; Lyu et al., 2018; Shi et al., 2016; Ke et al., 2018)
- Mean absolute error (Li et al., 2018; Chung and Sohn, 2017; Zang et al., 2018; Zheng et al., 2019; Zhan et al., 2018), mean square error (Sanz-Madoz et al., 2019), root mean square error (Chung and Sohn, 2017; Lee and Min, 2018; Zheng et al., 2019), and average vertex error (Lee and Kim, 2018)
- Area under curve (Dairi et al., 2018; Tian et al., 2019;Lee and Kim, 2018; Deng et al., 2020; Singh and Mohan, 2018; Gkolias and Vlahogianni, 2019), intersection over union (Königshof et al., 2019; Zhu et al., 2017; Campoverde and Barros, 2019; Xie et al., 2018), and overlapping (Chen et al., 2017; Li et al., 2018; Lee and Kim, 2018; Chen et al., 2017)
- True positive rate (Dairi et al., 2018; Mhalla et al., 2018; Bulan et al., 2017; Singh and Mohan, 2018), false positive rate (Dairi et al., 2018; Chen et al., 2014; Li et al., 2019; Bulan et al., 2017; Singh and Mohan, 2018), and false positives per image (Khan et al., 2016; Sangeetha and Deepa, 2017; Kuang et al., 2016; Chen et al., 2017; Campoverde and Barros, 2019; Mhalla et al., 2018)
- Miss rate (Khan et al., 2016; Sangeetha and Deepa, 2017; Kuang et al., 2016; Chen et al., 2017; Campoverde and Barros, 2019) and false alarm rate (Nguyen et al., 2016; Chen et al., 2017; Panahi and Gholampour, 2016)
- Specificity (Outay et al., 2019;Gkolias and Vlahogianni, 2019) and sensitivity (Gkolias and Vlahogianni, 2019)
- Detection rate (Hsiao et al., 2016; Hassannejad et al., 2015; Kuang et al., 2016; Nguyen et al., 2016; Weng and Chiu, 2018; Chen et al., 2014; Wang and Zhou, 2018; Yuan et al., 2016; Chen et al., 2017; Singh and Mohan, 2018) and recognition rate (Gudigar et al., 2019; Weng and Chiu, 2018; Chen et al., 2014; Jin et al., 2014)
- Processing time (Hassannejad et al., 2015; Kosaka and Ohashi, 2015; Han et al., 2017; Wang and Zhou, 2018; Jing et al., 2017; Vu et al., 2017), training time (Jin et al., 2014; Lu et al., 2016; Soon et al., 2018; Yu and Gu, 2019; Zheng et al., 2019), and testing time (Lu et al., 2016)
- Throughput (Chen et al., 2013; Chen et al., 2019; Giesemann et al., 2014; Wu et al., 2019), computing efficiency (Hsiao et al., 2016; Hsiao et al., 2015), and complexity (Wu et al., 2019; Xie et al., 2018)
- Operating frequency (Khan et al., 2016; Weng and Chiu, 2018), frame rate (Khan et al., 2016; Heredia and Barros-Gavilanes, 2019; Lee and Kim, 2018; Chen et al., 2017; Ke et al., 2018), and latency (Outay et al., 2019)

Thus, a holistic list of ML performance metrics is synthesized in Table 2 to describe their competency and complexity. Then, inspired by these metrics, a performance evaluation model of ML techniques is proposed in Table 5. This model evaluates an ML technique in terms of competency and complexity, given its estimated accuracy, training data requirements, scalability, additional abilities, time factor, additional needs, and robustness. As shown in Table 6, the algorithm is evaluated as highly competent, competent, or modest. It is also evaluated as complex, somewhat complex, or simple, as shown in Table 7. The competency of the ML technique describes its performance, whereas its complexity describes its computational and resource-demand characteristics. Such an evaluation provides details on the expected performance of the ML technique to facilitate the choice of model.

3.3. MHD Taxonomy and Performance Model

The various MHDs showcased in the literature are categorized into three classes, namely MDs, SDs, and HDs. These classes, shown

Table 5
The proposed levels of performance of ML indicators.

	Competency/Complexity Levels		
Indicator	Low	Moderate	High
Complexity	• Logarithmic low or logarithmic high	• Linear or almost quadratic	• Higher than quadratic or exponential
Accuracy	• Low accuracy • Low F-score	• Moderate accuracy • Moderate F-score	• High accuracy • High F-score
Training Data	• Small datasets • Numerical data • Low dimensionality	• Moderate-sized datasets • Numerical and/or acoustic and/or visual data • Moderate dimensionality	• Large datasets • Acoustic and/or visual data • High dimensionality
Scalability	• Non-distributed architecture • Sequential processing • Low number of hyperparameters	• Distributed and non-distributed architectures • Sequential and parallel processing • Moderate number of hyperparameters	• Distributed architecture • Parallel processing • High number of hyperparameters
Additional Abilities	• Does not support transfer learning • Cannot learn non-linear relationships	• Does not support transfer learning • Can learn non-linear relationships	• Supports transfer learning • Can learn non-linear relationships
Time Factor	• Long response time • Short training time	• Moderate response time • Moderate training time	• Short response time • Long training time
Additional Needs	• Requires pre-processing • No extensive hyperparameter tuning	• Requires pre-processing • Hyperparameter tuning	• Does not require pre-processing • Extensive hyperparameter tuning
Robustness	• Prone to overfitting	• Somewhat prone to overfitting	• Not prone to overfitting

in Fig. 2, comprise devices that are an integral part of any ITS system, each possessing a distinct bouquet of features that determines the device's competency.

In MDs, MCUs, which are regarded as low-power devices, offer limited computational capabilities, unlike the more sophisticated Computer Boards that can handle demanding computational tasks. MDs also include FPGAs that maintain programmable logic blocks, offering hardware reconfigurability after manufacturing. While MDs carry out general-purpose tasks, SDs are concerned with specific tasks for targeted applications. In SDs, AG FPGA boards and MCUs incorporate dedicated computational units that facilitate ITS applications, mitigating the complexity of their problems. This class also includes GPUs, which are used in computer vision applications, and ASICs, which possess specially designed functional units for a particular use. A third MHD class is proposed to address HDs that associate combinations of MDs and SDs.

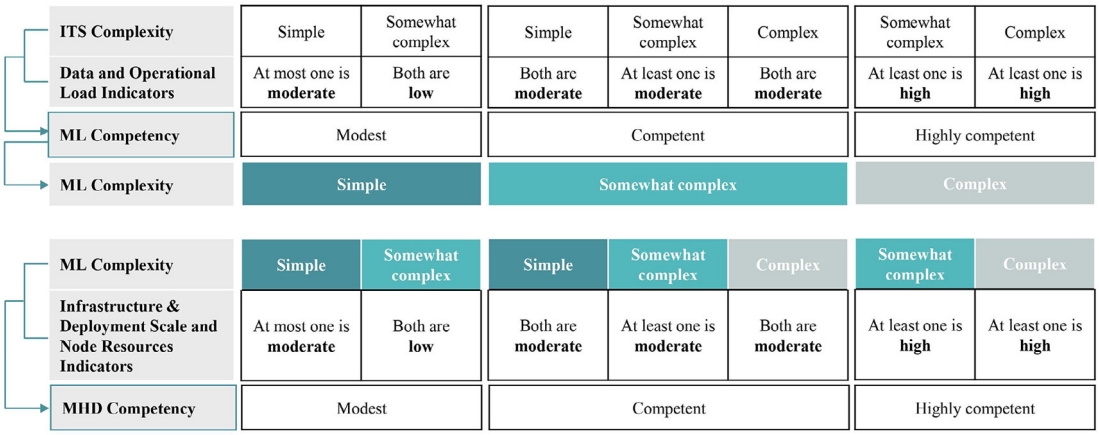

Fig. 3. A cross-matching evaluation chart that determines the competency of the MHD and ML technique needed to satisfy a pre-determined ITS complexity.

Table 6
The competency scheme for the proposed ML performance levels.

Evaluation	Indicator Level
Highly Competent: The technique holistically attains high performance	Most indicators are in high level and none are low
Competent: The technique holistically attains satisfactory performance	Most indicators are in mid-range level
Modest: The technique holistically attains low performance	Most indicators are in low level and none are high

Table 7
The complexity scheme for the proposed ML performance levels.

Evaluation	Indicator Level
Complex: The technique is computationally intensive and highly demanding in terms of resources	Most indicators are in high level and none are low
Somewhat Complex: The technique is computationally intensive and demanding in terms of resources	Most indicators are in mid-range level
Simple: The technique is neither computationally intensive nor demanding in terms of resources	Most indicators are in low level and none are high

To better assess the competency of MHDs, a performance evaluation model, as shown in Table 8, is developed based on thoughtfully chosen performance indicators listed in Table 2. This model evaluates the competency of a device based on its frequency, area, power consumption, memory capacity, interfacing capability, and time factor. It is important to note that some of the metrics in this evaluation model vary among different devices. For instance, the area is expressed in CUDA cores for GPUs, Logic Slices for FPGAs, and General Purpose Processing Cores for Computer Boards. Following the proposed evaluation model, hardware devices of different classes are evaluated as highly competent, competent, or modest as shown in Table 9. This approach lays out a robust evaluation method that aids in selecting a MHD that caters for the computational demands of a given ITS.

4. Proposed Framework

In this section, we propose an evaluation framework, stimulated by the taxonomies, and adapted from the evaluation model of Damaj et al. (Damaj et al., 2019; Damaj et al., 2020; Damaj and Kasbah, 2018; Damaj, 2016; Damaj et al., 2020). The performance indicators presented in the taxonomies of Section 3 are used to develop complexity and competency levels with a three-point scale. The overall complexity of an ITS application and an ML technique as well as the competency of an MHD and an ML technique can be individually evaluated using theses schemes. Furthermore, a cross-matching evaluation chart that integrates the different schemes to determine the needed ML and MHD competency for a desired ITS application is introduced to fulfill the sixth research objective mentioned in Section 1. The specific levels of performance of the ML technique and the MHD to be chosen can be then deduced in a process of back-tracking from the competency schemes to the performance levels.

4.1. Cross-Matching Evaluation Chart

The cross-matching evaluation chart in Fig. 3 is developed to deduce the recommended competency of the ML technique and MHD to be utilized for an ITS application. The complexity of an ITS application can be determined using Table 4 alongside the levels of Data and Operational Load indicators matched with their suitable ML competency. These ITS indicators encapsulate the volume and type of data the ML model has to deal with, as well as the response time expected from this model. Since they are integral in determining the desired ML competency, these indicators are singled out to ensure that they are of certain levels as per Table 3. All possible exclusive combinations of levels are explored, where the levels of the Data and Operational Load specified do not contradict the definitions of the corresponding ITS complexities as shown in Table 4. The ML complexity can be then easily inferred from the determined competency, as their indicators vary proportionally, e.g., Training Data (Competency) and Time Factor (Complexity). Hence, a modest ML technique is mapped to a simple ML technique, a competent one to a somewhat complex one, and a highly competent one to a complex one.

The MHD competency is deduced using the previously determined ML complexity and the Infrastructure & Deployment Scale and Node Resources indicators. These indicators were carefully chosen as they clearly indicate the competency needed in an MHD. Moreover, the different scenarios are presented in an exclusive manner and matched with their corresponding MHD competency, varying from modest to highly competent. After determining the needed ML and MHD competencies, the MHD and ML technique to be used can be determined using Tables 5–7,8.

Table 8
The proposed levels of performance of MHD indicators.

Indicator	Competency Levels		
	Low	Mid-range	High
Frequency	• Low Frequency Ranges from 450 MHz or lower to 1.5 GHz or higher in FPGAs (IntelProduct Catalog, 2021; HWDEVICE6) Ranges from 700 MHz or lower (Buy a raspberry pi zero w, 2020) to 1.5 GHz or higher (Raspberry Pi, 2021) in Computer Boards Ranges from 1.12 GHz (GeForce GT 1030, 2020) or lower to 1.54 GHz (GeForce RTX, 2020) or higher in GPUs	• Mid-range Frequency	• High Frequency
Area	• Small number of computational units Ranges from 391,996 Logic Elements to 813,000 Logic Elements or higher in FPGAs (IntelProduct Catalog, 2021; HWDEVICE6) Ranges from 1 General Purpose (Buy a raspberry pi zero w, 2020) Core to 4 General Purpose Cores (Buy a raspberry pi zero w, 2020) or more in Computer Boards Ranges from 384 (GeForce GT 1030, 2020) CUDA Cores to 4,352 CUDA Cores (GeForce RTX, 2020) or higher in GPUs	• Adequate number of computational units	• Large number of computational units
Power Consumption	• Not energy efficient Ranges from 2.1 Watts (Buy a raspberry pi zero w, 2020) to 6.4 Watts (Raspberry Pi, 2021) or Higher in Computer Boards Ranges from 75 Watts (GeForce GT 1030, 2020) to 250 Watts (Buy a raspberry pi zero w, 2020) or Higher in GPUs No Power Consumption ranges in FPGAs since this indicator is calculated per design	• Somewhat energy efficient	• Energy efficient
Memory Capacity	• Can process low volumes of data • Provides a low memory bandwidth Ranges from 512 MBs or less (Buy a raspberry pi zero w, 2020) to 4 GBs or more (Raspberry Pi, 2021) in Computer Boards Ranges from 2 GBs to 4 GBs or More in FPGAs (IntelProduct Catalog, 2021; HWDEVICE6) Ranges from 2 GBs (GeForce GT 1030, 2020) to 11 GBs (Buy a raspberry pi zero w, 2020) or More in GPUs	• Can process moderate volumes of data • Provides mid-range memory bandwidth	• Can process high volumes of data • Provides a high memory bandwidth
Interfacing Capability	• Small number of IO ports • A single or none communication interfaces • Only traditional communication standards Ranges from 284 programmable ports to 2,304 programmable ports or More in FPGAs (IntelProduct Catalog, 2021; HWDEVICE6) Ranges from 14 (Buy a raspberry pi zero w, 2020) programmable ports to 40 (Raspberry Pi, 2021) programmable ports or More in Computer Boards N/A in GPUs	• Mid-range number of IO ports • A couple communication interfaces • Modern communication standards	• Large number of IO ports • A few communication interfaces • Latest communication standards
Time Factor	• High execution time • Low maximum frequency No ranges are available since Execution Time, and Maximum Frequency indicators are calculated per design	• Mid-range execution time • Mid-range maximum frequency	• Low execution time • High maximum frequency

Table 9
The competency scheme for the proposed MHD performance levels.

Evaluation	Indicator Level
Highly Competent: is a high-end device	Most indicators are in high level
Competent: is a mid-range device	Most indicators are in mid-range level
Modest: is a low-end device	Most indicators are in low level

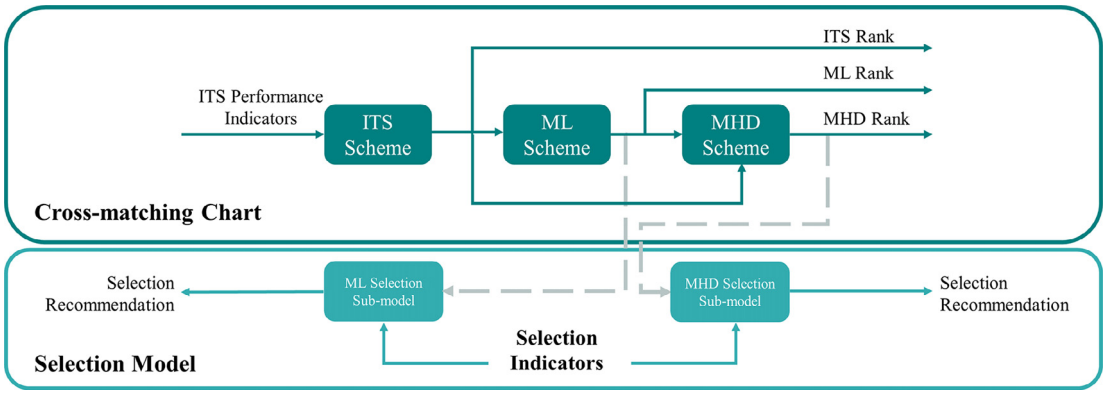

Fig. 4. A conceptual selection model that can be integrated with the proposed cross-matching evaluation model.

4.2. Selection Model Concept

The cross-matching evaluation model presented above aims at ranking ML techniques and MHDs that promise a certain level of performance per ITS complexity. A conceptual selection model is proposed in Fig. 4 with the purpose of selecting (or recommending) the ML technique and MHD suitable for an ITS application. The proposed selection model concept highlights the need for additional indicators that are beyond the cross-matching chart and the performance aspect of the evaluation process, which include the scope of application (real deployment or testing and validation), budget, and the choice of technology that matches the problem type. The MHD selection sub-model takes into consideration indicators that were not included in the cross-matching framework, such as, processor (general purpose versus specific purpose) and IC technologies (Programmable Logic Devices versus ASICs), as well as the suitable MHD ranks obtained from the cross-matching chart. It is worthy of note that the MHD selection indicator that chiefly influences the selection process is the cost of target device. As for the ML selection sub-model, it considers the type of problem (supervised learning, unsupervised learning, or reinforcement learning) and its operation scenario, as well as the suitable ML technique ranks inferred from the cross-matching chart. The proposed sub-models can combine heterogeneous, qualitative, and quantitative indicators (Damaj et al., 2019, 2021; Damaj and Kasbah, 2018). Furthermore, the sub-models can be formulated using traditional decision-making techniques based on multiple criteria as combined statistically (Damaj et al., 2019) or using machine learning (Damaj et al., 2021). It is worth noting that the cross-matching evaluation model and the selection model were kept distinct to establish separation of concern; as the first demonstrates a focus on performance while the latter aims at capturing the concept of aiding selection recommendations.

5. Discussion

The work presented in this investigation demonstrates significance on several levels. As shown in Section 2, a large body of literature has sought to provide advancements in implementing ITS applications using modern ML techniques, and thus a comprehensive survey of this literature is of importance. It is also worth mentioning that this survey, unlike similar surveys, has a special focus on the utilized MHDs. Looking at implementation of an ITS application through the lens of MHDs is vital, especially when considering the real-world deployment of this application. Despite its importance, it is apparent that the selection of ML techniques and MHDs by designers in the literature is rarely justified or dwelt upon. The proposed evaluation framework enables designers to make well-informed choices to attain high performance. Furthermore, the presented taxonomy of ML techniques and MHDs highlights techniques and devices that are over-utilized and those that are yet to be used. To further discuss our presented work, this section includes a validation procedure that tests the performance of the proposed evaluation model, an elaboration on common and good practices found in the literature, and a layout of limitations and future work recommendations.

5.1. Validation

To assess the performance of our proposed framework, we follow a validation procedure in which we reason about the authors' choices in the surveyed articles for ML and MHD competencies with those obtained using the cross-matching evaluation chart of Fig. 3. We set the following criteria for choosing references from the surveyed literature. The selected references must cover all types of MHDs, ML techniques, and ITS applications as per the proposed taxonomy in Fig. 2. The references should provide enough information to assess the complexity and competency of the ITS application, ML technique, and MHD used through the proposed indicators in Table 2 and frameworks. Additionally, to eliminate redundancy, we consider each member of the taxonomy only twice, resulting in a robust validation of 16 references.

The outcome of our validation is illustrated as radar charts in Fig. 5, 6, which present the classification outcome based on the authors' choices and the proposed cross-matching evaluation chart. The difference between the authors' choices and the result of the cross-matching evaluation chart is quantified in Fig. 7 in terms of the Difference in Classification metric (δ_c) (Damaj et al.,

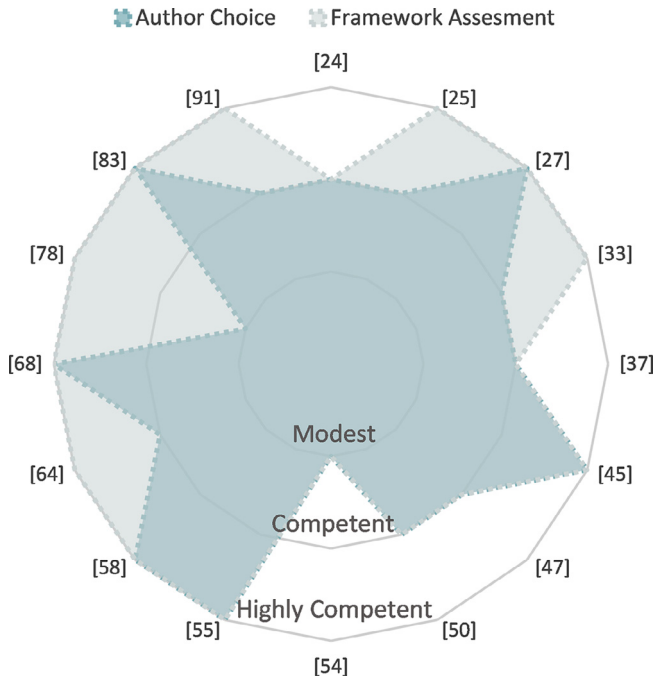

Fig. 5. ML Competency versus reference. Concentric circles depict the scale Modest (innermost circle), Competent, and Highly Competent.

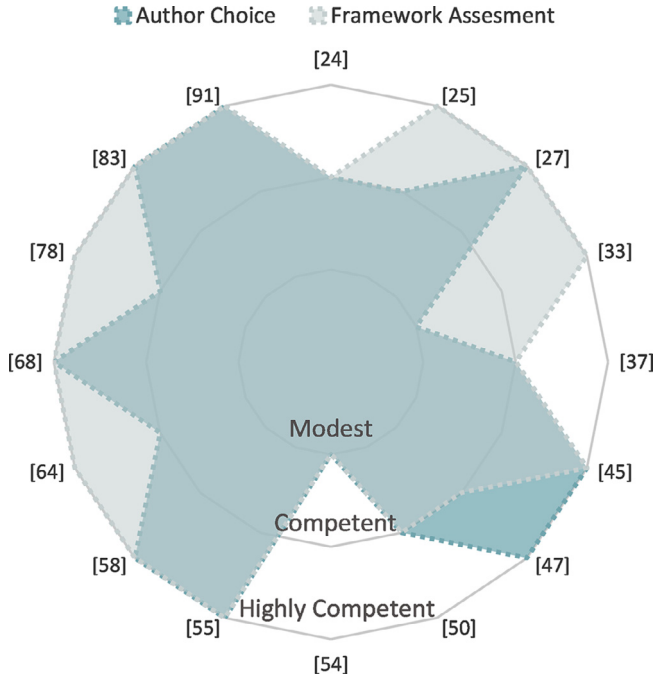

Fig. 6. MHD Competency versus reference. Concentric circles depict the scale Modest (innermost circle), Competent, and Highly Competent.

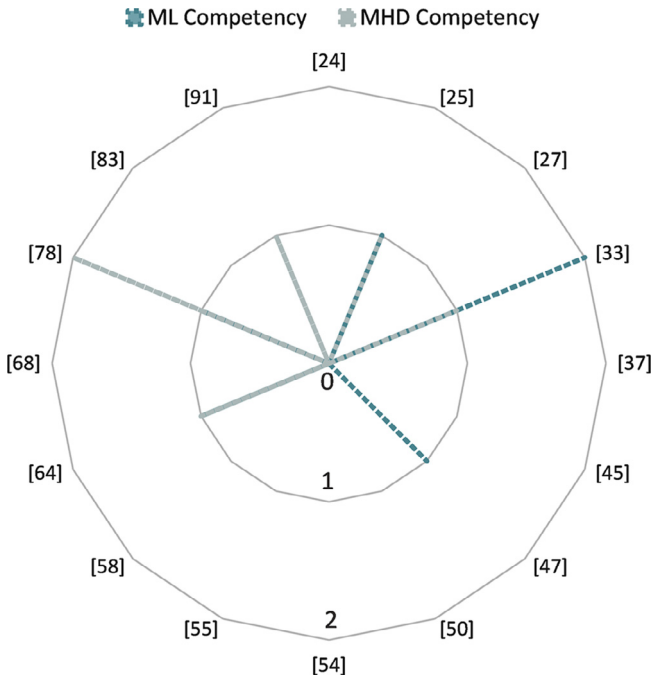

Fig. 7. Difference in Evaluation (δ_c). The concentric circles depict the scale: Nil (0), Small (1), and Significant (2).

2019). This metric is the absolute difference in scale levels which could be either 0 (Nil), 1 (Small), or 2 (Significant). The results show that 62.5% of the references have a Nil δ_c, 25% have a Small δ_c, and only 12.5% have a Significant δ_c, which shows that most implicit evaluations done by the authors of these papers are in conformity with the proposed evaluation model.

The differences in classification between the authors' choices and our proposed framework are due to several reasons. To begin with, our framework is tailored to recommend the ML and MHD competency choices that promise high performance due to intrinsic characteristics and specification, disregarding factors such as price, re-usability, and practicality (Hsiao et al., 2016; Chen et al., 2017). For instance, (Hsiao et al., 2016) used an FPGA to implement a processor that is tailored for human detection. Using a GPU or a hybrid system could promise higher performance; however, FPGAs offer higher flexibility, lower cost, and a much-desired practicality for producing a standalone SoC. Furthermore, in some investigations, such as in Lee and Kim (2018), the grade of performance-level of the chosen hardware device would be merely for the purpose of testing and experimenting with a field programmable device, regardless of the requirements needed in real ITS deployment scenarios.

In special cases, such as in Magrini et al. (2015), although images were used for training, pixel intensities were clustered for traffic flow detection. This problem is simpler than conventional image recognition tasks, justifying the choice of Gaussian Mixture Model instead of more sophisticated techniques such as CNNs. Such a case is not detectable by our framework since it considers all problems dealing with images as somewhat complex or complex, as detailed in Table 3. This issue also affects the choice of MHDs, where in Magrini et al. (2015), a modest device was used since the ML technique used was simple, while our framework recommended selecting a highly competent MHD. Additional work, where a classification difference was observed, include (Singh and Mohan, 2018; Khan et al., 2016); no clear justification or reasoning for the authors' choices was provided.

5.2. Common Practices

A common practice observed in the surveyed papers, namely in (Lyu et al., 2018; Heredia and Barros-Gavilanes, 2019; Nguyen et al., 2016; Shvai et al., 2020; Jin et al., 2014; Fang et al., 2016; Königshof et al., 2019; Li et al., 2019; Deng et al., 2020; Luo et al., 2017; Bulan et al., 2017; Soon et al., 2018; Wang and Zhou, 2018; Chen et al., 2017; Kuang et al., 2016; Xie et al., 2018; Vu et al., 2017; Abughalieh and Alawneh, 2020; Wang et al., 2017;

Table 10
Papers attaining high accuracy values in the surveyed literature for various ITS applications.

ITS Application	Ref.	ML	Acc.
Traffic Detection	Soon et al. (2018)	CNN	98.10%
Road Detection	Li et al. (2019)	CNN	96.87%
Traffic Detection	Han et al. (2017)	CNN	96.20%
Object Detection	Kuang et al. (2016)	CNN	95.95%
Vehicle Analysis	Vasconcelos et al. (2017)	K-means	95.45%
Instruction Recognition	Li et al. (2018)	ResNet	95.20%
Driver Assistance	Chen et al. (2017)	CNN	95.00%
Instruction Recognition	Yuan et al. (2016)	SVM	92.00%
Driver Assistance	Abughalieh and Alawneh (2020)	CNN	85.00%
Traffic Prediction	Zang et al. (2018)	CNN	83.00%

Ke et al., 2018; Zang et al., 2018; Zheng et al., 2019; Chung and Sohn, 2017; Lee and Kim, 2018; Dey et al., 2018; Han et al., 2017; Wang et al., 2019; Mhalla et al., 2018), is the selection of CNNs as the adopted ML technique for solving various ITS problems that rely on visual data as input. This pattern is repeatedly observed throughout the surveyed literature; Table 10 presents the investigations that attained the highest reported accuracy with a variety of ITS applications. The investigations listed in the tables confirms the domination of CNNs as the preferred choice among ML techniques. It is commonly known in the ML literature that CNNs promise high performance in detection and recognition without necessitating feature extraction as a pre-processing step. Additionally, CNNs and neural network-based solutions are highly flexible in terms of their architecture and can be tailored for a desired application. State-of-the-art CNN models are also widely available and pre-trained on large-scale image datasets. Hence, they can be fine-tuned on application-specific datasets in the ITS domain. From a hardware perspective, neural networks are considered to be embarrassingly parallel since they take little to no effort to separate the overall task into independent sub-tasks (Prabhu, 2007). The intrinsic parallelism in CNNs enables accelerating training and reducing inference time—this justifies the common use of GPUs and FPGAs for training and testing.

Additional common practices can be observed by tracking the pattern ML-MHD choices among the surveyed articles. As evident in Table 1, ML techniques are utilized mainly for the accuracy they attain. Despite the importance of choosing an ML technique with favorable performance, robustness, and a complexity that suits its target ITS application, authors of have commonly focused on accuracy for the evaluation of their models. Furthermore, the selection of MHDs was mainly influenced by their performance, as in processing capabilities, and the wealth of available computational resources within devices. Nevertheless, for certain applications, other factors were considered of significant importance, namely power consumption and flexibility. In fact, it is evident from Table 1 that most of the investigations that cited flexibility as one of the prime hardware device choice factors chose FPGAs for their intrinsic high degree of parallelism and their famous physical hardware reconfigurability.

5.3. Limitations and Pointers to Future Directions

Despite the significant progress in the literature, there exist several gaps and limitations in the investigated work. First, the various object and traffic detection problems tackled in investigations such as (Xie et al., 2018; Gkolias and Vlahogianni, 2019; Shvai et al., 2020; Deng et al., 2020) do not account for challenging conditions such as harsh lighting, harsh weather, low resolution, and more. In addition, various promising ML techniques, namely reinforcement learning techniques, and MHDs, like Tensor Processing Units (TPUs) and Coarse Grain Reconfigurable Arrays, have been underutilized for ITS applications in the literature. Little to no work using these techniques and devices has been reported despite their promising potential. Similarly, AG FPGAs, which exhibit high competence in Autonomous Driving and Advanced Driver Assistance Systems applications, are not exploited in research. FPGAs, such as Intel Cyclone 10 LP and Xilinx Artix-7 are sample devices that offer defense-grade security and promising performance while maintaining low power consumption. To that end, further consideration of such a type of devices is recommended. On the other hand, one observed limitation is that many investigations, such as (Shvai et al., 2020; Deng et al., 2020; Huang et al., 2020; Wang and Zhou, 2018; Ke et al., 2018; Fang et al., 2016), suffer from small datasets or from the lack of diverse datasets. This can hinder the process of achieving high accuracy or generalizing to unforeseen data. Additionally, the limited size of FPGAs does not allow for the implementation of resource-demanding algorithms. However, the expected introduction of dense FPGAs in the future will facilitate the implementation of computationally intensive applications.

Although certain investigations have adopted some of the best practices that guide future research, the numerous gaps and limitations observed pave the way for a wider range of future directions. From the surveyed articles, recommended pointers to future work are combined into the generalized categories presented in Table 11. These broad future work recommendations are classified into optimization, modification, or extended utilization of current approaches. Optimization aims to improve the performance of the existing approach without any major changes to the main MHD or ML technique to enhance efficiency (Tian et al., 2019; Zhu et al., 2017), accuracy (Lyu et al., 2018; Du et al., 2020; Lee and Kim, 2018; Zhu et al., 2017), scale (Zhan et al., 2018), or any of the performance indicators. This is accomplished by integrating multimedia processing, pre-processing, or post-processing techniques that improve the quality of initial data and obtained results. Another way is by diversifying the training data through different datasets (Wang et al., 2019; Soon et al., 2018; Wont et al., 2018; Zheng et al., 2019; Jin et al., 2014; Bulan et al., 2017), data augmentation (Shvai et al., 2020), or data synthesis to train the model for more challenging situations. Furthermore, other ML parameters may be considered for training such as various traffic conditions (Deng et al., 2020; Wang et al., 2019; Du et al., 2020; Zheng et al., 2019), vehicle parameters, or road features Shi et al. (2016) to achieve more inclusive results. In addition to these methods, it is recommended to develop performance evaluation frameworks to assess the overall performance for different scenarios and compare the abilities of different MHDs and ML techniques.

The surveyed literature also presents opportunities for modifying and extending the usage of the current approaches. Modification implies developing other hardware implementations or deploying different AI techniques. While FPGA- (Jing et al., 2017; Hassannejad et al., 2015; Nguyen et al., 2016; Panahi and Gholampour, 2016) and ASIC- (Sangeetha and Deepa, 2017) based implementations are recommended in some cases, GPUs are more

Table 11
General pointers to future work by chronological order with emphasis on first appearance in the surveyed literature

Reference	Year	Integrate Multimedia Processing	Integrate AI Techniques	Develop Evaluation Frameworks	Perform Testing in Real Scenarios	Develop Embedded Hardware	Consider Other ML Parameters	Reuse Contribution in Other Apps.	Diversify Training Data	Improve Performance
Chen et al. (2013)		✓								
Hsiao et al. (2014)			✓							
Hu et al. (2014)				✓						
Hu et al. (2014)			✓							
Giesemann et al. (2014)			✓							
Chen et al. (2014)		✓								
Jin et al. (2014)	2014				✓					
Magrini et al. (2015)				✓						
Alonso et al. (2015)			✓							
Hsiao et al. (2015)			✓							
Hassannejad et al. (2015)						✓				
de Paula and Jung (2015)							✓			
Hu et al. (2015)								✓		
Kosaka and Ohnishi (2015)	2015				✓					
Chen and Huang (2016)					✓					
Parsabi and Gholampour (2016)						✓				
Lee et al. (2016)			✓							
Khan et al. (2016)			✓							
Khan et al. (2016)						✓				
Hsiao et al. (2016)			✓							
Shi et al. (2016)					✓		✓			
Shi et al. (2015)	2016				✓					
Sangeetha and Deepa (2017)						✓				
Vasconcelos et al. (2017)					✓					
Han et al. (2017)				✓						
Jing et al. (2017)	2017			✓						
Chen et al. (2017)		✓								
Fang et al. (2016)										
Chen et al. (2017)							✓			
Bulan et al. (2017)									✓	
Koang et al. (2016)									✓	
Liu et al. (2016)									✓	
Nguyen et al. (2016)						✓	✓	✓		
Yuan et al. (2016)	2017						✓			
Dairi et al. (2018)						✓				
Cecilia et al. (2018)							✓			
Dey et al. (2018)							✓			
Vu et al. (2017)							✓			
Lyu et al. (2018)										
Weng and Chiu (2018)							✓			
Soon et al. (2018)							✓			
Lee and Min (2018)							✓			
Wont et al. (2018)							✓			
Zhan et al. (2018)										✓
Wang et al. (2017)							✓			
Lee and Kim (2018)							✓			
Chung and Sohn (2017)									✓	
Luo et al. (2017)							✓	✓		
Xie et al. (2018)							✓		✓	✓
Wang et al. (2017)							✓			
Zhu et al. (2017)	2018	✓					✓			
Tian et al. (2019)	2019						✓			✓
Gudigar et al. (2019)										
Wang et al. (2019)									✓	
Cabanes et al. (2019)										
Ozkay et al. (2019)									✓	
Wu et al. (2019)										
Chen et al. (2019)										
Heredia and Barros-Gavilanes (2019)										
Königshof et al. (2019)										
Li et al. (2019)										
Gkolias and Vlahogianni (2019)										
Mhalla et al. (2018)										
Zeng et al. (2018)										
Zheng et al. (2019)										
Yu and Gu (2019)										
Li et al. (2018)										
Ke et al. (2018)										
Wang and Zhou (2018)										
Singh and Mohan (2018)										
Ke et al. (2018)										
Sanz-Madoz et al. (2019)	2019									
Deng et al. (2020)										
Shvai et al. (2020)										
Camposverde and Barros (2019)										
Abughalieh and Alawneh (2020)										
Du et al. (2020)										
Huang et al. (2020)	2020									

suitable for others (Han et al., 2017; Abughalieh and Alawneh, 2020). Integration of different AI techniques encompasses deploying DL techniques, such as CNNs, and comparing them with chosen ML algorithms, or adopting more advanced algorithms for a certain application in order to perform more adequately in terms of speed, accuracy, power consumption, and multi-parameter capacity. Some investigations also present promising results that suggest the prospect of deploying them for different applications within as well as outside the scope of ITS. Within the scope of ITS, it is recommended to test an approach for real scenarios and assess the capabilities of the system under different operating conditions or data types with additional goals like detecting other objects or characteristics than those initially sought as in (Wont et al., 2018; Lee and Kim, 2018; Nguyen et al., 2016). Indeed, Ann Arbor Connected Vehicle Test Environment (AACVTE) is a prime example of a real-world deployment testing environment that is considered to be a step towards the national deployment of connected vehicles and infrastructure in the United States. Additionally, a contribution may be utilized for non-ITS applications relating to AI problems in industry and research (Yu and Gu, 2019). Notably, two of the most frequent pointers to future work are those concerned with the chosen ML techniques and hardware used for implementation, both of which are included in the previously presented cross-matching chart. Finally, these future directions present numerous opportunities for researchers to tackle MHD challenges for ML implementations within the scope of ITS while considering the evaluation framework developed in this work.

6. Conclusion

The expected amount of data to be produced by ITS in the era of Big Data has stimulated a notable research interest in applying ML techniques to operate certain applications. With its ability to process large amounts of data and find non-linear relationships in complex systems, ML has caused a wave that has swept across ITS by providing viable solutions to complex problems. While a critical role is played by ML techniques in the quality of services ITS offer, a similar role is imposed on the processing devices used to run these data-driven techniques, especially with the recent development in autonomous driving and connected vehicles. In this paper, we provide a comprehensive survey on the recent literature of ITS applications where ML techniques were applied along with MHDs to run these techniques. Inspired by the surveyed literature, we propose a taxonomy in Fig. 2 that classifies the sub-fields of ITS, ML, and MHDs, in a way that confines with the survey context. To solve the non-trivial task of selecting a suitable ML technique and MHD for an ITS application with a specific complexity, we develop a performance evaluation framework of which the cross-matching evaluation chart is illustrated in Fig. 3. The proposed framework provides a recommendation of what competency of ML technique and MHD to select for an ITS application, of which the complexity can be identified from the proposed performance levels. Upon validation, our framework proved adequately effective with only a few small differences in classification out of a pool of references, and only two significant differences between the framework's recommendation and authors' choices.

Despite the extensive efforts in the literature, we identified several gaps and limitations that could guide future research avenues. From an ML perspective, additional efforts should be directed at improving the robustness of models that will be used for real world deployment. Most of the surveyed ITS applications that rely on visual data would benefit from this as they are mostly trained on datasets that don't fully represent real-world challenges. Hence, investigating techniques such as domain adaptation could be useful to adapt models to real environments. Similarly, in the realm of hardware, MHDs such as TPUs, AG FPGAs, or others, were not used in the literature despite their promising characteristics. Investigating the potential of such devices (Shreejith and Fahmy, 2014; Microchip, 2021; Microchip, dspic33ch, 2020; NVIDIAAGX developer kit, 2021; Xilinx, 2021) could indeed lead improvements to attain the desired performance for real-world deployment.

Our future directions set a road map for bridging the gap between theoretical research and real-world deployment of ITS through several steps. Firstly, future investigations can make more use of recent advancements in ML research (Wu et al., 2019; Goodfellow et al., 2014; Ivanov and D'yakonov, 2019; Vaswani et al., 2017; Yosinski et al., 2014). Moreover, more focus can be directed towards developing and testing suitable MHDs to satisfy deployment requirements in real scenarios. Diversifying training procedures can enhance the robustness of ML models at inference against issues and conditions present in the real world. We expect our proposed framework to ease the selection of suitable ML techniques and MHDs in a manner that promises high performance. We aim to extend our work by creating a recommender system inspired by the developed framework.

The proposed cross-matching model can be generalized into a pattern that comprises an application area, a problem-solving technique, and implementation technology options. The proposed pattern inspires the development of similar cross-matching models for a variety of contexts. Work in progress comprises the development of a model that aids the interweaving of optimization of vehicular resources within a smart city, bio-inspired computing techniques, and MHDs.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abughalieh, K.M., Alawneh, S.G., 2020. Predicting pedestrian intention to cross the road. IEEE Access 8, 72558–72569.

J. Alonso, J. López, I. Pavón, C. Asensio, G. Areas, Platform for on-board real-time detection of wet, icy and snowy roads, using tyre/road noise analysis, in: 2015 International Symposium on Consumer Electronics (ISCE), IEEE, 2015, pp. 1–2.

An, S., Lee, B., Shin, D., 2011. A survey of Intelligent Transportation Systems. In: 2011 Third International Conference on Computational Intelligence, Communication Systems and Networks, pp. 332–337.

Arnold, E., Al-Jarrah, O.Y., Dianati, M., Fallah, S., Oxtoby, D., Mouzakitis, A., 2019. A survey on 3d object detection methods for autonomous driving applications. IEEE Trans. Intell. Transp. Syst. 20 (10), 3782–3795.

Asmara, R.A., Syahputro, B., Supriyanto, D., Handayani, A.N., 2020. Prediction of traffic density using yolo object detection and implemented in raspberry pi 3b + and intel ncs 2. In: 2020 4th International Conference on Vocational Education and Training (ICOVET), pp. 391–395. https://doi.org/10.1109/ICOVET50258.2020.9230145.

Bulan, O., Kozitsky, V., Ramesh, P., Shreve, M., 2017. Segmentation-and annotation-free license plate recognition with deep localization and failure identification. IEEE Trans. Intell. Transp. Syst. 18 (9), 2351–2363.

Buy a raspberry pi zero w – raspberry pi, url:https://www.raspberrypi.org/products/raspberry-pi-zero-w/, (Accessed on 07/27/2020).

Cabanes, Q., Senouci, B., Ramdane-Cherif, A., 2019. A complete multi-CPU/FPGA-based design and prototyping methodology for autonomous vehicles: Multiple object detection and recognition case study. In: 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC). IEEE, pp. 158–163.

Campoverde, A., Barros, G., 2019. Detection and classification of urban actors through tensorflow with an android device. In: Conference on Information Technologies and Communication of Ecuador. Springer, pp. 167–181.

Cecilia, J.M., Timón, I., Soto, J., Santa, J., Pereñíguez, F., Muñoz, A., 2018. High-throughput infrastructure for advanced its services: A case study on air pollution monitoring. IEEE Trans. Intell. Transp. Syst. 19 (7), 2246–2257.

Chen, Z., Huang, X., 2016. Accurate and reliable detection of traffic lights using multiclass learning and multiobject tracking. IEEE Intell. Transp. Syst. Mag. 8 (4), 28–42.

Chen, P.-Y., Huang, C.-C., Lien, C.-Y., Tsai, Y.-H., 2013. An efficient hardware implementation of hog feature extraction for human detection. IEEE Trans. Intell. Transp. Syst. 15 (2), 656–662.
Z. Chen, X. Huang, Z. Ni, H. He, A GPU-based real-time traffic sign detection and recognition system, in: 2014 IEEE symposium on computational intelligence in vehicles and transportation systems (CIVTS), IEEE, 2014, pp. 1–5.
Chen, W., Guo, F., Wang, F.-Y., 2015. A survey of traffic data visualization. IEEE Trans. Intell. Transp. Syst. 16 (6), 2970–2984.
Chen, L., Fan, L., Xie, G., Huang, K., Nüchter, A., 2017. Moving-object detection from consecutive stereo pairs using slanted plane smoothing. IEEE Trans. Intell. Transp. Syst. 18 (11), 3093–3102.
Chen, L., Hu, X., Xu, T., Kuang, H., Li, Q., 2017. Turn signal detection during nighttime by CNN detector and perceptual hashing tracking. IEEE Trans. Intell. Transp. Syst. 18 (12), 3303–3314.
L. Chen, Q. Zou, Z. Pan, D. Lai, L. Zhu, Z. Hou, J. Wang, D. Cao, Surrounding vehicle detection using an FPGA panoramic camera and deep CNNs, IEEE Transactions on Intelligent Transportation Systems.
Chung, J., Sohn, K., 2017. Image-based learning to measure traffic density using a deep convolutional neural network. IEEE Trans. Intell. Transp. Syst. 19 (5), 1670–1675.
Dairi, A., Harrou, F., Sun, Y., Senouci, M., 2018. Obstacle detection for intelligent transportation systems using deep stacked autoencoder and *k*-nearest neighbor scheme. IEEE Sens. J. 18 (12), 5122–5132.
Damaj, I.W., 2007. Higher-level hardware synthesis of the kasumi algorithm. J. Computer Sci. Technol. 22 (1), 60–70.
Damaj, I., 2016. A unified analysis approach for hardware and software implementations. In: 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1–4. https://doi.org/10.1109/MWSCAS.2016.7870083.
Damaj, I., Kasbah, S., 2018. An analysis framework for hardware and software implementations with applications from cryptography. Comput. Electr. Eng. 69, 572–584. https://doi.org/10.1016/j.compeleceng.2017.06.008. URL: http://www.sciencedirect.com/science/article/pii/S0045790617315653.
Damaj, I.W., El Hajj, A.M., Mouftah, H.T., 2019. An analytical framework for effective joint scheduling over TDD-based mobile networks. IEEE Access 7, 144214–144229.
Damaj, I., Elshafei, M., El-Abd, M., Aydin, M.E., 2020. An analytical framework for high-speed hardware particle swarm optimization. Microprocess. Microsyst. 72,. https://doi.org/10.1016/j.micpro.2019.102949. URL:http://www.sciencedirect.com/science/article/pii/S0141933119300407 102949.
I.W. Damaj, W.E. Mardini, H.T. Mouftah, A mathematical framework for effective routing over low power and lossy networks, International Journal of Communication Systems 33 (11) (2020) e4416, doi:10.1002/dac.4416. url: https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.4416.
Damaj, I.W., Serhal, D.K., Hamandi, L.A., Zantout, R.N., Mouftah, H.T., 2021. Connected and autonomous electric vehicles: Quality of experience survey and taxonomy. Vehicular Communication 28 In this issue 100312. https://doi.org/10.1016/j.vehcom.2020.100312.
Datondji, S.R.E., Dupuis, Y., Subirats, P., Vasseur, P., 2016. A survey of vision-based traffic monitoring of road intersections. IEEE Trans. Intelligent Transp. Systems 17 (10), 2681–2698.
T. Deng, H. Yan, L. Qin, T. Ngo, B. Manjunath, How do drivers allocate their potential attention? driving fixation prediction via convolutional neural networks, IEEE Transactions on Intelligent Transportation Systems.
de Paula, M.B., Jung, C.R., 2015. Automatic detection and classification of road lane markings using onboard vehicular cameras. IEEE Trans. Intell. Transp. Syst. 16 (6), 3160–3169.
S. Dey, G. Kalliatakis, S. Saha, A.K. Singh, S. Ehsan, K. McDonald-Maier, Mat-CNN-sopc: Motionless analysis of traffic using convolutional neural networks on system-on-a-programmable-chip, in: 2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), IEEE, 2018, pp. 291–298..
B. Du, H. Peng, S. Wang, M.Z.A. Bhuiyan, L. Wang, Q. Gong, L. Liu, J. Li, Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction, IEEE Transactions on Intelligent Transportation Systems.
Fang, J., Zhou, Y., Yu, Y., Du, S., 2016. Fine-grained vehicle model recognition using a coarse-to-fine convolutional neural network architecture. IEEE Trans. Intell. Transp. Syst. 18 (7), 1782–1792.
GeForce GT 1030 — specifications — GeForce, url:https://www.geforce.com/hardware/desktop-gpus/geforce-gt-1030/specifications, (Accessed on 07/27/2020).
GeForce RTX 2080 TI graphics card — NVIDIA, url:https://www.nvidia.com/en-me/geforce/graphics-cards/rtx-2080-ti/, (Accessed on 07/27/2020).
F. Giesemann, G. Payá-Vayá, H. Blume, M. Limmer, W. Ritter, A comprehensive ASIC/FPGA prototyping environment for exploring embedded processing systems for advanced driver assistance applications, in: 2014 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV), IEEE, 2014, pp. 314–321.
K. Gkolias, E.I. Vlahogianni, Convolutional neural networks for on-street parking space detection in urban networks, IEEE Transactions on Intelligent Transportation Systems.
I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in neural information processing systems, 2014, pp. 2672–2680.
C. Goutte, E. Gaussier, A probabilistic interpretation of precision, recall and f-score, with implication for evaluation, in: European conference on information retrieval, Springer, 2005, pp. 345–359.
Greathouse, J.L., Loh, G.H., 2018. Machine learning for performance and power modeling of heterogeneous systems, in. In: 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–6.
Gudigar, A., Chokkadi, S., Raghavendra, U., Acharya, U.R., 2019. An efficient traffic sign recognition based on graph embedding features. Neural Comput. Appl. 31 (2), 395–407.
Guerrero-Ibáñez, J., Zeadally, S., Contreras-Castillo, J., 2018. Sensor technologies for intelligent transportation systems. Sensors 18 (4), 1212.
Y. Han, E. Oruklu, Traffic sign recognition based on the NVIDIA JETSON TX1 embedded system using convolutional neural networks, in: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), IEEE, 2017, pp. 184–187.
Hassannejad, H., Medici, P., Cardarelli, E., Cerri, P., 2015. Detection of moving objects in roundabouts based on a monocular system. Expert Syst. Appl. 42 (9), 4167–4176.
A. Heredia, G. Barros-Gavilanes, Svm and ssd for classification of mobility actors on the edge, in: IEEE Colombian Conference on Applications in Computational Intelligence, Springer, 2019, pp. 3–15.
Hsiao, S.-F., Yeh, G.-F., Chen, J.-C., 2014. Design and implementation of multiple-vehicle detection and tracking systems with machine learning. In: 2014 17th Euromicro Conference on Digital System Design. IEEE, pp. 551–558.
Hsiao, P.-Y., Lin, S.-Y., Huang, S.-S., 2015. An FPGA based human detection system with embedded platform. Microelectron. Eng. 138, 42–46.
P.-Y. Hsiao, S.-Y. Lin, C.-Y. Chen, A real-time FPGA based human detector, in: 2016 International Symposium on Computer, Consumer and Control (IS3C), IEEE, 2016, pp. 1014–1017.
Y. Huang, Z. Liu, M. Jiang, X. Yu, X. Ding, Cost-effective vehicle type recognition in surveillance images with deep active learning and web data, IEEE Transactions on Intelligent Transportation Systems.
Hu, B., Li, Y., Chen, Z., Xiong, G., Zhu, F., 2014. Research on abandoned and removed objects detection based on embedded system. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 2968–2971.
Hu, B., Xiong, G., Li, Y., Chen, Z., Zhou, W., Wang, X., Wang, Q., 2014. Research on passenger flow counting based on embedded system. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 3116–3119.
Hu, C., Bai, X., Qi, L., Wang, X., Xue, G., Mei, L., 2015. Learning discriminative pattern for real-time car brand recognition. IEEE Trans. Intell. Transp. Syst. 16 (6), 3170–3181.
Intel FPGA Product Catalog, 2021. url:https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/sg/product-catalog.pdf, (Accessed on 07/27/2020).
Jensen, M.B., Philipsen, M.P., Møgelmose, A., Moeslund, T.B., Trivedi, M.M., 2016. Vision for looking at traffic lights: Issues, survey, and perspectives. IEEE Trans. Intell. Transp. Syst. 17 (7), 1800–1815.
Jin, J., Fu, K., Zhang, C., 2014. Traffic sign recognition with hinge loss trained convolutional neural networks. IEEE Trans. Intell. Transp. Syst. 15 (5), 1991–2000.
Jing, Y., Youssefi, B., Mirhassani, M., Muscedere, R., 2017. An efficient FPGA implementation of optical character recognition for license plate recognition. In: 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE). IEEE, pp. 1–4.
S. Ivanov, A. D'yakonov, 2019. Modern deep reinforcement learning algorithms, arXiv preprint arXiv:1906.10025.
I. Kaparias, M.G.H. Bell, A. Gal-Tzur, O.M. Komar, C.G. Prato, L. Tartakovsky, B. Aronov, Y. Zvirin, M. Gerstenberger, A. Tsakarestos, S. Nocera, Key performance indicators for traffic management and intelligent transport systems deliverable, in: CONDUITS Deliverable 3.5, 2011.
Ke, R., Li, Z., Tang, J., Pan, Z., Wang, Y., 2018. Real-time traffic flow parameter estimation from uav video based on ensemble classifier and optical flow. IEEE Trans. Intell. Transp. Syst. 20 (1), 54–64.
Ke, X., Shi, L., Guo, W., Chen, D., 2018. Multi-dimensional traffic congestion detection based on fusion of visual features and convolutional neural network. IEEE Trans. Intell. Transp. Syst. 20 (6), 2157–2170.
A. Khan, C.-M. Kyung, Memory efficient hardware accelerator for kernel support vector machine based pedestrian detection, in: 2016 International SoC Design Conference (ISOCC), IEEE, 2016, pp. 127–128.
A. Khan, C.-M. Kyung, Memory efficient hardware accelerator for kernel support vector machine based pedestrian detection, in: 2016 International SoC Design Conference (ISOCC), IEEE, 2016, pp. 127–128.
Klaine, P.V., Imran, M.A., Onireti, O., Souza, R.D., 2017. A survey of machine learning techniques applied to self-organizing cellular networks. IEEE Commun. Surveys Tutorials 19 (4), 2392–2431.
Königshof, H., Salscheider, N.O., Stiller, C., 2019. Realtime 3D object detection for automated driving using stereo vision and semantic information, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 1405–1410.
Kosaka, N., Ohashi, G., 2015. Vision-based nighttime vehicle detection using censure and svm. IEEE Trans. Intell. Transp. Syst. 16 (5), 2599–2608.
Kuang, H., Zhang, X., Li, Y.-J., Chan, L.L.H., Yan, H., 2016. Nighttime vehicle detection based on bio-inspired image enhancement and weighted score-level feature fusion. IEEE Trans. Intell. Transp. Syst. 18 (4), 927–936.
Kuutti, S., Bowden, R., Jin, Y., Barber, P., Fallah, S., 2021. A survey of deep learning applications to autonomous vehicle control. IEEE Transactions on Intelligent Transportation Systems.

Lee, H.S., Kim, K., 2018. Simultaneous traffic sign detection and boundary estimation using convolutional neural network. IEEE Trans. Intell. Transp. Syst. 19 (5), 1652–1663.
Lee, Y.-J., Min, O., 2018. Long short-term memory recurrent neural network for urban traffic prediction: a case study of seoul. In: in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, pp. 1279–1284.
E. Lee, S.-S. Lee, Y. Hwang, S.-J. Jang, Hardware implementation of fast traffic sign recognition for intelligent vehicle system, in: 2016 International SoC Design Conference (ISOCC), IEEE, 2016, pp. 161–162.
Li, C., Chen, Z., Wu, Q.J., Liu, C., 2018. Deep saliency with channel-wise hierarchical feature responses for traffic sign detection. IEEE Trans. Intell. Transp. Syst. 20 (7), 2497–2509.
Li, X., Li, J., Hu, X., Yang, J., 2019. Line-CNN: End-to-end traffic line detection with line proposal unit. IEEE Trans. Intell. Transp. Syst. 21 (1), 248–258.
Lu, X., Wang, Y., Zhou, X., Zhang, Z., Ling, Z., 2016. Traffic sign recognition via multi-modal tree-structure embedded multi-task learning. IEEE Trans. Intell. Transp. Syst. 18 (4), 960–972.
Luo, H., Yang, Y., Tong, B., Wu, F., Fan, B., 2017. Traffic sign recognition using a multi-task convolutional neural network. IEEE Trans. Intell. Transp. Syst. 19 (4), 1100–1111.
Y. Lyu, L. Bai, X. Huang, Real-time road segmentation using lidar data processing on an FPGA, in: 2018 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2018, pp. 1–5.
Magrini, M., Moroni, D., Palazzese, G., Pieri, G., Leone, G., Salvetti, O., 2015. Computer vision on embedded sensors for traffic flow monitoring. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. IEEE, pp. 161–166.
Malone, C., Zahran, M., Karri, R., 2011. Are hardware performance counters a cost effective way for integrity checking of programs. In: Proceedings of the sixth ACM workshop on Scalable trusted computing, pp. 71–76.
Martinez, C.M., Heucke, M., Wang, F.-Y., Gao, B., Cao, D., 2017. Driving style recognition for intelligent vehicle control and advanced driver assistance: A survey. IEEE Trans. Intell. Transp. Syst. 19 (3), 666–676.
Martinez, C.M., Heucke, M., Wang, F.-Y., Gao, B., Cao, D., 2017. Driving style recognition for intelligent vehicle control and advanced driver assistance: A survey. IEEE Trans. Intell. Transp. Syst. 19 (3), 666–676.
Meneguette, R.I., Grande, R.E.D., Loureiro, A.A.F., 2018. Intelligent Transport System in Smart Cities. Springer International Publishing. https://doi.org/10.1007/978-3-319-93332-0.
Meneguette, R.I., Grande, R.E.D., Loureiro, A.A.F., 2018. Intelligent Transport System in Smart Cities. Springer International Publishing. https://doi.org/10.1007/978-3-319-93332-0. url:https://doi.org/10.1007/978-3-319-93332-0.
Mhalla, A., Chateau, T., Gazzah, S., Amara, N.E.B., 2018. An embedded computer-vision system for multi-object detection in traffic surveillance. IEEE Trans. Intell. Transp. Syst. 20 (11), 4006–4018.
Microchip, Cryptoautomotive, 2021. url:https://www.microchip.com/design-centers/security-ics/cryptoautomotive, (Accessed on 08/07/2022).
Microchip, dspic33ch — microchip technology, url:https://www.microchip.com/design-centers/16-bit/products/dspic33c/dspic33ch, (Accessed on 02/01/2020)..
D.M. Manias, A. Shami, Making a case for federated learning in the internet of vehicles and intelligent transportation systems (2021). arXiv:2102.10142.
Mohri, M., Rostamizadeh, A., Talwalkar, A., 2018. Foundations of machine learning. The MIT Press, Cambridge, Massachusetts.
Nguyen, V.D., Van Nguyen, H., Tran, D.T., Lee, S.J., Jeon, J.W., 2016. Learning framework for robust obstacle detection, recognition, and tracking. IEEE Trans. Intell. Transp. Syst. 18 (6), 1633–1646.
Outay, F., Taha, B., Chaabani, H., Kamoun, F., Werghi, N., et al., 2019. Estimating ambient visibility in the presence of fog: a deep convolutional neural network approach. Pers. Ubiquit. Comput., 1–12
Panahi, R., Gholampour, I., 2016. Accurate detection and recognition of dirty vehicle plate numbers for high-speed applications. IEEE Trans. Intelligent Transp. Systems 18 (4), 767–779.
Prabhu, R.D., 2007. Gneuron: parallel neural networks with gpu. In: Proc. of International Conference on High Performance Computing, Citeseer.
NVIDIA DRIVE AGXdeveloper kit, 2021. url:https://developer.nvidia.com/drive/drive-agx, (Accessed on 07/08/2021).
Rouabeh, H., Abdelmoula, C., Masmoudi, M., 2017. VHDL design for a speed limit sign segmentation algorithm. IADIS International Journal on Computer Science & Information Systems 12 (2).
RyguŁlla, A., Baran, D., 2019. Assessment and classification of selected its in poland. In: 2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), pp. 1–8.
Sangeetha, D., Deepa, P., 2017. A low-cost and high-performance architecture for robust human detection using histogram of edge oriented gradients. Microprocess. Microsyst. 53, 106–119.
E. Sanz-Madoz, J. Echanobe, O. Mata-Carballeira, I. del Campo, M. Martínez, Reduced kernel extreme learning machine for traffic sign recognition, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 4101–4106.
Ser, J.D., Sanchez-Medina, J.J., Vlahogianni, E.I., 2019. Introduction to the special issue on online learning for big-data driven transportation and mobility. IEEE Trans. Intell. Transp. Syst. 20 (12), 4621–4623. https://doi.org/10.1109/tits.2019.2955548. url:https://doi.org/10.1109/tits.2019.2955548.
Shi, Z., Zou, Z., Zhang, C., 2015. Real-time traffic light detection with adaptive background suppression filter. IEEE Trans. Intell. Transp. Syst. 17 (3), 690–700.
Shi, Y., Cui, L., Qi, Z., Meng, F., Chen, Z., 2016. Automatic road crack detection using random structured forests. IEEE Trans. Intell. Transp. Syst. 17 (12), 3434–3445.
Shi, W., Alawieh, M.B., Li, X., Yu, H., 2017. Algorithm and hardware implementation for visual perception system in autonomous vehicle: A survey. Integration 59, 148–156.
Shreejith, S., Fahmy, S.A., 2014. Extensible flexray communication controller for FPGA-based automotive systems. IEEE Trans. Vehicular Technol. 64 (2), 453–465.
Raspberry Pi, 2021. url:https://www.raspberrypi.org/products/raspberry-pi-4-model-b/, (Accessed on 08/07/2021).
Shvai, N., Hasnat, A., Meicler, A., Nakib, A., 2020. Accurate classification for automatic vehicle-type recognition based on ensemble classifiers. IEEE Transactions on Intelligent Transportation Systems.
Singh, D., Mohan, C.K., 2018. Deep spatio-temporal representation for detection of road accidents using stacked autoencoder. IEEE Trans. Intell. Transp. Syst. 20 (3), 879–887.
Sistu, G., Leang, I., Yogamani, S., 2019. Real-time joint object detection and semantic segmentation network for automated driving. arXiv:1901.03912.
Soon, F.C., Khaw, H.Y., Chuah, J.H., Kanesan, J., 2018. Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition. IET Intel. Transport Syst. 12 (8), 939–946.
Sze, V., Chen, Y.-H., Emer, J., Suleiman, A., Zhang, Z., 2017. Hardware for machine learning: Challenges and opportunities, in: 2017 IEEE Custom Integrated Circuits Conference (CICC), IEEE, 2017, pp. 1–8.
Talib, M.A., Majzoub, S., Nasir, Q., Jamal, D., XXXX. A systematic literature review on hardware implementation of artificial intelligence algorithms, J. Supercomputing.
Y. Tian, J. Gelernter, X. Wang, J. Li, Y. Yu, Traffic sign detection using a multi-scale recurrent attention network, IEEE Transactions on Intelligent Transportation Systems.
Sussman, J.S., 2005. Perspectives on Intelligent Transportation Systems (ITS). Springer Science & Business Media. https://doi.org/10.1007/b101063.
Vasconcelos, I., Vasconcelos, R.O., Olivieri, B., Roriz, M., Endler, M., Junior, M.C., 2017. Smartphone-based outlier detection: a complex event processing approach for driving behavior detection. J. Internet Services Appl. 8 (1), 13.
A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, ŁL. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008..
Velez, G., Cortés, A., Nieto, M., Vélez, I., Otaegui, O., 2015. A reconfigurable embedded vision system for advanced driver assistance. J. Real-Time Image Proc. 10 (4), 725–739.
Veres, M., Moussa, M., 2019. Deep learning for intelligent transportation systems: A survey of emerging trends. IEEE Trans. Intelligent Transportation Syst.
Vinh, T.Q., 2015. Real-time traffic sign detection and recognition system based on friendlyarm tiny4412 board. In: 2015 International Conference on Communications, Management and Telecommunications (ComManTel), pp. 142–146. https://doi.org/10.1109/ComManTel.2015.7394276.
N. Vu, C. Pham, Traffic incident recognition using empirical deep convolutional neural networks model, in: Context-Aware Systems and Applications, and Nature of Computation and Communication, Springer, 2017, pp. 90–99.
Wang, J.-G., Zhou, L.-B., 2018. Traffic light recognition with high dynamic range imaging and deep learning. IEEE Trans. Intell. Transp. Syst. 20 (4), 1341–1352.
Wang, Q., Gao, J., Yuan, Y., 2017. Embedding structured contour and location prior in siamesed fully convolutional networks for road detection. IEEE Trans. Intell. Transp. Syst. 19 (1), 230–241.
Wang, J., Zheng, H., Huang, Y., Ding, X., 2017. Vehicle type recognition in surveillance images from labeled web-nature data using deep transfer learning. IEEE Trans. Intell. Transp. Syst. 19 (9), 2913–2922.
Wang, X., Zhang, W., Wu, X., Xiao, L., Qian, Y., Fang, Z., 2019. Real-time vehicle type classification with deep convolutional neural networks. J. Real-Time Image Proc. 16 (1), 5–14.
Weng, H.-M., Chiu, C.-T., 2018. Resource efficient hardware implementation for real-time traffic sign recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 1120–1124.
W.-J. Wont, T.H. Kim, M.-K. Choi, S. Kwon, AggNet: Simple aggregated network for real-time multiple object detection in road driving scene, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 3505–3510.
Wu, X.-H., Hu, R., Bao, Y.-Q., 2019. Parallelism optimized architecture on FPGA for real-time traffic light detection. IEEE Access 7, 178167–178176.
Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y., 2019. A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems.
Xie, L., Ahmad, T., Jin, L., Liu, Y., Zhang, S., 2018. A new CNN-based method for multi-directional car license plate detection. IEEE Trans. Intell. Transp. Syst. 19 (2), 507–517.
Xu, H., Mannor, S., 2011. Robustness and generalization. Machine Learning 86 (3), 391–423. https://doi.org/10.1007/s10994-011-5268-1.
Yalla, V.G., Pillai, P.J., Oguchi, K., 2015. Chase algorithm: ease of driving classification. In: in: 2015 IEEE 18th International Conference on Intelligent Transportation Systems, pp. 644–649. https://doi.org/10.1109/ITSC.2015.111.
Xilinx, Versal AI core series VCK190 evaluation kit, 2021. url:https://www.xilinx.com/products/boards-and-kits/vck190.html, (Accessed on 08/07/2021).
J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in neural information processing systems, 2014, pp. 3320–3328.

Yu, J.J.Q., Gu, J., 2019. Real-time traffic speed estimation with graph convolutional generative autoencoder. IEEE Trans. Intell. Transp. Syst. 20 (10), 3940–3951.
Yuan, Y., Xiong, Z., Wang, Q., 2016. An incremental framework for video-based traffic sign detection, tracking, and recognition. IEEE Trans. Intell. Transp. Syst. 18 (7), 1918–1929.
Zang, D., Ling, J., Wei, Z., Tang, K., Cheng, J., 2018. Long-term traffic speed prediction based on multiscale spatio-temporal feature learning network. IEEE Trans. Intell. Transp. Syst. 20 (10), 3700–3709.
Zappone, A., Di Renzo, M., Debbah, M., 2019. Wireless networks design in the era of deep learning: Model-based, ai-based, or both? IEEE Trans. Commun. 67 (10), 7331–7376. https://doi.org/10.1109/tcomm.2019.2924010.
Zhan, H., Gomes, G., Li, X.S., Madduri, K., Sim, A., Wu, K., 2018. Consensus ensemble system for traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 19 (12), 3903–3914.
Zhang, J., Wang, F.-Y., Wang, K., Lin, W.-H., Xu, X., Chen, C., 2011. Data-driven intelligent transportation systems: A survey. IEEE Trans. Intell. Transp. Syst. 12 (4), 1624–1639.
Zheng, Z., Yang, Y., Liu, J., Dai, H.-N., Zhang, Y., 2019. Deep and embedded learning approach for traffic flow prediction in urban informatics. IEEE Trans. Intell. Transp. Syst. 20 (10), 3927–3939.
Zhou, Y., Chen, Z., Huang, X., 2016. A system-on-chip fpga design for real-time traffic signal recognition system, in. In: 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1778–1781. https://doi.org/10.1109/ISCAS.2016.7538913.
Zhu, Y., Liao, M., Yang, M., Liu, W., 2017. Cascaded segmentation-detection networks for text-based traffic sign detection. IEEE Trans. Intelligent Transp. Systems 19 (1), 209–219.
Zhu, H., Akrout, M., Zheng, B., Pelegris, A., Jayarajan, A., Phanishayee, A., Schroeder, B., Pekhimenko, G., 2018. Benchmarking and analyzing deep neural network training, in. In: 2018 IEEE International Symposium on Workload Characterization (IISWC), pp. 88–100.
Zhu, L., Yu, F.R., Wang, Y., Ning, B., Tang, T., 2018. Big data analytics in intelligent transportation systems: A survey. IEEE Trans. Intell. Transp. Syst. 20 (1), 383–398.
7 series product tables and product selection guide, 2021. url:https://www.xilinx.com/support/documentation/selection-guides/7-series-product-selection-guide.pdf, (Accessed on 08/07/2021).