future internet

MDPI

Review

A Review of Machine Learning and IoT in Smart Transportation

Fotios Zantalis [1,†], Grigorios Koulouras [1,2,*,†], Sotiris Karabetsos [1,†] and Dionisis Kandris [3,†]

1 TelSiP Research Laboratory, Department of Electrical and Electronic Engineering, School of Engineering, University of West Attica, University Campus 2, 250 Thivon Str., Egaleo, GR-12241 Athens, Greece; fzantalis@uniwa.gr (F.Z.); sotoskar@uniwa.gr (S.K.)

2 Hellenic Telecommunications and Post Commission, 60 Kifissias Avenue, Maroussi, GR-15125 Athens, Greece

3 microSENSES Research Laboratory, Department of Electrical and Electronic Engineering, School of Engineering, University of West Attica, University Campus 2, 250 Thivon Str., Egaleo, GR-12241 Athens, Greece; dkandris@uniwa.gr

* Correspondence: gregkoul@uniwa.gr; Tel.: +30-2105381560

† These authors contributed equally to this work.

Received: 19 March 2019; Accepted: 8 April 2019; Published: 10 April 2019

Abstract: With the rise of the Internet of Things (IoT), applications have become smarter and connected devices give rise to their exploitation in all aspects of a modern city. As the volume of the collected data increases, Machine Learning (ML) techniques are applied to further enhance the intelligence and the capabilities of an application. The field of smart transportation has attracted many researchers and it has been approached with both ML and IoT techniques. In this review, smart transportation is considered to be an umbrella term that covers route optimization, parking, street lights, accident prevention/detection, road anomalies, and infrastructure applications. The purpose of this paper is to make a self-contained review of ML techniques and IoT applications in Intelligent Transportation Systems (ITS) and obtain a clear view of the trends in the aforementioned fields and spot possible coverage needs. From the reviewed articles it becomes profound that there is a possible lack of ML coverage for the Smart Lighting Systems and Smart Parking applications. Additionally, route optimization, parking, and accident/detection tend to be the most popular ITS applications among researchers.

Keywords: internet of things; machine learning; smart transportation; smart city; intelligent transportation systems; big data

1. Introduction

Over the last decade, applications based on mobile devices, sensors, and actuators have become smarter, enabling the communication among devices and the execution of more complex tasks. In 2008 the number of connected devices surpassed the global population [1] and the number keeps increasing exponentially until today. Smart phones, embedded systems, wireless sensors, and almost every electronic device are connected to a local network or the internet, leading to the era of the Internet of Things (IoT). With the number of devices increasing, the amount of data collected by those devices is increasing as well. New applications emerge that analyze the collected data to make meaningful correlations and possible decisions, leading to Artificial Intelligence (AI) via Machine Learning (ML) algorithms.

1.1. Internet of Things

In IoT terms, every connected device is considered a thing. Things usually consist of physical sensors, actuators, and an embedded system with a microprocessor. Things need to communicate with each other, creating the need for Machine-to-Machine (M2M) communication. The communication can be short-range using wireless technologies such as Wi-Fi, Bluetooth, and ZigBee, or wide-range using mobile networks such as WiMAX, LoRa, Sigfox, CAT M1, NB-IoT, GSM, GPRS, 3G, 4G, LTE, and 5G [2]. Due to the massive usage of IoT devices in all kinds of everyday life applications, it is essential to keep the cost of IoT devices low. Additionally, IoT devices should be able to handle basic tasks like the data collection, M2M communication, and even some pre-processing of the data depending on the application. Thus, it is mandatory to find a balance among cost, processing power, and energy consumption when designing or selecting an IoT device. IoT is also tightly attached to "big data", since IoT devices continuously collect and exchange a great amount of data. So, an IoT infrastructure usually implements methods to handle, store, and analyze big data [3]. It has become a common practice in IoT infrastructures, to use an IoT platform such as Kaa, Thingsboard, DeviceHive, Thingspeak, or Mainflux in order to support the M2M communication, using protocols like MQTT, AMQP, STOMP, CoAP, XMPP, and HTTP [4]. Additionally, IoT platforms offer monitoring capabilities, node management, data storing and analyzing, data driven configurable rules, etc. Depending on the application, it is sometimes essential that some data processing takes place in the IoT devices instead of some centralized node as it happens in the "cloud computing" infrastructure. So, as the processing partially moves to the end network elements, a new computing model is introduced, called "edge computing" [5]. However, since those devices are most of the times low-end devices, they may not be suitable to handle intense processing tasks. As a result, there is a need for an intermediate node, with sufficient resources, able to handle advanced processing tasks, physically located close to the end network elements, in order to minimize the overload caused by massive sending of all the data to some central cloud nodes. The solution came with the introduction of the "Fog nodes" [6]. Fog nodes help IoT devices with big data handling by providing storage, computing, and networking services. Finally, the data are stored in cloud servers, where they are available for advanced analysis using a variety of ML techniques and sharing among other devices, leading to the creation of modern added value smart applications. IoT applications have already emerged in many aspects the so called, smart city. We could group the most important applications in the following categories [7]:

- **Smart Homes:** This category includes traditional home devices, such as fridges, washing machines, or light bulbs, that have been developed and are able to communicate with each other or with authorized users via internet, offering a better monitoring and management of the devices as well as energy consumption optimization. Apart from the traditional devices, new technologies spread, providing smart home assistants, smart door locks, etc.
- **Health-care assistance:** New devices have been developed in order to improve a patient's well-being. Plasters with wireless sensors can monitor a wound's state and report the data to the doctor without the need for their physical presence. Other sensors in the form of wearable devices or small implants, can track and report a wide variety of measurements, such as heart rate, blood oxygen level, blood sugar level, or temperature.
- **Smart Transportation:** Using sensors embedded to the vehicles, or mobile devices and devices installed in the city, it is possible to offer optimized route suggestions, easy parking reservations, economic street lighting, telematics for public means of transportation, accident prevention, and autonomous driving.
- **Environmental Conditions Monitoring:** Wireless sensors distributed in the city make the perfect infrastructure for a wide variety of environmental conditions monitoring. Barometers, humidity sensors, or ultrasonic wind sensors can help to create advanced weather stations. Moreover, smart sensors can monitor the air quality and water pollution levels across the city.

- **Logistics and Supply Chain Management:** With the use of smart RFID tags, a product can be easily tracked from the production to the store, reducing cost and time significantly. In addition, smart packaging can offer features such as brand protection, quality assurance, and client personalization.
- **Security and Surveillance Systems:** Smart cameras can obtain video input across the streets. With real-time visual object recognition, smart security systems can identify suspects or prevent hazardous situations.

As discussed so far, although there is a lot to be done in terms of standardization when it comes to IoT infrastructure and technologies, Figure 1 can accurately describe the key elements of the infrastructure as they have been used in the majority of the applications.

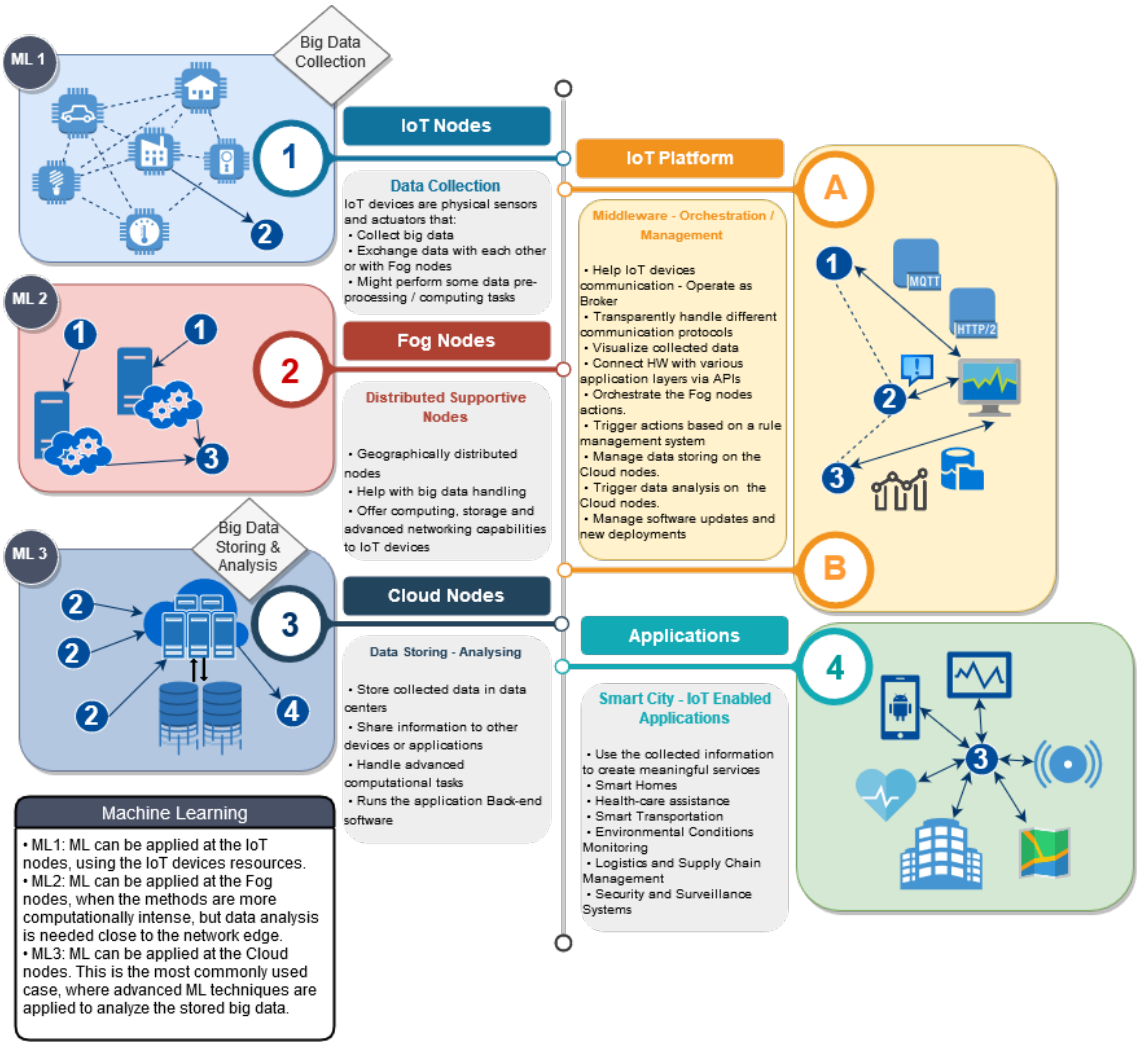

Figure 1. Key elements of the Internet of Things (IoT) infrastructure.

Figure 1 is organized by separating the infrastructure key elements in numbered blocks. Each block depicts a representative image of the described element, and arrows are linking the images with numbers, indicating how each element interconnects with the other blocks. Additionally, text blocks are included, giving the most important aspects of each key element in bullets. The IoT infrastructure consists of: (1) IoT nodes, namely the IoT devices (sensors and actuators) at the edge of the network. (2) The Fog nodes, servers that assist IoT devices by providing computing, storage, and advanced networking capabilities. (3) Cloud nodes that consist of data centers, which handle data storing, computationally intense data analysis using ML techniques, data sharing, etc. (4) IoT applications that use the collected and analyzed information to create services for the end user. Points (A) and (B)

symbolize the IoT platforms, which have the orchestrator's role and support the elements throughout the infrastructure in various ways as described in the figure. Moreover, Figure 1, presents the role of ML in the IoT infrastructure. ML techniques can be applied at the IoT nodes, the fog nodes, or the cloud nodes, depending on the application needs.

1.2. Machine Learning

Machine Learning (ML) is not a new concept. ML is closely related to Artificial Intelligence (AI). AI becomes feasible via ML. Through ML, computer systems learn to perform tasks such as classification, clustering, predictions, pattern recognition, etc. To archive the learning process, systems are trained using various algorithms and statistical models to analyze sample data. The sample data are usually characterized by measurable characteristics called features and an ML algorithm attempts to find a correlation between the features and some output values called labels [8]. Then, the information obtained during the training phase is used to identify patterns or make decisions based on new data. ML is ideal for problems such as regression, classification, clustering, and association rules determination. Depending on the learning style, ML algorithms can be grouped into four categories:

- **Supervised Learning**: Supervised learning deals with problems involving regression such as weather forecasting, estimating life experience, and population growth prediction, by using algorithms like Linear Regression or Random Forest. Additionally, supervised learning addresses classification problems such as digit recognition, speech recognition, diagnostics, and identity fraud detection, by using algorithms such as Support Vector Machines, Nearest Neighbor, Random Forest, and others. There are two phases in supervised learning. The training phase and testing phase. The data sets used for the training phase need to have known labels. The algorithms learn the relationship between the input values and labels and try to predict the output values of the testing data [9].
- **Unsupervised Learning**: Unsupervised learning deals with problems involving dimensionality reduction used for big data visualisation, feature elicitation, or the discovery of hidden structures. Moreover, supervised learning is used for clustering problems such as recommendation systems, customer segmentation, and targeted marketing. Contrary to supervised learning, in this type, no labels are available. Algorithms in this category try to identify patterns on testing data and cluster the data or predict future values [9].
- **Semi-supervised Learning**: This is a combination of the previous two categories. Both labeled data and unlabeled data are used. It works mostly like the unsupervised learning with the improvements that a portion of labeled data can bring [8].
- **Reinforcement Learning**: In this learning style, the algorithms try to predict the output for a problem based on a set of tuning parameters. Then, the calculated output becomes an input parameter and new output is calculated until the optimal output is found. Artificial Neural Networks (ANN) and Deep Learning, which will be presented later, use this learning style. Reinforcement learning is mainly used for applications like AI gaming, skill acquisition, robot navigation, and real-time decisions. [9].

When using ML techniques, there are two major parameters to consider; how computationally intense and how fast a given technique is. Depending on the application type, the most appropriate ML algorithm is chosen. If there is a need for real-time analysis for example, the chosen algorithm should be fast enough to track the changes of the input data and produce the desired output in a timely manner.

1.3. Machine Learning in IoT

With IoT, devices are connected with each other, communicate with each other and collect an enormous amount of data every day. In many applications, IoT devices may also be programmed to trigger some actions based either on some predefined conditions or on some feedback from the

collected data. However, in order to analyze the collected data and extract meaningful information and create smart applications, human intervention is required. IoT devices need not only to collect data, and communicate with other devices, but also to be autonomous. They need to be able to take context based decisions and learn from their collected data. This need led to the creation of the term "Cognitive IoT" (CIoT) [10]. Also, there is a necessity for Intelligent IoT devices, able to create automated smart applications with automated resource allocation, communication, and network operation. Deploying ML algorithms in an IoT infrastructure can introduce significant improvements in the applications or the infrastructure itself. ML can be applied for network optimization, congestion avoidance, and resource allocation optimization, but also for real-time or offline data analyzing and decision making.

Moreover, as the number of devices increases, the amount of the collected data increases as well. Having to deal with "big data" is very common in IoT applications. Big data cannot be handled properly with conventional databases. Special infrastructure is needed to handle the great volume of structured and unstructured data and special techniques to analyze them [11]. There are many ML algorithms like "Ensemble", or Artificial Neural Networks (ANN) that can help dealing efficiently with big data, and they will be discussed in the following sections.

1.4. Smart Transportation

As the IoT technology expands, new applications are created in order to make people's lives better. Cities are getting "smarter" and smart city applications are developed to take advance of the latest technological improvements. With the introduction of IoT in the field of transportation, transportation systems begin to "feel" and "think", leading to the development of Intelligent Transportation Systems (ITS). Smart transportation has attracted the attention of many researchers since there are plenty of opportunities for further enhancements. One of the most significant areas of interest in smart transportation is navigation or route optimization. Using data from the users' mobile devices [12], or with side units placed in specified locations on the road [13], applications try to estimate traffic congestion and propose optimal route options to minimize traveling times, and therefore reduce car emissions and energy consumption. Furthermore, to support the energy consumption reduction, street lights are proposed that can detect traffic conditions and operate accordingly, instead of being constantly on with a time schedule. IoT devices have been widely used to create smart parking systems, too. Using cameras [14], or other wireless sensors like magnetic field or IR sensors [15], researchers have proposed new parking reservation systems that allow to maximize a parking lot's availability and capacity and minimize the searching time. Moreover, systems that help detect road surface anomalies based on input data from sensors attached to cars or the driver's phone have been proposed. By detecting bad road conditions, accidents could possibly be avoided. There have also been efforts to detect or prevent road accidents using IoT devices. Finally, the IoT M2M communication option has given the opportunity to develop vehicle to vehicle communication and vehicle social networks, where vehicles can exchange useful information with each other and give many more possibilities for new applications [16].

The rest of the article is organized as follows. In Section 2.1, there is a presentation and a detailed analysis of the ML algorithms used in the reviewed researches to support smart transportation applications. Additionally, in Section 2.2, there is a review of the IoT and ML applications about smart transportation, organized in categories, based on the application type. In Section 3, a discussion on the schemes presented is performed, and concluding remarks are drawn.

2. Machine Learning and IoT in Smart Transportation

Smart Transportation is a very popular area of research, since it encounters many everyday problems, with a huge footprint in a modern smart city. Additionally, the nature of the problems it deals with favors the use of both IoT and ML technologies. This review aims to both identify the current trend in the use of ML and IoT in smart transportation, and examine the research coverage in

each one of the smart transportation categories. For this reason, the review focuses on the most recent research works, which address the smart transportation categories (route optimization, parking, lights, accident detection/prevention, road anomalies, and infrastructure), using IoT and/or ML techniques.

2.1. Machine Learning Algorithms

This section will focus on references that use ML algorithms to support Smart Transportation. The algorithms and their use will be thoroughly analyzed in the following subsections. Table 1 summarizes the combined findings from the reviewed literature, presenting the algorithm name, the number of times each algorithm is used in the literature, the algorithm type and the respective learning type.

Table 1. Machine Learning algorithms in Internet of Things (IoT) smart transportation applications.

No	Algorithm	References	Algorithm Type	Learning Type
1	AdaBoost	[17]	Ensemble	Supervised
2	**B**ayesian **N**etwork **S**easonal **A**utoregressive **I**ntegrated **M**oving **A**verage (BN-SARIMA)	[18]	Bayesian	Supervised
3	**C**onvolutional **N**eural **N**etwork (CNN) and **D**eep CNN (DCNN)	[19–21]	Deep Learning	Reinforcement
4	**C**oupled **H**idden **M**arkov **M**odel (CHMM)	[22]	Markov Model	Reinforcement
5	Decision Tree	[23]	Decision Trees	Supervised
6	**D**eep **B**elief **N**etworks (DBN)	[12,24]	Deep Learning	Reinforcement
7	**D**eep **R**ecurrent **A**ttention **M**odel (DRAM)	[25]	Recursive Neural Networks-Deep Learning	Reinforcement
8	**F**uzzy **C**-**M**eans (FCM)	[26]	Clustering	Unsupervised
9	**F**eed **F**orward **N**eural **N**etworks (FF-NN)	[18,23,27–30]	Artificial Neural Networks	Supervised
10	**F**ully **C**onnected **N**etworks (FCN)	[20]	Deep Learning	Reinforcement
11	**S**tacked **A**uto **E**ncoder (SAE) with Greedy Layer-wise training	[31]	Deep Learning	Reinforcement, Unsupervised
12	Inception Neural Networks	[32]	Deep Learning	Reinforcement
13	K-Means	[12,33,34]	Clustering	Unsupervised
14	*k*-**N**earest **N**eighbor (*k*-NN)	[24,29,35]	Instance Based	Supervised
15	**L**ogistic **R**egression	[23]	Regression	Supervised
16	**M**arkov **D**ecision **P**rocess (MDP)	[36]	Discrete Time Stochastic Control	Reinforcement
17	**M**arkov **R**andom **F**ield (MRF)	[14]	Markov Model	Unsupervised
18	**N**onlinear **A**uto**R**egressive e**X**ogenous model (NARX)	[18]	Recursive Neural Networks-Deep Learning	Reinforcement
19	Q-Learning	[36]	Stochastic Control-Markov Model	Reinforcement
20	**R**andom **F**orest (RF)	[23,27,34,35,37]	Ensemble	Supervised
21	Regression Tree	[27,29]	Decision Trees	Supervised
22	**S**upport-**V**ector **M**achine (SVM)	[14,23,24,35,37,38]	non-probabilistic Linear Classification	Supervised

In the following subsections, the aforementioned ML algorithms are grouped by their algorithm type and they are briefly analyzed.

2.1.1. Ensemble

Ensemble algorithms address a problem, by training multiple classifiers and combining their results. The main advantage of ensemble methods is their ability to boost the so called weak classifiers to strong classifiers. In this way, the systems can use the weak classifiers which are easily constructed and also gain the quality of a strong classifier [39]. In [17], the authors aim to prevent road accidents by creating a framework to detect the driver's consciousness. The input data are collected by a camera fixed in front of the driver, in order to track eye movement and head nodding. Then, Human HAAR features [40] are selected from an extracted integral image, which become input for the Adaptive Boosting (AdaBoost) algorithm. AdaBoost is an ensemble algorithm which is used together with other weak classifiers to form a strong classifier. In the research, a large number of features need to be processed in real time and AdaBoost shows a great advantage in that aspect. To further improve the AdaBoost results in terms of speed, the authors set the strong classifiers in a cascading order, where each classifier is fed with the features processed from the previous one. Random Forest (RF) is another popular ensemble algorithm. RF is used for regression and classification tasks, using a plethora of decision trees as weak learners, while getting their mean value or the most sampled values as an output. RF is usually less susceptible to the over-fitting effect than single decision trees. The authors in [27] developed four models, in order to forecast short-term and long-term traffic flow in work zones. They compared the methods results using the following three metrics: (a) Root Mean Square Error (RMSE), (b) Mean Absolute Error (MAE), and (c) Mean Absolute Percentage Error (MAPE). The results of this research showed that RF outperformed the implemented baseline predictor, the regression tree and a Feed Forward Neural Network in all three measurements for both long and short-term predictions. In [35], the goal is to identify road surface abnormalities. This is achieved by collecting data using accelerometer sensors with Arduino microcontrollers and then apply a set of feature selection and ML algorithms on the data. For the feature selection three techniques are used, Ranker, Greedy Algorithm, and Particle Swarm Optimisation (PSO). Consequently, three ML algorithms are compared, k-Nearest Neighbour (*k*-NN), RF, and Support Vector Machine (SVM). Considering the RF algorithm, it was tuned by adjusting three parameters for optimal results. The parameters used are the RF seeds, the number of iterations, and the number of the selected features. The algorithms were evaluated using the Correctly Classified Rate (CCR) measurement. The results indicate that all the classifiers performed adequately with over 99% accuracy. In [37], the authors use RF in comparison with SVM and ANN, to detect road accidents. For the data collection a traffic simulation environment, SUMO [41] is used. Vehicle to vehicle communication is established in the simulation and vehicles exchange information such as speed and location. The aforementioned classifiers are trained using the 10-fold cross validation method and are validated using accuracy, sensitivity, and specificity measurements. The RF algorithm seems to perform better than the other two classifiers in accuracy and sensitivity. Finally, in [23], five ML algorithms are trained using a labeled set of data. The data consist of speed and location data and the label of whether the respective path was congested or not. The tested algorithms are RF, decision tree, MLP, SVM, and Logistic Regression. The evaluation was based on Precision, Recall, and Accuracy, and lead to the result that the Logistic Regression slightly outperformed the other methods due to the power of regression methods against time dependent data. The ML methods mentioned above that do not belong to the ensemble category, will be reviewed separately in the following sections.

2.1.2. Bayesian

A Bayesian network is represented by a Directed Acyclic Graph (DAG), which implements a probability distribution for a given set of random variables. Given a training set as variables, the model can be estimated, which can be used later to assign labels to new data. A Bayesian network is a model that describes all the variables and their relationship. Thus, the observation of some variables in a Bayesian network can give information about the state of another set of variables in the network [42].

In [18], the authors use a method called Bayesian Network Seasonal Autoregressive Integrated Moving Average–(BN-SARIMA), in order to achieve short-term traffic forecasting. The implemented method is a combination of a Bayesian Network and an ARIMA model as an estimator. An ARIMA model is a form of regression analysis, which is commonly used for time series analysis and forecasting. In an ARIMA model, one dependent variable regresses on the previous values of the same variable (AR). The predictions are based on the difference of the variables current and past values in a time series (I). Additionally, the Moving Average (MA) shows that the regression error is a linear combination of previously calculated errors. The authors compare the BN-SARIMA method with two deep learning algorithms. The data are location and speed information of vehicles, obtained from two large datasets. The methods were evaluated against the RMSE, MAE, and MAPE metrics. The results indicate that both deep learning algorithms and the Bayesian method offer similar results in short-term traffic forecasting.

2.1.3. Markov Models

Markov Models are stochastic sequence models based on probability distribution. The simplest Markov model is a Markov chain. In a Markov chain, a distribution of a variable that changes its value randomly over time depends only on the distribution of the previous state. A Hidden Markov Model is similar to a Markov chain, but it uses hidden states for fine tuning. Each state is represented by a probabilistic function of a state. Every HMM can be characterized by the number of the hidden states, the number of the system outputs and the state transition probability distribution [43]. In [22], the authors face motion intention inference, by enabling a cooperative vehicle perception. With the vehicles cooperative sensing, the perception range is increased, offering better perception quality and sufficient response time for autonomous triggered actions. This is achieved using a simplified version of a Coupled Hidden Markov Model (CHMM). The model is trained by using velocity records of various driving profiles of a vehicle sequence. A coupled HMM is a form of HMM, better suited for scenarios where multiple processes are taking place.

To implement a parking space detection system, the authors in [14] use a combination of an SVM and a Markov Random Field (MRF) algorithm. The MRF is an undirected graph describing a set of random variables, which have the Markov property. The authors of this research first obtain input data from cameras fixed at parking lots. The images are pre-processed and the most useful features are extracted. Then the SVM is used to classify parking slots to free or occupied and finally MRF corrects possible conflicts in the SVM classification, improving it by 7.95%.

Markov Decision Process (MDP) is another implementation of the Markov Model. In [36] a group routing suggestion algorithm is proposed based on MDP. Instead of finding the best route for a single vehicle, group routing will be suggested based on vehicle similarities and V2V communication to minimize traffic congestion. The MDP is a model suited for optimization and decision making problems. It is similar to the Markov chains but it introduces two new components. MDP has a set of actions that can lead to a certain state. The selection of the most appropriate actions is encouraged by MDP rewards.

In the study above, the MDP uses a set of actions to make several decisions, the actions selection policy is performed using a reinforcement learning technique, named Q-Learning. Q-Learning is a model-free reinforcement learning method that offers the capability to estimate the reward or the penalty of an action in a Markovian domain. The algorithm learns by executing all the possible actions repeatedly and estimating the resulting state. In the end, the algorithm has learnt the optimum set of actions [44].

2.1.4. Decision Trees

Decision trees are structures that deal with classification tasks and they represent a decision process with several possible outcomes. A decision tree is built top down and consists of several nodes, where every node can be either a class, or a condition that will drive a testing item to a class. It is a

simplistic approach to classification problems. A new classification sub-process takes place at every level of a decision tree, breaking the main task to smaller sub-tasks [45].

As mentioned above, in [23] a decision tree is tested among other four ML algorithms (RF, MLP, SVM, and Logistic Regression), to predict traffic congestion. The decision tree performed equally well with the other algorithms in terms of precision, recall and accuracy, but Logistic Regression was slightly better.

Among the ML algorithms implemented in [27], the authors tested a Regression Tree to forecast short-term and long-term traffic flow in work zones. Regression trees are similar to decision trees but the response variable of the tree is a numeric value instead of a binary class. In a regression tree, the independent variables are used to fit a regression to the response variable. Then, a binary recursive partitioning takes place, where the squared regression error is calculated. The branch with the variables holding the smallest sum of squared errors, is chosen. In the aforementioned research, the regression tree was less effective at the traffic predictions compared to the FF-NN or the RF.

2.1.5. Clustering

Clustering is an unsupervised method to classify elements into discrete groups based on their similarities or discovered patterns [46]. Contrary to the classification methods, in clustering there are no known labels to train a model.

In [26], the authors suggest a Fuzzy C-Means clustering method in order to make short-term traffic predictions. While in a regular clustering method, an instance has to belong to one of the examined classes, in Fuzzy C-Means, every instance belongs to a given class to a certain degree. This degree is described by the so called membership function. In the experiment described in this study, the authors use a traffic simulator with real traffic data from the road network in Japan. They compare the results of the Fuzzy C-Means method to the *k*-Nearest Neighbour (*k*-NN) method, which will be discussed in the instance based algorithms section. The suggested method shows about 26% smaller error rate compared to the *k*-NN. The authors attribute this superiority to the fact that the fuzzy method takes into consideration the inflow and outflow of cars to the selected area rather than using only the data from the area itself.

One of the most popular clustering algorithms is K-Means. The K-Means algorithm's goal is to separate data into a k number of clusters, such that the given data and their cluster's mean value have a minimum squared error [47].

In [12], the purpose is to optimize the traffic network configuration by estimating the best number and location of the processing centers for a sum of suppliers. The authors use K-Means to calculate the distance of the suppliers and the centers, so as to minimize the cost. Prior to the K-Means clustering, the real-time data obtain from GIS devices, are pre-processed with a Deep Belief Network algorithm. The DBN actually finds the initial k spots for the k-Means. To find the location of road anomalies such as bumps, speed breakers, and potholes, the suggested method in [33] uses mobile devices to collect data from their accelerometers. The data are then sent to nearest fog node for processing. K-Means is used to cluster the data and find the threshold values that distinguish a road anomaly from a normal road. A similar approach for road anomalies detection is suggested in [34]. The authors use mobile phones to get raw data of accelerometers and GPS. The data are pre-processed, root mean square is applied on the accelerometer data, threshold values are defined for the z-axis and k-Means is used to cluster the data used to train the system.

2.1.6. Artificial Neural Networks

Artificial Neural Networks are mostly characterized by the structure of the neural network and the learning method. A neural network is formed by creating weighted connections among neurons (Figure 2). In a neural network there is an input layer where the input variables are inserted to the network, and each neuron represents a variable, and there is the output layer where the labels are assigned and each neuron represents a label. Between those layers there are one or more hidden layers.

When there is no looping in the neurons connections the network is called Feed Forward Neural Network (FF-NN), which is most simple form of an ANN.

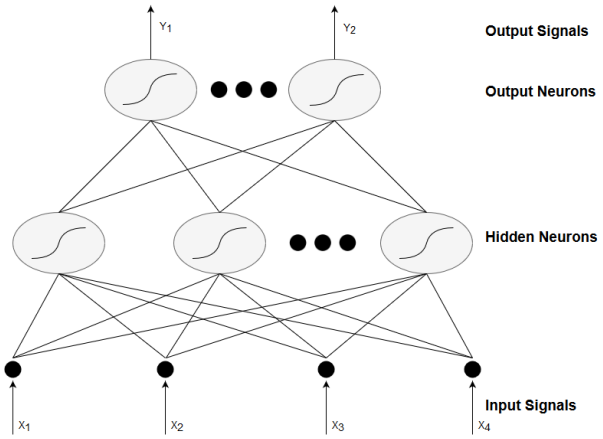

Figure 2. A neural network example consisting of two interconnected layers [9].

In Section 2.1.2, BN-SARIMA algorithm was mentioned and compared with two ANN algorithms, in order to make short-term traffic predictions in large urban traffic networks [18]. The one ANN method was an FF-NN. The predictions in this paper were made for 5, 10, and 15 min in the future. The implemented FF-NN indicated a slight advantage at the 10 and 15 min prediction over the other NN algorithm but it seems to be less effective than the BN-SARIMA approach. As already mentioned, in [27], the authors compared four methods to make short-term traffic predictions in work-zones. Among the methods already analyzed, there is a Multilayer FF-NN. The FF-NN utilizes the linear combination of the input variables to extract features, which are combined using a nonlinear function to predict the traffic flow. This scheme is called a multilayer NN because apart from the input and the output layers, it consists of some hidden layers, where the calculated features are represented. The results highlight that the FF-NN method had the second smallest error metrics after the RF method. The authors in [28], use an FF-NN in order to predict the variations of travel time. Traffic data like speed, history of travel time, and volume are collected from road sensors. Then traffic patterns are clustered with unsupervised learning methods and then an ANN model is trained for each cluster. The ANN which is a FF-NN uses one hidden layer and is trained with the back-propagation technique in order to minimize the predicted error [48]. The output of the FF-NN is a predicted time variation. The full travel time is a sum of the travel time and the variations for each cluster. Back-propagation is a supervised method to train an FF-NN and uses a gradient descent. Taking into consideration the ANN's weighted values, the gradient of the output error is calculated and it is distributed backwards. Multilayered NN use back-propagation like perceptrons use the delta rule. The evaluation of the model is done using the RMSE and the Root Mean Square Percentage Error (RMSP) metrics, which showed satisfactory results. An FF-NN trained with back-propagation is also used in [29] in order to predict road accidents in real time. The FF-NN method is compared with a *k*-NN method and a Regression tree. The models take input data from historical data collected from road sensors, with information such as the number of cars, speed, lane occupancy ratio, etc. The results show that all the models are able to identify accidents with great accuracy. However, there is a significant number of false positive estimations. In another research [30], a method to identify potholes using an ANN is suggested. Accelerometer data are collected from Android phones, feeding an FF-NN which runs in the Android phones in real time, using the 'Encog' framework. The FF-NN consists of 6 hidden layers and it is trained with back-propagation. The results indicate a 90%–95% detection accuracy. As mentioned before, in [23] a comparison of the following algorithms is performed for traffic congestion prediction: decision tree, RF, MLP, SVM, and Logistic Regression. MLP stands for a MulitLayer Perceptron [49]. It is a simple FF-NN consisting of multiple perceptrons forming layers.

2.1.7. Deep Learning

The term Deep Learning encloses powerful methods for reinforcement learning able to process a large amount of unstructured data. Deep learning techniques are suited for big-data handling and computationally intense processes like image pattern recognition, speech recognition and synthesis, etc. As the needs for CPU power increases, powerful GPUs are widely used to perform Deep learning tasks. Like ANNs, with deep learning, deep neural networks are created. The word 'deep' refers to the large number of hidden layers that compose the neural network. The number of layers in a deep learning technique is tight to the number of calculated features. In deep learning, the features are automatically estimated and there is no need for feature calculation and extraction before applying such a method. Additionally, a large variety of network structures is introduced with the advance of deep learning [50].

A deep learning approach based on Convolutional Neural Networks (CNN) is used in [20]. CNNs are widely used in the literature for image recognition. A CNN is a type of ANN since it consists of an input, an output layer, and some hidden layers. The hidden layers in a CNN architecture can be pooling, convolutional, non-linear, subsampling, or Fully Connected (FC) layers. At convolutional layers, features are learned from the input data. The features or filters consist of weight and bias values. A significant characteristic of CNN is that multiple neurons use the same filters. Neurons are units that apply a transfer function on the weighted sum of its input signals [9]. Convolutional layers are followed by non-linear layers, converting all negative values to zeros. Then, dimensionality reduction is applied with sub-sampling layers. Finally, an FC layer which is actually a Multi-Layer Perceptron performs the classification of the input data. The authors use CNN to detect a parking lot occupancy based on smart cameras input running with Raspberry Pi 2 as a controller. The goal is to execute the CNN method exclusively on the smart cameras, so a minimized version of the AlexNet model [51] is implemented (mAlexNet). The evaluation is performed using a dataset created by the authors (CNRPark-EXT) and a well-known dataset (PKLot) [38], achieving an accuracy of 98.27% and 90.13%, respectively. Additionally, an extended review in deep learning methods based on CNNs and Deeper CNNs (DCNNs) was done in [21]. The author summarized studies whose main purpose is the pavement distress detection using deep learning techniques. The reviewed studies apply CNN and DCNN methods in a wide variety of image datasets, such as Google StreetView images, smartphone images, and 3D images constructed with specialized hardware (e.g., a Ground Penetrating Radar (GPR) [52]). In the review's conclusion, it is mentioned that both CNN and DCNN perform very well in pavement image classification.

In [19], the authors study the development of a blind-spot detection system for smart vehicles. The proposed method is based on Fully Connected Networks (FCN), which are said to perform better than CNN in real-time image input. The proposed system uses input from three cameras mounted outside a car. An FCN is similar to a CNN, lacking the convolution layers. The term Fully Connected means that every neuron in the previous layer is connected to every node in the proceeding layer. The studied method utilizes an adaptation of Adam Optimizer [53] and Relu [50] activations.

As mentioned earlier, the purpose of [12] is to perform real-time traffic network assignments. This study uses a Deep Belief Network (DBN) to pre-process input data and initialize the clustering centers that are fed in K-Means to perform the needed clustering. The DBN method is designed with two hidden layers, based on the Restricted Boltzmann machine (RBM), to enable training simplicity. A DBN a set of unsupervised layers such as RBM. Such a network has connections between the layers, but not inside the layer. DBNs can be also be trained greedily layer by layer. DBNs are also used in [24], in order to assist digital map creation by automatic street elements detection such as traffic lights, roundabouts, etc. The input data for the system are obtained only by users GPS data. The first step in their method is the implementation of an outlier detection technique in order to make some early estimations for the street elements position. The DBN is a stack of RBMs. An RBM is described by an energy function and a probability distribution. The DBN tries to maximize the energy function. For each and every class, the DBN will extract a set of features that describe accurately the input data.

After the DBN training, a final classifier is used to finally separate input data to classes. The research combined results present a recall of 0.89 and a precision of 0.88.

The authors in [25], developed a deep learning method for sequential classification of multiple objects from an image input. The method developed is called Deep Recurrent Attention Model (DRAM) and the research results indicate that the model prevails over CNNs on multiple digits recognition when tested on Google street view house images. Additionally DRAM seems to be less computationally intense than the best CNNs. A DRAM architecture consists of the following five network structures or inter-connected hidden layers: (a) A glimpse network, which consists of three convolutional layers and a fully connected layer, and its purpose is to extract the important features from an input image chunk. (b) A recurrent network, that combines the information of glimpse networks output preserving spatial information. (c) An emission network, which predicts the next image chunk position to feed the glimpse network. (d) A context network, which uses a low resolution version of the initial image and helps emission network to find significant objects in the image. (e) The classification network, that performs the object classification based on the final feature vector.

To enable traffic flow predictions based on big data, a Stacked Auto Encoder (SAE) is suggested in [31]. On top of the SAE a Logistic Regression algorithm is used to enable traffic flow predictions. The network layers are trained as Auto Encoders (AEs) with a greedy layer-wise unsupervised learning method and it is fine-tuned using Back Propagation in the end. The proposed method is tested using a Caltrans Performance Measurement System (PeMS) dataset and the evaluation is based on the MRSE, MRE, and MAE indexes. The results indicate that the SAE outperforms the compared techniques (Random Walk, Support Vector Machines, and Radial Basis Function). AEs are layered structures that aim at reconstructing the input data, in an unsupervised way, in order to extract more robust features. Stacked Auto Encoders use multiple AEs, where the output of the first becomes the input of the other. The AEs are trained with an unsupervised layer-wise manner until all the stack is trained. Then a technique like Back Propagation is used, based on labeled data, so as to update the network's weights [54].

In a research about traffic accident hotspots and their automated detection and classification [32], the authors train an Inception Neural Network to help with the image classification. Analysis on road data obtained from the Swiss Road Authority (FEDRO) shows that there is a connection with the accident occurrence and the actual accident spot. Thus, Google Maps images, combined with the FEDRO data, are used to train an inception neural network (TensorFlow Inception v3) and attempt to detect accident-prone areas. The results show a 30% accuracy, which confirms the location–accident correlation hypothesis. The inception network is more advanced than a CNN and it intends to be a less computationally intense method. The idea is to use various filters to perform the convolution of the input data. Additionally, maximum pooling is also performed. The outputs are concatenated and sent to the next inception layer. Inception layers stacked form an Inception Network. In the inception network structure, there are also some auxiliary classifiers. These are structures used to compute an auxiliary loss on the inception layers and support the training phase [55].

A comparison of 3 methods was mentioned earlier to achieve short-term traffic predictions in large urban traffic networks [18]. The authors compared the performance of BN-SARIMA, an FF-NN, and a Nonlinear Autoregressive Exogenous Model (NARX). The evaluation was performed with the MAE, MAPE, MRSE indexes and considering the NARX method they revealed that it gave reliable 5 min predictions, similar to the BN-SARIMA. However, NARX performance was poorer in the other time ranges predictions. NARX is a recurrent Neural Network that finds a correlation between the current values of a time series, with older values of the same series.

2.1.8. Instance Based

Instance Based algorithms perform data classification based on the comparison of new test data, directly to the training instances. The comparison based on a similarity function and its output is fed

into a classification function [56]. Instance based algorithms store training instances in their memory, therefore they can be pretty memory consuming as the data are increased.

Between an FF-NN, and Regression Tree, the instance based algorithm *k*-Nearest Neighbor (*k*-NN) is used in [29] to make real-time autonomous accident detection as mentioned in Section 2.1.6, *k*-NN is an algorithm used for supervised classification and regression. The main principle of *k*-NN is that similar data have minimum distance with each other. *k*-NN is run multiple times on the testing data in order to select the ideal *k* factor that minimizes the errors while classifying data. We have already met *k*-NN in [35], where road surface abnormalities are detected using accelerometer data and applying *k*-Nearest Neighbour (*k*-NN), RF, and Support Vector Machine (SVM) algorithms. The parameter *k* of *k*-NN represents the number of the test sample's neighbours. It was adjusted empirically to optimize the results. All the classifiers presented an accuracy over 99% based on the CCR measurements. Finally, in [24], the problem of the automated detection of street elements dealt with the application of a DBN is shown. At the end of the DBN a classifier is used for the final element classification. When using *k*-NN as the selected classifier, a precision close to 90% was reached for street crossings, while a percentage of 80% was achieved for traffic lights and roundabouts.

2.1.9. Regression Analysis

Methods in the regression analysis category try to construct mathematical models able to describe, or identify, the relationship between two or more variables. The dependent or response variable is the system's output. By fitting a regression model to the system, the relations between the dependent variable and the independent ones (regressors) can be discovered.

We have already discussed some of the methods used in [23] to perform traffic congestion predictions. Logistic Regression is compared to a decision tree, an RF, an MLP, and an SVM algorithm. Since the input data are time dependent, the Logistic Regression in this research clearly outperformed the other methods giving 100% Precision, 99.5% recall, and 99.9% Accuracy. Additionally, the small complexity of the Logistic Regression renders it capable to be executed by low end devices according to the authors. Logistic regression is a form of regression analysis, used to describe the relationship of a dependent variable and one or more independent variables. Logistic regression is applicable when the dependent variable has binary values. Since it is a regression analysis, the logistic regression is suitable for prediction tasks in ML.

2.1.10. Non-Probabilistic Linear Classification

Depending on the type of a classification problem, one can use either a probabilistic or a non-probabilistic approach. When the probabilities play no apparent role in the data classification, non-probabilistic methods are usually preferred. Apart from the problem type, there are other parameters to consider that can lead to a non-probabilistic solution. Non-probabilistic classification is useful when dealing with structured data. Moreover, an important aspect to consider is whether cluster construction or assignment is needed in the respective classification problem [57].

We have seen in [35], the comparison of *k*-NN, RF, and Support Vector Machine (SVM) in order to identify road surface abnormalities. Linear SVM, as well as the other two classifiers, performed very well for this task, with a CCR Rate over 99%. SVM is a binary classifier used to separate labeled data in an optimum way using a hyperplane. The hyperplane needs to have a maximum distance from the data of both classes. The data points at that are closer to the hyperplane are called support vectors because they play a significant role in the hyperplane's orientation and position. SVM has also been met in [37], dealing with the task of road accident detection. The SVM algorithm was compared to an ANN and an RF Method. The results indicated that the SVM had better Specificity than the RF but the least good accuracy of the three algorithms. The last method found in the research for traffic congestion predictions [23] is SVM. In this study, it was compared to a decision tree, an RF, an MLP and a Logistic Regression algorithm. SVM performed equally well with the other algorithms, slightly outperformed by the Logistic Regression. The MRF algorithm was discussed earlier, in the

research about a parking space detection system [14], as a way to correct possible conflicts in the SVM classification. This research a modified version of the classic binary SVM. A one-against-one strategy enables the implementation of a multi-class SVM classifier. So, SVM takes image patches as input data and attempts to classify the detected parking spaces to 8 possible classes. The SVM kernel function is a Radial Basis Function (RBF) adoption. The purpose of [38] is to create a dataset with images from parking lots, collected from HD cameras, located near parking lots (PKLot). To test the dataset capabilities, an SVM classifier was used. Two texture based features where used: Local Binary Patterns (LBP) and Local Phase Quantization (LPQ). The best kernel for the SVM problem was found to be Gaussian. The results were evaluated using the Overall Error Rate (OER) metric. Two SVM versions are also applied in [24]. The research goal is the automatic detection of street elements, as discussed earlier. A Deep Belief Network is applied in order to extract the best features and then a classifier is used to separate the elements. Apart from the *k*-NN, an SVM with a quadratic and an SVM with a Gaussian kernel were used offering satisfactory results.

2.2. ML and IoT Applications in Smart Transportation

In this section, we review and summarize based on the type of the Smart Transportation problems they deal with. We have identified six major categories of Smart Transportation challenges, and we will present the methods approached by the authors using IoT technology and ML algorithms. Table 2 groups all the researches in the six aforementioned categories, indicating also whether an ML technique is used or not in the respective research.

Table 2. References to IoT smart transportation applications.

No	ITS Application	References-ML	References-No ML
1	Route Optimization-Navigation	[12,18,23,27,28,31,36]	[13,58–62]
2	Parking	[14,20,38]	[63–68]
3	Lights		[69–71]
4	Accident Detection	[17,19,22,24,25,29,32,37]	[72]
5	Road Anomalies	[21,30,33–35]	
6	Infrastructure		[16,73,74]

The ML supported methods have been extensively discussed in the previous section. For the sake of completeness and to avoid repetition, these methods will be just referenced in the respective category and we will elaborate on the non ML approaches.

2.2.1. Route Optimization

Traffic congestion is a common issue in urban areas and it is only getting worse as the number of vehicles increases. Route optimization is a method to propose the best route for a specified destination, in order to minimize traffic congestion. By minimizing the traffic congestion, both the traveling time and vehicle emissions are reduced [13]. The route optimization problem has been widely challenged in the literature by various ML approaches over the IoT infrastructure. Namely, in [36], an MDP algorithm and V2V communication have been used for a group routing to minimize traffic congestion. In [12], a combination of k-Means and DBN is preferred to optimize the traffic network configuration. Short-term traffic forecasting is addressed in [18], and it is faced with multiple approaches such as a BN-SARIMA, an FF-NN, and a NARX model. Short-term along with long-term traffic predictions have been also challenged in [27] comparing the performance of four ML methods: RF, a baseline predictor, a regression tree, and an FF-NN. An FF-NN trained with a BP is also selected in [28] to calculate time-travel variations and estimate the total travel time. The authors in [23] dealt with traffic congestion predictions comparing a Logistic Regression with SVM, a decision tree, an RF, and an MLP method. Finally, ref. [31] performs traffic flow predictions based on big data, using a Stacked Auto Encoder approach.

The idea in [58] is to explore the capabilities of Mobile Crowd-Sensing for Intelligent Transportation Systems, by using a swarm intelligence algorithm for route optimization. The authors implement a Modified Crowd-Sensing version of the Ant Colony Optimization algorithm (MoCSACO). Users will exchange information with each other and will navigate to less congested routes, following the messages received by other users, similar to the way ants follow pheromone paths when searching for food.

An innovative approach to support route optimization using IoT technology is described in [13]. Scalable Enhanced Road Side Units (SERSU) are used alongside with a Master Control Center (MCC). The SERSU consists of, a camera, weather condition sensors, pollution sensors, a wireless modem, and a Radio Frequency module, and is capable of monitoring the traffic and the environmental conditions and sending the data to the MCC through the 4G network. The MCC processes the data and can adapt the speed limit of each road based on the congestion, the weather conditions, and the pollution levels and send the new speed limits and route suggestions back to the SERSU in order to notify the drivers. In this way, traffic congestion can be diminished and possible accidents due to bad weather conditions can be avoided.

A Vehicular Ad-hoc Network (VANET)-based algorithm for enhanced route planning is implemented in [59]. The authors design a mobile navigation application that collects traffic data and proposes the optimum route for the shortest travel time or the least fuel consumption. In a VANET, the cars exchange information when neighbouring cars are within a IEEE 802.11p range, in order to give real-time updates about the status of a road even when an abnormal event has occurred in a route. Traffic information is also collected from the Google Maps application for the routes that no VANET information is available. The traffic information from the two sources is combined and the best route is proposed to the user.

The authors in [60] experiment in finding a possible correlation among traffic congestion, fuel emissions, and the crowdsourced information of Google 'popular times' feature. In the research, traffic data are collected by cameras in specific roads, and a vehicle equipped with a GNSS data logger. Emissions are estimated based on the Vehicle Specific Power (VSP) model, and popular time data are obtained from Google Maps. The results showed that there is a correlation between the crowdsourced data of 'popular times' and emissions, although calibration among the data needs to done, and some adaptive learning algorithm to analyze similar cases would be needed to make accurate correlations.

Google was one of the first to use the power of crowdsourcing, in order to create new services. Google Maps application is available for free and is compatible with all modern mobile devices. Mobile devices have integrated GPS, accelerometer, and gyroscope sensors. Into 2009, Google announced a new service that would provide traffic information integrated in Google Maps [61]. The traffic information was not collected by fixed location sensors or other monitoring systems. The end users mobile devices can send anonymous information about their speed and location just by using the maps application. Google Maps can now suggest new routes based on traffic information, in order to avoid congestion.

Lastly, in [62], crowdsourcing route planning in performed for the last mile towards a destination. The authors state that the last mile is usually not accurately planned by simple approaches, such as shortest path or shortest time. Locals on the other hand, could share their driving patterns around various destinations, and that would provide more accurate directions at the last segment of the navigation. A mobile application (CrowdNavi) is implemented in the research that uses crowdsourcing to obtain information and make suggestions about the last mile of a trip. Part of the application is to identify the last segment. The first part is suggested using information from services like Google Maps, and the last segment is calculated by crowd data and a method called landmark scoring.

2.2.2. Parking

Parking applications are developed to effectively track availability in parking lots, and offer some reservation options to the users, and even some parking detection and notification mechanisms.

Many IoT devices have been used to detect car presence in a parking spot and convey the information to a centralized system. Additionally, some researches use image data to massively detect free parking slots, applying ML algorithms to the data.

The authors in [20], use CNNs on data obtained from smart cameras on parking lots, to detect free parking slots. Parking lot images are also processed in [14], using a combination of an SVM and an MRF algorithm. Lastly, the PKLot dataset is created in [38], which is afterwards tested using an SVM algorithm.

A smart parking approach is suggested in [63], with the idea of an IoT supported parking lot and a smart signboard to convey related information. The parking lot will use ultrasonic sensors to check the parking space availability and a WiFi module will collect and send the data to a cloud server. At this point, a user can check the parking availability through a mobile application, or the smart signboard. The signboard consists of an LCD or LED screen driven by a raspberry pi that will collect and display information about parking availability, weather conditions, distance to certain destinations, etc.

Ref. [64] also uses ultrasonic sensors at each parking slot to detect the availability. The sensor is connected with an Arduino Uno that uses an ESP8266-01 WiFi module, which is responsible for sending the data to a cloud server. The communication is done via the MQTT protocol. The cloud server operates running ThingSpeak which is an IoT platform and offers a variety of management and monitoring capabilities to the users. Finally, the end users can install an android application that enables them to reserve parking slots and automate the parking payments.

The authors in [65] propose a smart parking solution using a hardware and a software part. For the hardware part, magnetic sensors are used to detect cars on parking slots, and a gateway device is located on the road side to gather the information and send it to a remote server. The goal of the software part is to propose the nearest free parking slot to the user. This is achieved by running a genetic algorithm to obtain the minimum route from the users' location to the nearest free slot. Genetic algorithms are actually an optimization methodology, and thus we do not consider them as classic ML.

An end-to-end smart parking implementation is presented in [66]. The system consists of physical sensors and micro-controllers at the parking slots, a modular cloud server, a mobile application, and a third party payment service. At the parking slots, geomagnetic vehicle detectors check for vehicle presence and send the data to the cloud server with the use of a BC95-B5 NB-IoT module. The cloud server has several modules. A basic information module that manages the sensor nodes and other administrative tasks, a module with the system's maintainers manageable information, a charging module that calculate charging and sends reports to the third party charging service, a sensor node surveillance system that enables monitoring of the parking slots availability and the sensor's operating status, a business intelligence module that enables parking data querying and visualization, and a task management service that handles requests to the maintainers and status tracking. The end user has access to the system via a web based mobile application. Finally, payments with a third party paying service like Alipay and WeChat pay is enabled through the cloud server and the mobile web-app.

Another approach to smart parking is introduced in [67] with the creation of an Agent-Oriented Smart Parking Recommendation System (ASPIRE). In the ASPIRE system, users can set up their parking preferences, like maximum walking time, preferred location, etc, through a 'Local Agent' and the system will take all these data into consideration when a parking reservation request is made. A cloud based software agent will then decide list with most suitable parking spots for the user. The user is notified through the Local Agent and chooses the preferred parking slot. The parking recommendation algorithm is based on the Analytic Hierarchy Process (AHP), a structured mathematical technique to analyze complex decisions. Parking lots are equipped with RFID readers at the entrance that will track and identify cars entering and leaving the parking. The cars must also be equipped with an RFID tag to make the identification possible.

A reliability analysis of a smart parking system is conducted in [68], resulting in a success rate greater than 96%. The system uses ultrasonic sensors controlled by an ESP8266 controller with

embedded WiFi. The sensors at each parking spot communicate with a private cloud server using a REST API. The sensor registration and their availability status is managed by the Orion Broker which is part of the FIWARE IoT platform. On top of the described setup resides an application layer that offers a user friendly interface for the users to access through their mobile devices.

2.2.3. Lights

An important part of a smart city, which we consider a part of the smart transportation services, are the Smart Street Lights (SSL). Smart lights can reduce energy consumption and offer dynamic operation and manageability.

An SSL implementation based on the IoT infrastructure is implemented in [69]. Street lights obtain smart characteristics by adding a light sensor, an IR sensor, GPS and a wireless communication module. In this way the lamps can be aware of crowded areas and adapt their light intensity dynamically, making areas with dense population safer and also saving energy. The GPS can help a centralized system to monitor the lamps, location and state, and speed up maintenance processes in case of a broken lamp. The communication of the SSL and the management system is based on the NB-IoT network. The management system is based on Fog nodes, that collect data from a collection of lamps and check their state on a regular basis. Apart from the automated procedures that the SSLs provide, they can be managed remotely through the implemented management platform.

A similar but more simplistic approach for smart lights is presented in [70]. In this implementation, a raspberry pi is used as a micro-controller, connecting the lamp with a light sensor, an IR sensor, and an IR led. The light sensors will detect sun rise and set and will trigger the lamp with on/off signals. Additionally, the lamps are able to detect bypassing cars or pedestrians, and switch the lamps dynamically to reduce energy consumption.

Finally, in [71] the authors propose a smart lighting system, where each lamp post will be a WiFi hotspot that will enable to transfer a different kind of collected information to a central web server. The lights will dynamically switch on/off or dim depending on the surrounding environment to enable significant cost reduction. Lamp posts will be also equipped with cameras and environment checking sensors to ensure people safety during emergencies and measure environmental conditions accordingly and enhance the capabilities of a regular lamp post.

2.2.4. Accident Detection/Prevention

Accident detection and prevention is a domain of smart transportation and a crucial activity for every city since an operating prevention method can help save human lives. Road accidents can be prevented if the drivers stay more focused throughout the traveling period. An accident prevention system can notify the driver about critical situations and allow them to act in a timely manner. Accident detection can also reduce the accidents number and the traffic congestion, by identifying accident prone areas or accidents that have taken place in the live traffic network. ML has been proven very useful in detecting road accidents, or detect patterns that could lead in new accidents and notify drivers in order to avoid them.

The authors in [22] developed a method for cooperative vehicle perception, using a CHMM method so that information can be shared among vehicles in a timely manner to avoid accidents. Real-time autonomous accident detection is performed in [29], by analyzing data obtained from road sensors with an FF-NN, a Regression Tree, and the *k*-NN algorithms. V2V communication together with the use of an RF method compared with an SVM and an ANN is used for accident detection in the roads [37]. A novel approach to detect objects in the blind spot of smart vehicles is used in [19], using an FCN method. The DRAM method developed in [25] can be also used for road accident detection since it can help with multiple object detection from image data. In the research [17], the drivers' consciousness is monitored with image data, processed by AdaBoost to prevent possible accidents. An Inception Neural Network is used in [32], to detect accident prone areas. Finally, the work of [24]

to detect street elements could also support accident prevention. The research uses a combination of a DBN and a *k*-NN or SVM classifier.

The authors in the research [72] propose an IoT cloud platform that will enable traffic visualization and early notifications about sudden slowdowns that can lead to accidents. The implemented setup would provide Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), and a novel approach called IoT as a Service (IoTaaS). GPS data are collected by devices placed in volunteer vehicles, and they are sent in a cloud server via a 4G network. The cloud server handles GPS data with the OpenGTS platform and enables traffic visualization with OpenStreetMaps. The data are stored in an SQL and a Distributed MongoDB to enable further analysis. The back end scalability is supported with the use of Docker containers. The system response time a key element in the success of the implementation, and the suggested systems achieves sending an alert for a 1 km distance in about 120 ms.

2.2.5. Road Anomalies Detection

Road anomalies detection has a significant role in smart transportation since the road conditions have an immediate impact on many aspects of transportation. The main purpose of a road anomaly detection system is the detection of bumps and pot-holes on the roads and the notification of the drivers. Bad road conditions can lead to vehicle damage, road accidents, and traffic congestion. The problem of detecting road anomalies is a task very suitable for some ML techniques, so the literature reviewed here follows the ML direction too.

Accelerometer data collected from mobile devices are processed with an FF-NN in [30], in order to detect pot-holes. Accelerometer data are also used in [33], clustered by the K-Means algorithm to distinguish normal from abnormal roads. A similar approach to detect road anomalies is used in [34], and the data in this research are also clustered using the k-Means method. Last but not least, in [35] the authors compare the methods *k*-NN, RF, and SVM for the task of road abnormalities detection.

The author in [21] has conducted a detailed review of deep learning techniques applied for pavement distresses detection. Furthermore, a review of the existing and emerging deep learning frameworks used for road anomalies detection is performed. The review summarizes 12 of the most recent studies in the pavement distress detection which are based on CNN and DCNN methods. Moreover, the reviewed studies are compared in terms of objectives, datasets used, hardware, network architecture, hyper-parameters, and selected framework. Finally, the author concludes that CNNs show the best results in terms of pavement image classification in most studies. There is no clear indication that DCNNs can be more accurate than shallower CNNs and there are a lot of other parameters that affect the classification results, such as data pre-processing, augmentation, and feature selection. Unsupervised end-to-end image classification has yet to be explored in this field of study but is a promising method for the future of pavement distress detection.

2.2.6. Infrastructure

The rise of IoT technology has benefited modern transportation in many ways. It has created not only new applications, making transportation smarter, but also novel ways of thinking. Changing the infrastructure of Intelligent Transportation Systems, widely expands the systems' capabilities.

A new way of communication is proposed in [73]. The authors suggest and simulate a Vehicle to Vehicle (V2V) communication framework based on the IoT principle of M2M communication. In the proposed framework, the cars will be aware of their position using GPS and will exchange information about their speed, movements, location with the cars near them, and, at the same time, uploading the data at a server. In that way, accidents can be avoided by early notifying following cars about a sudden speed change, or traffic congestion information could be distributed to other cars to enable better navigation services.

In [74], a combination of hardware and software is proposed to support bus fleet monitoring and improve the user's experience. The proposed hardware consists of RFID tags to uniquely identify

buses, IR sensors to count passengers entering and leaving the bus, and GPS to track the bus location in real time. All the collected data are collected to a cloud server using a TI CC3200 microcontroller with embedded WiFi module. There is also a TI CC3200 module at every bus stop, connected to an LCD to present the available information to the passengers. The information is also available through a mobile application offered to the users.

Combining the social network principles with the internet of things for smart transportation application, the Social Internet of Vehicles (SIoV) term arises. The authors in [16] propose a cross layered Vehicular Social Network Protocol (VSNP) to reduce communication congestion in SIOV. The protocol spans to the MAC, Physical, and network layers to speed up the communication. The MAC layer divides circular time slots in a ring type routing. The physical layer consists of Wireless Sensor Network nodes and the network layer supports the routing from the outer rings to a fixed access point. The suggested protocol performs better than an existing protocol (MERLIN) in Matlab conducted simulations.

3. Discussion/Conclusions

A review on Machine Learning and Internet of Things techniques exploited for smart transportation applications has been presented. This study highlighted the fact that a wide variety of Machine Learning algorithms has been proposed and evaluated for Smart Transportation applications, indicating that the type and scale of IoT data in these applications is ideal for ML exploitation. On the other hand, given the current applications and infrastructure regarding IoT and ML, a comparatively smaller ML coverage for the smart lighting systems and parking applications is detected. Therefore, there is a definite need for supplementary coverage in those areas, from the ML perspective, in the future.

Regarding the IoT approaches for the ITS application categories, route optimization, parking, and accident prevention/detection have proven to be the most popular among them.

Considering the problems that the smart transportation applications address, some common points of interest have been identified from this review. These are: environmental precautions, transport finance, human safety, and time saving.

Additionally, the great progress which has already been achieved in the field of smart transportation with the help of IoT and ML became evident, while an even better advancement in this topic is expected in the upcoming years. As the number of IoT devices rises, the data diversity and volume scale up, therefore, ML can create many meaningful applications.

Author Contributions: Conceptualization, F.Z., G.K. and S.K.; Methodology, F.Z., G.K., S.K. and D.K.; Investigation, F.Z.; Resources, F.Z.; Writing—Original Draft Preparation, F.Z.; Writing—Review & Editing, G.K., S.K. and D.K.; Visualization, F.Z.; Supervision, G.K., S.K. and D.K.;

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
GIS	Geographic Information System
GNSS	Global Navigation Satellite System
GPRS	General Packet Radio Service
GPS	Global Positioning System
GPU	Graphics Processing Unit
GSM	Global System for Mobile Communications
HAAR	Human Automated Activity Recognition

HD	High Definition
HTTP	HyperText Transfer Protocol
IR	Infra-Red
LTE	Long Term Evolution
MQTT	Message Queuing Telemetry Transport
NB-IoT	Narrow Band - IoT
RFID	Radio Frequency IDentification
STOMP	Streaming Text Oriented Messaging Protocol
XMPP	Extensible Messaging and Presence Protocol

References

1. Swan, M. Sensor mania! the internet of things, wearable computing, objective metrics, and the quantified self 2.0. *J. Sens. Actuator Netw.* **2012**, *1*, 217–253. [CrossRef]
2. Vangelista, L.; Zanella, A.; Zorzi, M. Long-range IoT technologies: The dawn of LoRa™. In *Future Access Enablers of Ubiquitous and Intelligent Infrastructures*; Springer: Cham, Switzerland, 2015; pp. 51–58.
3. Hashem, I.A.T.; Chang, V.; Anuar, N.B.; Adewole, K.; Yaqoob, I.; Gani, A.; Ahmed, E.; Chiroma, H. The role of big data in smart city. *Int. J. Inf. Manag.* **2016**, *36*, 748–758. [CrossRef]
4. Naik, N. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In Proceedings of the 2017 IEEE international systems engineering symposium (ISSE), Vienna, Austria, 11–13 October 2017; pp. 1–7.
5. Satyanarayanan, M. The emergence of edge computing. *Computer* **2017**, *50*, 30–39. [CrossRef]
6. Luan, T.H.; Gao, L.; Li, Z.; Xiang, Y.; Wei, G.; Sun, L. Fog computing: Focusing on mobile users at the edge. *arXiv* **2015**, arXiv:1502.01815.
7. Talari, S.; Shafie-Khah, M.; Siano, P.; Loia, V.; Tommasetti, A.; Catalão, J. A review of smart cities based on the internet of things concept. *Energies* **2017**, *10*, 421. [CrossRef]
8. Mohammed, M.; Khan, M.B.; Bashier, E.B.M. *Machine Learning: Algorithms and Applications*; Crc Press: Boca Raton, FL, USA,2016.
9. Kubat, M. *An Introduction to Machine Learning*; Springer: Cham, Switzerland, 2017.
10. Wu, Q.; Ding, G.; Xu, Y.; Feng, S.; Du, Z.; Wang, J.; Long, K. Cognitive internet of things: A new paradigm beyond connection. *IEEE Internet Things J.* **2014**, *1*, 129–143. [CrossRef]
11. Madden, S. From databases to big data. *IEEE Internet Comput.* **2012**, *16*, 4–6. [CrossRef]
12. Yang, J.; Han, Y.; Wang, Y.; Jiang, B.; Lv, Z.; Song, H. Optimization of real-time traffic network assignment based on IoT data using DBN and clustering model in smart city. *Future Gen. Comput. Syst.* **2017**, in press. [CrossRef]
13. Al-Dweik, A.; Muresan, R.; Mayhew, M.; Lieberman, M. IoT-based multifunctional scalable real-time enhanced road side unit for intelligent transportation systems. In Proceedings of the 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, ON, Canada, 30 April–3 May 2017; pp. 1–6.
14. Wu, Q.; Huang, C.; Wang, S.Y.; Chiu, W.C.; Chen, T. Robust Parking Space Detection Considering Inter-Space Correlation. In Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, 2–5 July 2007.
15. Araújo, A.; Kalebe, R.; Girão, G.; Filho, I.; Gonçalves, K.; Melo, A.; Neto, B. IoT-Based Smart Parking for Smart Cities. In Proceedings of the 2017 IEEE First Summer School on Smart Cities (S3C), Natal, Brazil, 6–11 August 2017; pp. 31–36.
16. Jain, B.; Brar, G.; Malhotra, J.; Rani, S.; Ahmed, S.H. A cross layer protocol for traffic management in Social Internet of Vehicles. *Future Gen. Comput. Syst.* **2018**, *82*, 707–714. [CrossRef]
17. Ghosh, A.; Chatterjee, T.; Samanta, S.; Aich, J.; Roy, S. Distracted Driving: A Novel Approach towards Accident Prevention. *Adv. Comput. Sci. Technol.* **2017**, *10*, 2693–2705.
18. Fusco, G.; Colombaroni, C.; Comelli, L.; Isaenko, N. Short-term traffic predictions on large urban traffic networks: Applications of network-based machine learning models and dynamic traffic assignment models. In Proceedings of the 2015 IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), Budapest, Hungary, 3–5 June 2015; pp. 93–101.

19. Kwon, D.; Park, S.; Baek, S.; Malaiya, R.K.; Yoon, G.; Ryu, J.T. A study on development of the blind spot detection system for the IoT-based smart connected car. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 12–14 January 2018; pp. 1–4.
20. Amato, G.; Carrara, F.; Falchi, F.; Gennaro, C.; Meghini, C.; Vairo, C. Deep learning for decentralized parking lot occupancy detection. *Expert Syst. Appl.* **2017**, *72*, 327–334. [CrossRef]
21. Gopalakrishnan, K. Deep Learning in Data-Driven Pavement Image Analysis and Automated Distress Detection: A Review. *Data* **2018**, *3*, 28. [CrossRef]
22. Liu, W.; Kim, S.W.; Marczuk, K.; Ang, M.H. Vehicle motion intention reasoning using cooperative perception on urban road. In Proceedings of the 2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014; pp. 424–430.
23. Devi, S.; Neetha, T. Machine Learning based traffic congestion prediction in a IoT based Smart City. *Int. Res. J. Eng. Technol.* **2017**, *4*, 3442–3445.
24. Munoz-Organero, M.; Ruiz-Blaquez, R.; Sánchez-Fernández, L. Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on GPS traces while driving. *Comput. Environ. Urban Syst.* **2018**, *68*, 1–8. [CrossRef]
25. Ba, J.; Mnih, V.; Kavukcuoglu, K. Multiple object recognition with visual attention. *arXiv* **2014**, arXiv:1412.7755.
26. Kanoh, H.; Furukawa, T.; Tsukahara, S.; Hara, K.; Nishi, H.; Kurokawa, H. Short-term traffic prediction using fuzzy c-means and cellular automata in a wide-area road network. In Proceedings of the 2005 IEEE Intelligent Transportation Systems, Vienna, Austria, 16 September 2005; pp. 381–385.
27. Hou, Y.; Edara, P.; Sun, C. Traffic flow forecasting for urban work zones. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 1761–1770. [CrossRef]
28. Yu, J.; Chang, G.L.; Ho, H.; Liu, Y. Variation based online travel time prediction using clustered neural networks. In Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems, Beijing, China, 12–15 October 2008; pp. 85–90.
29. Ozbayoglu, M.; Kucukayan, G.; Dogdu, E. A real-time autonomous highway accident detection model based on big data processing and computational intelligence. In Proceedings of the 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 5–8 December 2016; pp. 1807–1813.
30. Kulkarni, A.; Mhalgi, N.; Gurnani, S.; Giri, N. Pothole detection system using machine learning on Android. *Int. J. Emerg. Technol. Adv. Eng.* **2014**, *4*, 360–364.
31. Lv, Y.; Duan, Y.; Kang, W.; Li, Z.; Wang, F.Y. Traffic Flow Prediction With Big Data: A Deep Learning Approach. *IEEE Trans. Intell. Transp. Syst.* **2014**, *16*, 865–873. [CrossRef]
32. Ryder, B.; Wortmann, F. Autonomously detecting and classifying traffic accident hotspots. In Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and 2017 ACM International Symposium on Wearable Computers, Maui, HI, USA, 11–15 September 2017; pp. 365–370.
33. Al Mamun, M.A.; Puspo, J.A.; Das, A.K. An intelligent smartphone based approach using IoT for ensuring safe driving. In Proceedings of the 2017 IEEE International Conference on Electrical Engineering and Computer Science (ICECOS), Palembang, Indonesia, 22–23 August 2017; pp. 217–223.
34. Ghadge, M.; Pandey, D.; Kalbande, D. Machine learning approach for predicting bumps on road. In Proceedings of the 2015 IEEE International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Davangere, India, 29–31 October 2015; pp. 481–485.
35. Ng, J.R.; Wong, J.S.; Goh, V.T.; Yap, W.J.; Yap, T.T.V.; Ng, H. Identification of Road Surface Conditions using IoT Sensors and Machine Learning. In *Computational Science and Technology*; Springer: Singapore, 2019; pp. 259–268.
36. Sang, K.S.; Zhou, B.; Yang, P.; Yang, Z. Study of Group Route Optimization for IoT Enabled Urban Transportation Network. In Proceedings of the 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, UK, 21–23 June 2017; pp. 888–893.
37. Dogru, N.; Subasi, A. Traffic accident detection using random forest classifier. In Proceedings of the 2018 15th Learning and Technology Conference (L&T), Jeddah, Saudi Arabia, 25–26 February 2018; pp. 40–45.
38. Almeida, P.R.D.; Oliveira, L.S.; Britto, A.S.; Silva, E.J.; Koerich, A.L. PKLot—A robust dataset for parking lot classification. *Expert Syst. Appl.* **2015**, *42*, 4937–4949. [CrossRef]
39. Zhou, Z.H. *Ensemble Methods: Foundations and Algorithms*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2012.

40. Lienhart, R.; Maydt, J. An extended set of haar-like features for rapid object detection. In Proceedings of the International Conference on Image Processing, Rochester, NY, USA, 22–25 September 2002; Volume 1, p. I.
41. Behrisch, M.; Bieker, L.; Erdmann, J.; Krajzewicz, D. SUMO–simulation of urban mobility: An overview. In Proceedings of the SIMUL 2011, The Third International Conference on Advances in System Simulation, ThinkMind, Barcelona, Spain, 23–28 October 2011.
42. Friedman, N.; Geiger, D.; Goldszmidt, M. Bayesian network classifiers. *Mach. Learn.* **1997**, *29*, 131–137. [CrossRef]
43. Rabiner, L.R. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in Speech Recognition*; Elsevier: San Mateo, CA, USA, 1990; pp. 267–296.
44. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
45. Utgoff, P.E. Incremental induction of decision trees. *Mach. Learn.* **1989**, *4*, 161–186. [CrossRef]
46. Jain, A.K.; Murty, M.N.; Flynn, P.J. Data clustering: A review. *ACM Comput. Surv. (CSUR)* **1999**, *31*, 264–323. [CrossRef]
47. Jain, A.K. Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.* **2010**, *31*, 651–666. [CrossRef]
48. Werbos, P.J. Backpropagation through time: What it does and how to do it. *Proc. IEEE* **1990**, *78*, 1550–1560. [CrossRef]
49. El Naqa, I.; Murphy, M.J. What is machine learning? In *Machine Learning in Radiation Oncology*; Springer: Cham, Switzerland, 2015; pp. 3–11.
50. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef] [PubMed]
51. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS 2012), Stateline, NV, USA, 3–8 December 2012.
52. Daniels, D.J. Ground penetrating radar. *Encyclopedia RF Microw. Eng.* **2005**. [CrossRef]
53. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
54. Liu, G.; Bao, H.; Han, B. A Stacked Autoencoder-Based Deep Neural Network for Achieving Gearbox Fault Diagnosis. *Math. Probl. Eng.* **2018**, *2018*, 5105709. [CrossRef]
55. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
56. Aha, D.W.; Kibler, D.; Albert, M.K. Instance-based learning algorithms. *Mach. Learn.* **1991**, *6*, 37–66. [CrossRef]
57. Gower, J.; Ross, G. Non-probabilistic Classification; In *Advances in Data Science and Classification. Studies in Classification, Data Analysis, and Knowledge Organization*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 21–28, doi:10.1007/978-3-642-72253-0_3.
58. Distefano, S.; Merlino, G.; Puliafito, A.; Cerotti, D.; Dautov, R. Crowdsourcing and Stigmergic Approaches for (Swarm) Intelligent Transportation Systems. In Proceedings of the International Conference on Human Centered Computing (HCC 2017), Kazan, Russia, 7–9 August 2017.
59. Chang, I.C.; Tai, H.T.; Yeh, F.H.; Hsieh, D.L.; Chang, S.H. A VANET-Based A* Route Planning Algorithm for Travelling Time- and Energy-Efficient GPS Navigation App. *Int. J. Distrib. Sens. Netw.* **2013**, *9*, 794521. [CrossRef]
60. Tafidis, P.; Teixeira, J.; Bahmankhah, B.; Macedo, E.; Coelho, M.C.; Bandeira, J. Exploring crowdsourcing information to predict traffic-related impacts. In Proceedings of the 2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe), Milan, Italy, 6–9 June 2017.
61. Barth, D. The Bright Side of Sitting in Traffic: Crowdsourcing Road Congestion Data, 2009. Available online: https://googleblog.blogspot.com/2009/08/bright-side-of-sitting-in-traffic.html (accessed on 11 March 2019).
62. Fan, X.; Liu, J.; Wang, Z.; Jiang, Y.; Liu, X. Crowdsourced Road Navigation: Concept, Design, and Implementation. *IEEE Commun. Mag.* **2017**, *55*, 126–128. [CrossRef]
63. Saarika, P.; Sandhya, K.; Sudha, T. Smart transportation system using IoT. In Proceedings of the 2017 IEEE International Conference on Smart Technologies for Smart Nation (SmartTechCon), Bangalore, KA, India, 17–19 August 2017; pp. 1104–1107.

64. Gupta, A.; Kulkarni, S.; Jathar, V.; Sharma, V.; Jain, N. Smart Car Parking Management System Using IoT. *Am. J. Sci. Eng. Technol.* **2017**, *2*, 112.
65. Aydin, I.; Karakose, M.; Karakose, E. A navigation and reservation based smart parking platform using genetic optimization for smart cities. In Proceedings of the 2017 5th International Istanbul Smart Grid and Cities Congress and Fair (ICSG), Istanbul, Turkey, 19–21 April 2017; pp. 120–124.
66. Shi, J.; Jin, L.; Li, J.; Fang, Z. A smart parking system based on NB-IoT and third-party payment platform. In Proceedings of the 2017 17th International Symposium on Communications and Information Technologies (ISCIT), Cairns, QLD, Australia, 25–27 September 2017; pp. 1–5.
67. Rizvi, S.R.; Zehra, S.; Olariu, S. ASPIRE: An Agent-Oriented Smart Parking Recommendation System for Smart Cities. *IEEE Intell. Transp. Syst. Mag.* **2018**. [CrossRef]
68. Araujo, A.; Kalebe, R.; Girao, G.; Filho, I.; Goncalves, K.; Neto, B. Reliability analysis of an IoT-based smart parking application for smart cities. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017.
69. Jia, G.; Han, G.; Li, A.; Du, J. SSL: Smart Street Lamp Based on Fog Computing for Smarter Cities. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4995–5004. [CrossRef]
70. Kokilavani, M.; Malathi, A. Smart street lighting system using IoT. *Int. J. Adv. Res. Appl. Sci. Technol.* **2017**, *3*, 8–11.
71. Tripathy, A.K.; Mishra, A.K.; Das, T.K. Smart lighting: Intelligent and weather adaptive lighting in street lights using IOT. In Proceedings of the 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, India, 6–7 July 2017.
72. Celesti, A.; Galletta, A.; Carnevale, L.; Fazio, M.; Ĺay-Ekuakille, A.; Villari, M. An IoT Cloud System for Traffic Monitoring and Vehicular Accidents Prevention Based on Mobile Sensor Data Processing. *IEEE Sens. J.* **2018**, *18*, 4795–4802. [CrossRef]
73. Chowdhury, D.N.; Agarwal, N.; Laha, A.B.; Mukherjee, A. A Vehicle-to-Vehicle Communication System Using Iot Approach. In Proceedings of the 2018 IEEE Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; pp. 915–919.
74. Geetha, S.; Cicilia, D. IoT enabled intelligent bus transportation system. In Proceedings of the 2017 2nd International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 19–20 October 2017.