Multimedia Tools and Applications https://doi.org/10.1007/s11042-020-09714-8



TVD-MRDL: traffic violation detection system using  
MapReduce-based deep learning for large-scale data

Shiva Asadianfam 1 & Mahboubeh Shamsi 2 & Abdolreza Rasouli Kenari2

Received: 19 January 2020 /Revised: 13 August 2020 /Accepted: 25 August 2020 # Springer Science+Business Media, LLC, part of Springer Nature 2020



**Abstract**

Maintaining a fluid and safe traffic is a major challenge for human societies because of its social and economic impacts. Various technologies have considerably paved the way for the elimination of traffic problems and have been able to effectively detect drivers’ violations. However, the high volume of the real-time data collected from surveillance cameras and traffic sensors along with the data obtained from individuals have made the use of traditional methods ineffective. Therefore, using Hadoop for processing large-scale structured and unstructured data as well as multimedia data can be of great help. In this paper, the TVD-MRDL system based on the MapReduce techniques and deep learning was employed to discover effective solutions. The Distributed Deep Learning System was implemented to analyze traffic big data and to detect driver violations in Hadoop. The results indicated that more accurate monitoring automatically creates the power of deterrence and behavior change in drivers and it prevents drivers from committing unusual behaviors in society. So, if the offending driver is identified quickly after committing the violation and is punished with the appropriate punishment and dealt with decisively and without negligence, we will surely see a decrease in violations at the community level. Also, the efficiency of the TVD-MRDL performance increased by more than 75% as the number of data nodes increased.

Keywords MapReduce-based . Deep learning . Distributed processing . Drivers’ behavior detection . Hadoop . Unsafe behaviors

IMAGE

1 Department of Computer Engineering, Qom Branch, Islamic Azad University, Qom, Iran

2 Faculty of Electrical & Computer Engineering, Qom University of Technology, Qom, Iran

Abdolreza Rasouli Kenari rasouli@qut.ac.ir

Shiva Asadianfam sh\_asadianfam\_stu@qom-iau.ac.ir

\* Mahboubeh Shamsi shamsi@qut.ac.ir

Multimedia Tools and Applications

**1 Introduction**

Drivers, as human beings, commit violations through various behaviors and subsequently cause various kinds of damage. In fact, drivers’ risky behavior brings about unforeseen incidents and consequently irreparable damage. In general, potential risks pertain to uncontrolled conditions or activities that can cause damage. Identifying risky conditions and eliminating or controlling them enables us to avoid damage. The social costs of injuries deaths and casualties are too much. Mostly drivers are blamed for causing accidents and injuries [33]; Therefore, modifying drivers’ behavior is considered as one of the most important and challenging issues in transportation. In order to control drivers’ traffic violations, traffic policemen are present at the entry points to fine violating vehicles. In this method, due to the fact that human beings (policemen) are in charge of controlling violations via observation, error occurrence and misuse, including collusion, violations’ being unnoticed by the police, the absence of the police, traffic congestion at entry points, etc., are inevitable [29, 40, 41, 49].

In recent years, owing to the increase in the number of vehicles on streets and roads, more attention has been directed toward traffic control problems and the enforcement of traffic laws. In some countries, certain solutions and systems have been proposed to control urban traffic automatically [4, 31, 37]. Also, installing different cameras in high-traffic junctions and streets and providing efficient systems for the automatic detection of drivers’ risky behaviors seem to be essential. Given the significance of the issue, any study in this regard can be of great contribution to society. In a similar vein, in paper [48], a general system is proposed to track objects and identify incidents. Also, in paper [44], a monitoring system is propounded for object tracking.

Since video surveillance cameras generate fast, large, and varied data, the data provided by them can be deemed as big data. Big data is a term referring to datasets whose simultaneous and efficient management, control, and processing are beyond the capability of traditional software tools due to being too large or complex [15]. Hadoop is a popular technology for big data processing. Hadoop and the MapReduce techniques are utilized in various fields, such as medicine, meteorology, behavior detection, databases, and also in the implementation of various MapReduce-based machine-learning algorithms for big data analysis. This method can also be used to analyze traffic data derived from urban surveillance cameras. MapReduce is a programming model for large-scale data processing, which reduces processing time using distributed data processing. Therefore, this paper presents a MapReduce-based Deep Learning for Traffic Violation Detection (TVD-MRDL) using large-scale and high-volume images.

In this paper, the TVD-MRDL system, as a system of driver behavior detection, is proposed as a system aiming at accelerating the processing time of varied and big data sets. It is designed in such a way that it can detect risky behaviors of traffic violators in large-scale data in the Hadoop framework via the distribution technique. Using performance analysis, the TVD-MRDL technique can reduce the processing time by employing more slave-nodes. Here, a MapReduce space is suggested, which is able to analyze the data of traffic control centers and detect drivers’ violations (unsafe behavior) using a distributed architecture and a wide variety of data.

The TVD-MRDL system can make the automated control of traffic violations possible based on the distributed architecture and the MapReduce programming model. Furthermore, an approach is proposed to detect drivers’ violations based on MapReduce-based deep learning on a large scale. In fact, the distributed learning algorithm of violation detection requires a lot of time which linearly boosts with the increase in the Receptive Field. Providing this condition, the proposed approach seems to be suitable for big data learning. Generally speaking, the

IMAGE

Multimedia Tools and Applications

purpose of the TVD-MRDL system is to lessen errors considerably, increase speed, and keep maintenance costs very low, all of which make it a suitable technique in traffic control centers to detect the risky behaviors of drivers.

The rest of paper is organized as follows: In Section 2, the background knowledge required to understand the proposed approach is presented. In Section 3, the studies using the MapReduce technique in different fields of study are reviewed. In Section 4, the overall process of the TVD-MRDL system along with the used MapReduce technique and the process of deep learning are explained and the components and function of each stage are described. In Section 5, the adjustments made for the implementation of the proposed approach and the results of using it on a case of violating behavior are examined, and finally, in Section 6, an overall conclusion is drawn based on the findings.

**2 Background**

**2.1 Big data**

Big data analytics introduces new methods and technologies for collecting, storing, and analyzing unstructured data on a scalable basis [18, 46]. Big data may be obtained from industry, agriculture, traffic and transportation, medical care, public organizations, families, and so on. In the past 20 years, the volume of data in different fields has significantly grown. According to the International Data Institute, the volume of data created around the world was 1.8ZB (1021 bytes) in 2011 [21], and by 2020, it will have grown to more than 35ZB [28]. In a research report, Daugh Lanie, an analyst at Meta (currently named Gartner), defined the opportunities and challenges created by an increase in data via the 3 V model (increased volume -velocity - variety) [27]. The big data value chain can be divided into four stages: data production, data acquisition, data storage, and data analysis [15]. Big data production refers to the main source of big data, which includes the data collected in a variety of fields via different technologies. The big data acquisition stage involves data collection, data transfer to storage infrastructure, and data preprocessing. In the data storage stage, various mechanisms and certain types of programming model (like MapReduce) are employed. In the big data analysis stage, the methods related to big data analysis are employed [16].

**2.2 Hadoop and the MapReduce technique**

In the general structure of information in Hadoop, the information is broken by the Hadoop system and sent to multiple servers (nodes). The servers process or store the received information on the basis of its type, which can be either processing information or storage information. When the system plans to retrieve information, it receives it from different servers, assembles it, and displays it as the output. The advantage of Hadoop is its capability to back up information automatically. Each piece of information is stored in several parts (servers), and in case one of the servers is out of order, another server can take on responsibility and replace information. MapReduce is a software framework that provides a secure and scalable condition for the development of distributed applications. To be more precise, MapReduce includes the automatic parallelization of tasks, computational load and data balancing, the optimization of disk and network transfers, and the management of machine defects. Hadoop allows us to run applications on thousands of nodes with thousands of

Multimedia Tools and Applications

terabytes of data [53]. Hadoop consists of two major sections, namely MapReduce and HDFS (Hadoop Distribution File System). This system is actually designed to run on multiple servers. The MapReduce section and the HDFS section run on the main server and the secondary servers, respectively. The HDFS [47]is able to increase the rate of data transfer between nodes and enables the system to continue operating incessantly in case of the failure of one node. This approach actually reduces the risk of catastrophic system failure even when a significant number of nodes are deactivated. MapReduce is a software framework in which the applications are divided into smaller sections. Each of these sections (also called a section or block) can be run on each node in a cluster set of nodes [19]. In fact, MapReduce is a parallel programming model for processing data on clusters and consists of two main phases, namely, the Mapping phase and the Reduce phase. Figure 1 illustrates an overall diagram of the MapReduce model. In the Map phase, input data is divided into smaller sections. The Map function runs on every small part of the data (on the slave nodes). After the responses of the small sections are organized, in the Reduce phase, they are combined to form the desired output (via the Master node) [53].

## 2.3 Application Scenarios of Hadoop in Traffic Big Data and Its Advantages

The primary sources of transportation big data include fixed detection data (such as geomagnetic coils), mobile detection data (such as floating automotive data), GPS detection data, and smartphone detection data. By utilizing Hadoop big data technology, these data could be mined and analyzed, promoting the research and application of Hadoop big data technology in the transportation industry. Upon reviewing the literature, we have identified eight primary application scenarios that current research in this field has mainly focused on. These scenarios include transportation infrastructure monitoring, taxi [operation management](https://www.sciencedirect.com/topics/social-sciences/operations-management), travel feature analysis, traffic flow prediction, traffic event monitoring and status discrimination, transportation big data analysis platforms, license plate recognition, and shortest path. In the field of transport, Hadoop is used in a variety of application scenarios, including traffic flow analysis: by collecting vehicle GPS data, the vehicle flow on the road can be monitored in real time. Using Hadoop for data storage and analysis can help traffic management departments optimise signal light control, plan road construction, etc. to improve traffic efficiency. Path planning optimisation: Based on historical traffic data and real-time road condition information, data mining and analysis using Hadoop can provide drivers with the best path planning to avoid congested roads and save time and fuel costs. Traffic Accident Prediction: By analysing information such as vehicle speed and density in traffic data, combined with external factors such as weather and road conditions, a prediction model can be constructed using Hadoop to warn of possible traffic accidents in advance, which can help to reduce the rate of accidents. Parking management: Combining real-time parking space information and vehicle entry and exit records in the car park, using Hadoop for data analysis and processing, it can achieve intelligent allocation of parking spaces, optimise the parking process, and improve the utilisation rate of the car park. Public Transport Optimisation: By analysing data such as passenger travel records and real-time bus locations, and using Hadoop for real-time scheduling and optimisation, the operational efficiency of the public transport system can be improved, and passenger waiting time can be reduced. All these show the diverse application scenarios of Hadoop in the field of transport. Through the storage, processing and analysis of big data, it can provide smarter and more efficient solutions for traffic management, traffic planning, traffic safety and other aspects.

And the application scenarios of Hadoop in traffic big data also have the following advantages. Storage of large-scale data: traffic data usually contains a large number of vehicle tracks, traffic flow and other information, Hadoop's distributed file system (HDFS) can effectively store these massive data, and support high reliability and high availability. Parallel processing: Hadoop's MapReduce framework can divide the data into small pieces and process them in parallel in the cluster to speed up data processing, which is suitable for traffic data analysis scenarios that require a large number of calculations. Real-time data processing: Combined with real-time data processing frameworks (e.g. Apache Storm, Apache Flink), Hadoop can achieve real-time processing and analysis of traffic data, helping to monitor traffic conditions, predict traffic congestion and so on. Data Mining and Pattern Recognition: With Hadoop's machine learning libraries (e.g. Apache Mahout) and data mining tools, traffic data can be mined and analysed to discover traffic patterns, optimise traffic management strategies, etc. Scalability and elasticity: Hadoop clusters can be horizontally scaled as needed to support the processing of growing volumes of traffic data, while having automatic fault tolerance and fault recovery mechanisms to guarantee the stability of data processing. Cost-effectiveness: Hadoop is open-source software with relatively low deployment and maintenance costs, suitable for processing traffic big data scenarios, which can reduce the cost of data processing and analysis. In summary, the application of Hadoop in traffic big data has the advantages of storing large-scale data, parallel processing, real-time data processing, data mining and pattern recognition, scalability and elasticity, and cost-effectiveness, which provides a powerful support for data analysis and application in the field of transport.

**2.4 Deep learning**

Deep learning, an area associated with machine learning, is a set of algorithms in which high-order abstract concepts are modeled through learning at different levels and layers [22]. Deep learning is actually derived from the way human brains function and it requires advanced tools, such as powerful graphics cards, for complicated calculations and large volumes of big data. The low volume of data in this algorithm yields weaker results and performance; in other words, deep learning involves using neural networks for a large amount of data. The most important advantage of this learning method is its representational learning which is an approach extracting features from low-order input automatically [45]. Deep learning is widely employed in various fields, including computer vision, such as image classification, object detection, semantic segmentation, image retrieval, and human pose estimation, which are the major factors in understanding images.

In general, deep learning methods are divided into four different categories based on the base method from which they are derived: Convolutional Neural Networks, Restricted Boltzmann Machines (RBMS), Autoencoders, and Sparse Coding. For each of these categories, certain architectures are proposed. For example, in CNNs, there are Alexnet, SPP, VGGnet, Clarfai and Googlenet architectures which are often different in terms of the number of their convolutional and fully-connected layers. Learning methods vary based on their features, such as Generalization, Unsupervised Learning, Feature Learning, Real-Time

Deep learning is the process of learning the intrinsic patterns and levels of representation of sample data, and the information gained from these learning processes can be of great help in the interpretation of data such as text, images, and sounds. Its ultimate goal is to enable machines to have analytical learning capabilities like a human being and be able to recognise data such as text, images and sounds. Deep learning is a complex machine learning algorithm that has achieved results in speech and image recognition that far exceed previous related techniques. [1]

Deep learning has achieved much in search technology, data mining, machine learning, machine translation, natural language processing, multimedia learning, speech, recommendation and personalisation techniques, and other related fields. Deep learning has enabled machines to mimic human activities such as seeing, hearing, and thinking, and has solved many complex pattern recognition challenges, leading to significant advances in AI-related technologies.

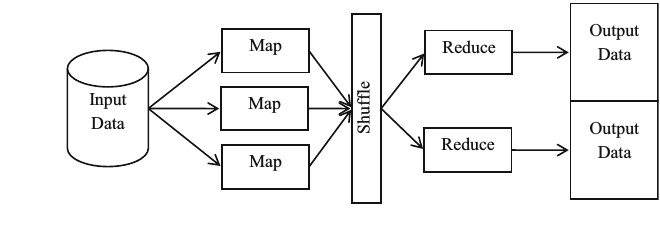


Fig. 1 The MapReduce Programming Model

IMAGE

Multimedia Tools and Applications

Training, Real-Time Prediction, Biological understanding, Theoretical justification, Invariance, and Small training set [38].

**3 Optimization of Hadoop**

Hadoop is a data processing tool that handles data placement. However, its current method for data placement focuses primarily on balancing data distribution and doesn't take into account the relationships between different datasets. Consequently, all HDFS data is placed based on the workload needs of the Hadoop cluster. This can result in a large amount of [data transfer](https://www.sciencedirect.com/topics/engineering/data-transfer) when MapReduce computations are conducted, leading to higher I/O expenses. To improve processing efficiency, several optimization methodologies have emerged. One such approach is CoHadoop, an optimization mechanism developed by IBM that assigns data blocks based on the needs of the application. However, before submitting large data to HDFS, CoHadoop requires partitioning of the data according to the application's requirements, which incurs a significant processing cost. Another tool that has gained popularity for data processing is Spark, a general-purpose parallel computing framework similar to Hadoop MapReduce. Spark also uses the MapReduce algorithm, but with some notable differences. Unlike Hadoop MapReduce, Spark can store the intermediate output of a job in memory, eliminating the need to read and write to HDFS. As a result, Spark is particularly well-suited for data mining and [machine learning algorithms](https://www.sciencedirect.com/topics/engineering/machine-learning-algorithm) that require iterative processing. [Table 4](https://www.sciencedirect.com/science/article/pii/S2095756423001009?ref=pdf_download&fr=RR-2&rr=863263e72f1e5ddc" \l "tbl4) provides a comparison between Hadoop and Spark in terms of various features such as fault tolerance, scalability, language support, visualization, real-time analysis, machine learning, and SQL support. This comparison highlights the distinct application scopes of each tool.

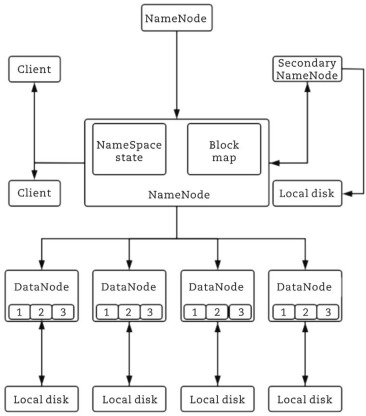
Table 4. Comparison table between Hadoop and Spark.

| Empty Cell | **Hadoop** | **Spark** |
| --- | --- | --- |
| Fault tolerance | Failure does not require restarting the application. | Recover lost work without additional code or configuration. |
| Expandability | Has strong scalability potential and has been used in production on tens of thousands of nodes. | Highly scalable with the ability to continuously add n nodes to the cluster. |
| Language support | Mainly supports Java, others are C, C++, Ruby, Groovy, Perl, Python. | Support for Java, Scala, Python and R. |
| Visualization | Data visualization is zoomdata's ability to connect directly to HDFS as well as SQL-on-Hadoop technology. | Through a web interface for job submission and execution, or integrated into Apache Zeppelin. |
| Real-time analysis | MapReduce cannot handle real-time data because it is designed to perform batch processing on large amounts of data. | Real-time data can be processed. |
| SQL support | Users are able to run SQL queries using Apache Hive. | Users are able to run SQL queries using Spark-SQL. |
| Machine learning | Machine learning tools like Apache Mahout are needed. | There is a set of machine learning MLlib. |

There have been numerous studies published on Hadoop that focus on its computational model optimization and the optimization of Hadoop when used in conjunction with Spark. This section will delve into these two key issues.

### 3.1Distributed file system

Hadoop distributed file system (HDFS) is an open-source implementation of Google file system (GFS). It's designed to provide high-throughput data access and is well-suited for storing and processing parallel data on a large scale. The fundamental structure of HDFS is illustrated in [Fig. 3](https://www.sciencedirect.com/science/article/pii/S2095756423001009?ref=pdf_download&fr=RR-2&rr=863263e72f1e5ddc" \l "fig3). It uses a master-slave architecture, with the HDFS cluster consisting of a single metadata node called the NameNode, multiple data nodes called DataNodes, and a secondary node called the secondary Namenode ([Kim et al., 2015](https://www.sciencedirect.com/science/article/pii/S2095756423001009?ref=pdf_download&fr=RR-2&rr=863263e72f1e5ddc" \l "bib25)).



In HDFS, the NameNode serves as the master node that oversees the operation of each slave node. Specifically, it's responsible for keeping track of changes to the NameSpace and managing the NameSpace itself. HDFS data operations follow a “write once, read many times” model. When a file is stored in HDFS, it's typically divided into multiple 64 MB data blocks, each of which is stored on a separate DataNode. The NameNode plays a critical role in this process by managing and mapping the data blocks to the corresponding DataNodes.

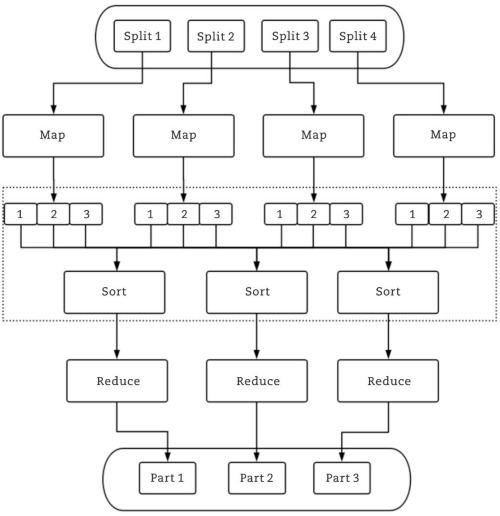
When a client wants to read or write a file, it communicates with the appropriate DataNode(s) to perform the requested operation. The DataNode processes the client's requests and carries out actions such as creating or deleting data blocks based on the instructions provided by the NameNode. To access a file in HDFS, a client must first retrieve the location of each data block within the file from the NameNode. It can then retrieve the corresponding data from the appropriate DataNodes. This two-step process ensures that data is retrieved efficiently and accurately.

Overall, HDFS's distributed architecture and data handling mechanisms make it a powerful tool for managing large volumes of data in parallel and distributed environments.

### 3.2Distributed computing framework

The MapReduce framework, initially proposed by Google in 2004, is a programming model for distributed parallel computing. It enables the processing of vast amounts of data, overcoming the inefficiencies of traditional computing methods. The MapReduce program consists of two main phases: the Map phase and the Reduce phase.

The Map function only accepts input in the <key, value> format, and Hadoop utilizes the InputFormat() method to automatically generate input data as <key, value> pairs for processing by the Map function. The key value denotes the byte offset of each data record in the data slice, and the value represents the content of each row. Similarly, the Reduce function also has input and output in the form of <key, value>. It takes the output of the Map function as input and operates on it. The core idea is to divide tasks into smaller, more manageable portions using the Map process, and then combining the results through the Reduce process. This approach enhances the efficiency of processing large-scale data by leveraging the Hadoop data platform servers to process vast amounts of data in parallel through distributed computing. [Fig. 4](https://www.sciencedirect.com/science/article/pii/S2095756423001009?ref=pdf_download&fr=RR-2&rr=863263e72f1e5ddc" \l "fig4) depicts the typical execution process of a MapReduce program ([Kang et al., 2012](https://www.sciencedirect.com/science/article/pii/S2095756423001009?ref=pdf_download&fr=RR-2&rr=863263e72f1e5ddc" \l "bib24)).



The MapReduce execution process includes the following parts.

* (1)

Dividing the massive input data into smaller portions and distributing them to different machines for processing.

* (2)

The Map task worker parses the input data into <key, value> pairs, and the user-defined Map function transforms them into intermediate <key, value> pair.

* (3)

Sorting and aggregating the intermediate <key, value> pairs based on their key values.

* (4)

Distributing different key values and their corresponding value sets to various machines to execute the Reduce operation.

* (5)

Generating the output of the Reduce operation.

IMAGE

Multimedia Tools and Applications

**4 The proposed approach**

The present study mainly focuses on using MapReduce technique to detect drivers’ behaviors. The TVD-MRDL system is designed in such a way that it would be able to detect risky behaviors of traffic violators in big data using the Hadoop framework and the distribution technique. The TVD-MRDL system with a MapReduce space uses deep learning for a wide variety of data. This system is able to analyze the data obtained from traffic control centers and the police descriptions in a distributed manner and it is capable of identifying drivers’ violations.

**4.1 Pseudo-code and flowchart**

The flowchart related to the proposed method (TVD-MRDL) is illustrated in Fig. 2. This system involves analyzing structured and unstructured data as well as analyzing multimedia data (including visual files and videos). Structured data refer to the data that have a predictable and regular format, including records, fields, keys, and indexes. Unstructured data refer to the unpredictable data with a structure which is unidentifiable by computers. Access to unstructured data is difficult, especially when long strings of data are required to be searched sequentially (i.e., parsing) in order to derive a data unit from them. Various types of unstructured data are available, and it can be stated that the most common type is images. In the analysis of police data, initially the police take pictures of drivers’ risky behaviors and then label them using certain interface. These labels reveal the types of drivers’ behaviors and include a full description of them. For example, if the police find a vehicle stopping next to the

IMAGE

Multimedia Tools and Applications

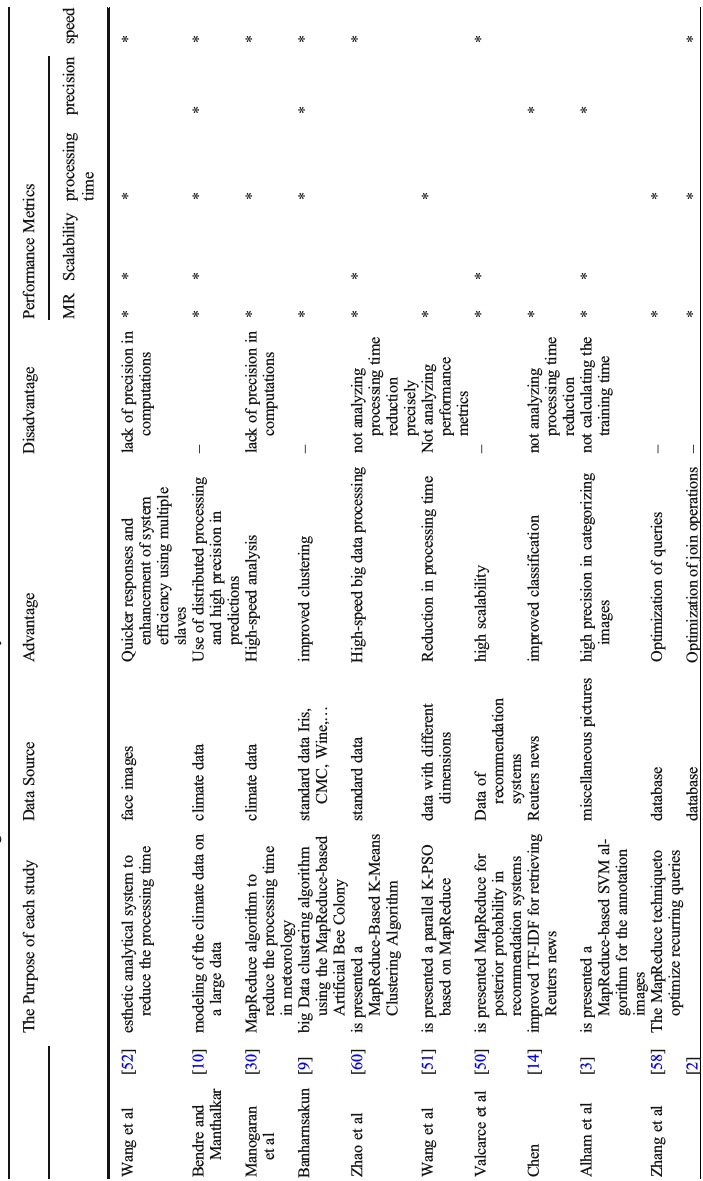


Table1   
A brief   
review of the   
relevant   
studies on   
big   
data in various   
fields of study

IMAGE

Multimedia Tools and Applications

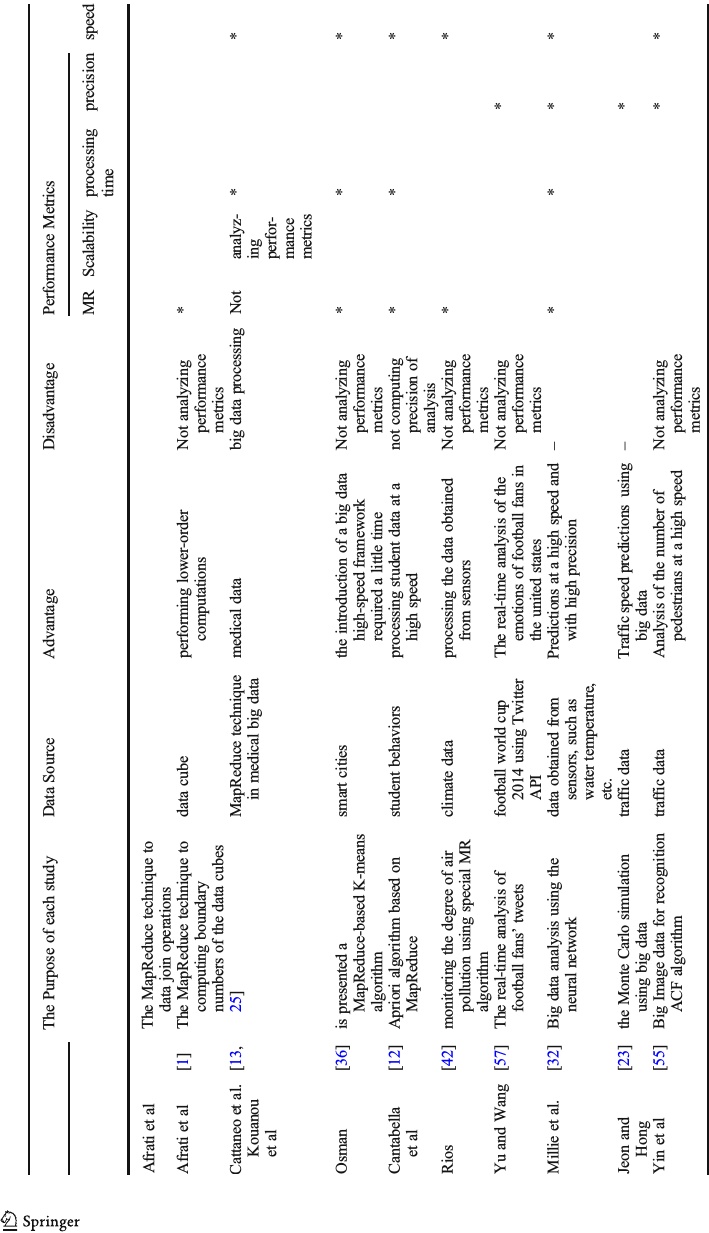


Table1   
(continued)

Multimedia Tools and Applications

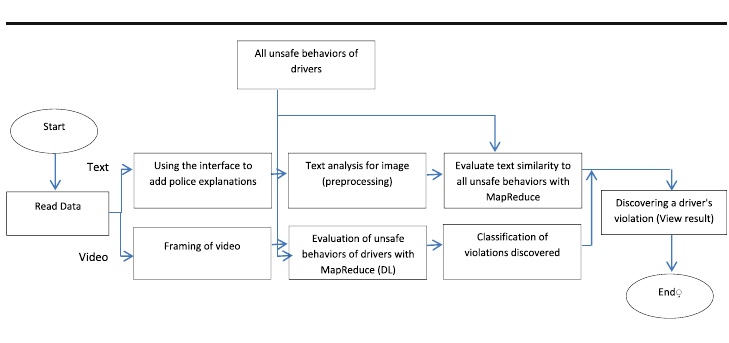


Fig. 2 The Flowchart of TVD-MRDL

“No Stopping” sign, they immediately take a picture of it and label that picture as “the car is parked along the NO-STOPPING traffic sign on the street”.

In the next step, preprocessing operations are performed to extract useful data. Preprocessing operations include omitting useless data (redundant letters), omitting repetitive data (the presence of two similar behaviors in the police descriptions), and filling the lost data (which are not fully described by the police due to being difficult to understand). These operations make the police descriptions be displayed in a usable form. After the preprocessing operations are performed, the useful extracted data are compared with all the pre-defined driving behaviors in Traffic Department (driving violations, parking violations, etc.) collected by the specialists in the field of driving (the approved Table of the government or Traffic Department). Evaluating and analyzing the similarities of the police descriptions to all the unsafe behaviors is done by the MapReduce function in the Similarity-Description and Behaviors algorithm.

As the Similarity-Description and Behaviors algorithm indicate, in order to detect the similarity, TF-IDF, as the similarity criterion [59], is used,as shown in Eq. (1). In Eq. (1), d1and d2 are the vectors for police descriptions and for each of the predefined driving behaviors, respectively, and θis the degree of their similarities. In line 2, the algorithm begins with two input parameters, i.e., all the predefined driving behaviors and the preprocessed police descriptions. In the Map function, in lines 3 to 8, the frequency vector of the police statements and the predefined behavioral data are extracted and in lines 9 to 16, their similarity is calculated using the cosine criterion in Eq. 1. In line 17, the similarity between the police descriptions and each of the predefined unsafe driving behaviors is determined and shown as output. The Reduce function, in the first line, receives all the calculated valuesalong with the degree of their similarities and in lines 3 to 7, calculates the largest value, and ultimately in line 8, provides the output of the Reduce function as the type of the driver’s behavior and violation. In this way, human error, such as visual errors and the exact diagnosis of behavior types, is realized (Fig. 3).

FORMULA

As shown in Fig. 2, in the next phase, the information gathered by the video surveillance cameras and sent to the traffic control centers is first converted to a picture or frame, and then,

IMAGE

Multimedia Tools and Applications

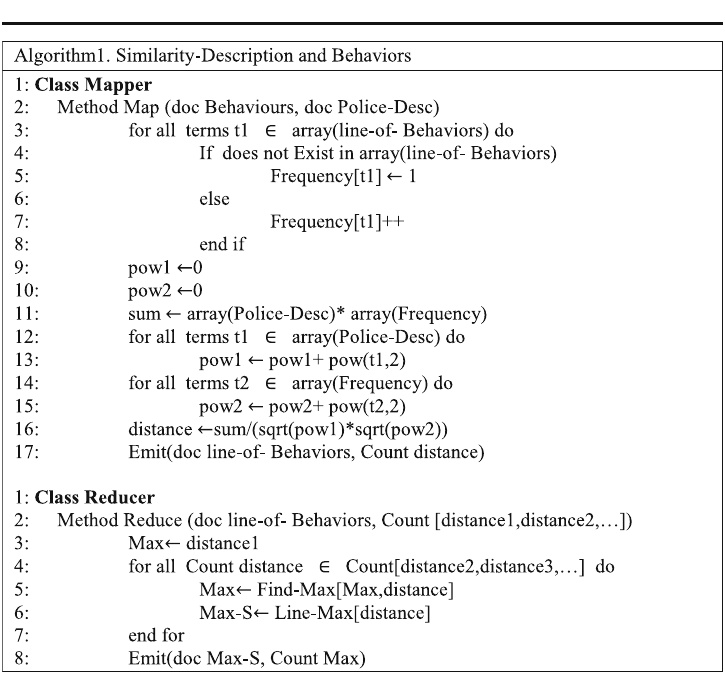


Fig. 3 The Pseudo-code of the MapReduce function to determine the similarity between police descriptions and the predefined driving behaviors.

the driver behavior is detected using the MapReduce function. The Pseudo-code related to the mechanized diagnosis of the driver behavior using videos is explained in the Mechanized diagnosis of driver behavior algorithm (Fig. 4).

As seen in the mechanized diagnosis of driver behavior algorithm, in line 2, the Map function receives the image or frame of the captured video, and the length and width of the image are extracted in lines 3 and 4, respectively. In line 5, using the length and width of the initial image, an empty frame of the same size is generated. In line 6, the preprocessing operations, such as image graying, are applied to the image and the created gray image is placed in the empty frame. Then, in line 7 of the Map function, the street area is extracted (i.e., lane detection) using the method employed in papers [35, 54]. In lines 8 to12, to determine the type of traffic signs, the traffic sign detection function is called using the method utilized in papers [11, 17, 24, 26]. In lines 14 to 21, the traffic lights and their color are identified using the functions introduced in papers [20, 43] and the color is stored in a variable. Afterwards, in line 22, crosswalks are detected in the street area extracted in line 7 using the method proposed in paper [39]. In line 24, if the traffic light is red or yellow, the vehicle detection function in the area of crosswalks is called using the method introduced in paper [24]. Then, by using several conditions written in the next lines of the Mechanized diagnosis of driver behavior algorithm, it is possible to identify the violation type for each behavior. For example, in line 27, if the

IMAGE

Multimedia Tools and Applications

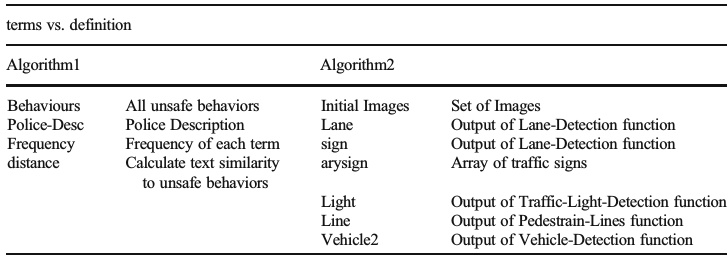
vehicle detection function within the specified range identifies a vehicle and the color of the traffic light is red, a violation, labeled as “red-light crossing”, is stored in a variable. This case is an example of a “red-light crossing” violation. In lines 30 to 34 of this algorithm, the vehicle detection function in the street area is called. In lines 35 and 36, if the type of the extracted traffic signs is “no parking” or anything related to parking, a violation labeled “parking” is stored in a variable. In the remaining lines of the Map function (lines 37 to 44), given the aforementioned conditions, two other violations, namely, “no entry” and “no stopping” are also identified. It’s worth noting that just a few prevalent violations are mentioned. Finally, all the functions extracted from the called functions are used as the input of the Reduce function.

The obtained results from each function in the Map function, including the last extracted image, the type of traffic signs, the color of the traffic light, and the violation type (red-light crossing, no parking, no entry, no stopping, etc.), are used as the input (line 2) of the Reduce function. Then, via several sequential conditions in lines 3 to 13, the outputs of different functions are compared with each other, and ultimately the violation type and the driver behavior are determined for the generated frame. For instance, in line 3, if a violation associated with “traffic lights” is detected in the frame under analysis, the output of the Reduce function is the final extracted image along with the color of the traffic light and the type of the committed violation. In line 6, if a violation related to “parking” is done in the frame under analysis, the output of the Reduce function is the final extracted image along with the type of the traffic signs and the violation type, i.e., “parking in the area of traffic signs.” In line 9, if a violation pertaining to “no entry” is committed in the frame, the output of the Reduce function is the final extracted image along with the type of the traffic signs and the violation type, i.e., “no entry in the area of traffic signs”. Similarly, in line 12, the violation of “no stopping” along with the image in which this violation is committed is shown as the output. Definition of terms used in Algorithm1 and Algorithm2 are listed in Table 2.

**4.2 Designing the TVD-MRDL system**

The overall process of MapReduce traffic violation detection is depicted in detail in Fig. 5. In Fig. 5(a), the file associated with the approved driving violation Table and the police descriptions are broken down on various computers (Slave-Nodes). In the Map phase, the Map function applies to all the input data. Then, the similarity of the police descriptions to the predetermined driving behavior Table is calculated. In the Reduce phase, the results obtained from different nodes are combined, which shows the degree of their similarity. In Fig. 5(b),

Table 2 Definition of terms used in Algorithm1 & Algorithm2



IMAGE

Multimedia Tools and Applications

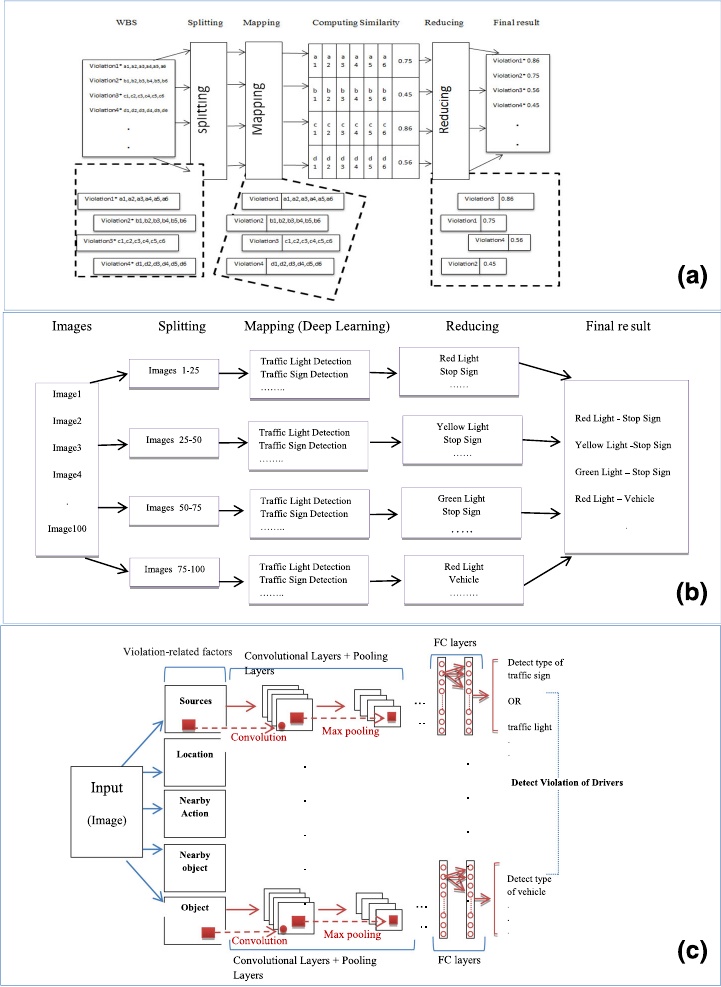


Fig. 5 The overall MapReduce processes of TVD-MRDL. (**a**) Police Descriptions (**b**) Images captured by the traffic control centers (**c**) deep learning

initially, the images obtained from the traffic control centers are divided on the slave nodes. Each slave node analyzes the image independently. Therefore, in the Map phase, in each slave node, the functions of traffic sign detection, traffic lights, street areas, vehicles, etc. are run using deep learning. In the Reduce phase, different outputs of each of these systems for the

IMAGE

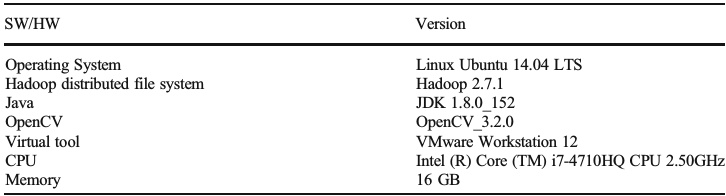
Multimedia Tools and Applications

input images are combined and a label, as the violation found in the image, is introduced. There are a number of deep learning-related methods; however, in this paper, the Convolutional Neural Networks are used. In general terms, a CNN consists of three main layers: the convolutional layer, the pooling layer, and the fully-connected layer. In each neural network, there are two stages, namely, the Feed-Forward stage and the Back-Propagation stage. The Back-Propagation stage is for training and Feed-Forward stage is for prediction. The input image is fed into the network, which is done via the dot product between the input and the parameters of each neuron, and finally, the convolutional operations are done in each layer. In the proposed structure, a convolutional network is shared among all the pixels of the input image. Its weight sharing significantly reduces the number of the trainable free parameters of the network, and as a result, increases generalizability. Then, the network output is calculated. Figure 5(c) illustrates the overall architecture of the deep convolutional neural networks for driver violation detection in different layers. First, the main image is divided into several images including the predefined factors of driving violation (the resources, location, nearby action, nearby object, and vehicle), and then based on the layered training data, each of the deep nervous networks is trained. Finally, each of them generates a multilayer network with convolutional and connected layers. In this study, the convolutional neural networks are of VGGnet type consisting of 13 to 15 convolutional layers and three fully-connected layers. Its convolutional layers and pooling layers are located alternately, and after these layers, there are three fully-connected layers. This provides a thorough incrementally deep evaluation of the network. Each of the deep convolutional neural networks learns the features in the driver violation images, such as traffic signs, traffic lights, street areas, vehicles, etc., and extracts the required features for categorizing the images. The combination of the output of each Convolutional Neural Network is considered as the driver violation type.

**5 The experiment and the results**

An elaborate set of experimentation takes place for the investigation of the improved performance of the projected model. In this experimental study, the proposed approach is implemented via Java language along with OpenCV library. The tested computer system is implemented in the VMware environment using two clusters. The first cluster has only one node and the second cluster contain a master node and seven slave nodes. The Linux Ubuntu operating system is installed and run on all the nodes. Software and hardware specifications of the proposed system are shown in Table 3. To evaluate the performance of TVD-MRDL, two training datasets, including unstructured textual data and multimedia data analysis (such as

Table 3 Software and hardware specifications



IMAGE

Multimedia Tools and Applications

visual files and videos) are employed. In the first phase, a series of textual data collected by the police are used. In the second phase, the data gathered from the surveillance cameras and sent to the traffic control center are used (Table 4).

To consider various criteria in analyzing the results of TVD-MRDL, four scenarios are used as follows: performance and efficiency, scalability, accuracy of error detection, and communication overhead. In what follows, these scenarios are discussed in detail.

**5.1 Performance**

To compare the performance of TVD-MRDL, two computational experiments were carried out. In these experiments, the sequential program with the Hadoop cluster in stand-alone mode was investigated and the performance of the TVD-MRDL, sequential program, and traditional system were compared. The se comparisons are explained separately in the following sections.

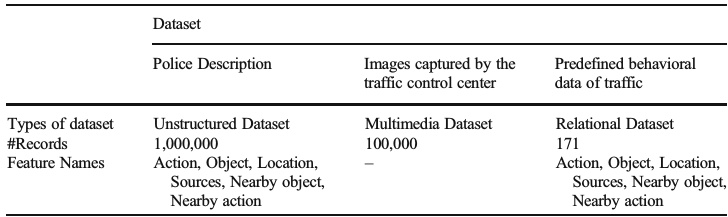
5.1.1 The comparison between the performance of the sequential program and Hadoop in stand-alone mode

To compare the performance of TVD-MRDL in the sequential program without the MapReduce technique and in Hadoop in stand-alone mode, the processing time (CPU time) of a number of police descriptions and the images of the surveillance cameras were examined in the range of 10,000 to 100,000 and then 1000 to 10,000. The processing time was calculated from the time the police descriptions and the images were received to the time the output of TVD-MRDL was obtained. The results are shown in Figs. 6 and 7.

In Fig. 6, the blue line and the red line, as the processing time of the sequential program and that of the Hadoop cluster in stand-alone mode, respectively, are used to analyze a different number of unstructured data (police descriptions). Clearly, Hadoop in stand-alone mode requires more processing time due to high overhead. In addition, the difference in their processing time increases as the number of textual data (police descriptions) boosts. For example, the processing time of 10,000 textual data using Hadoop in stand-alone mode is 2400 units while the processing time of 100,000 data increases to 23,000 units. Therefore, the performance of the sequential program is much better than Hadoop in stand-alone mode since in Hadoop in stand-alone mode, there is overhead when the data are loaded into the Hadoop cluster.

In Fig. 7, the blue line and the red line, as the processing time of the sequential program and that of Hadoop in stand-alone mode, are employed to analyze a different number of

Table 4 Dataset descriptions



IMAGE

Multimedia Tools and Applications

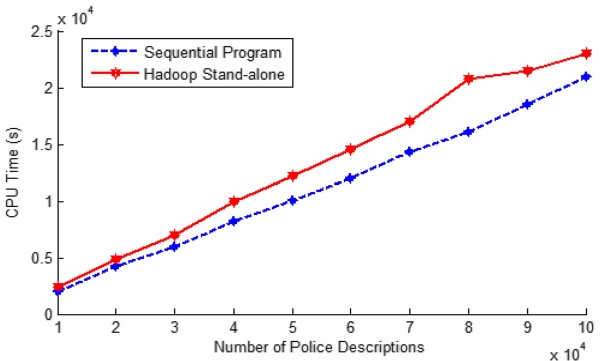


Fig. 6 The processing time of the police descriptions using TVD-MRDL in the sequential program and Hadoop in Stand-alone mode

unstructured data (the surveillance camera images). Like the analysis of the processing time for the unstructured data (textual data), the processing time increases as the number of multimedia data (the images) augments. For example, in the processing of 100 images via Hadoop in stand-alone mode, the processing time is 3300 units while in the processing of 10,000 images, it increases to 33,000 units. Therefore, regarding the textual data, the efficiency of the driver behavior detection system using Hadoop in stand-alone mode reduces due to excessive overhead.

**5.2 Scalability**

In order to estimate the scalability of TVD-MRDL, four computational tests have been performed, which are separately explained in detail in the following sections.

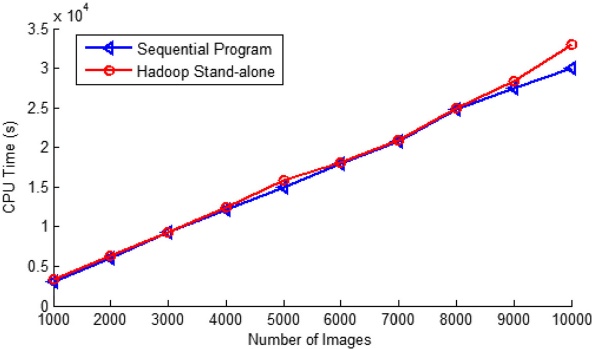


Fig. 7 The processing time using TVD-MRDL for the surveillance camera images in the sequential program and Hadoop in stand-alone mode

IMAGE

Multimedia Tools and Applications

**5.2.1 The comparison of the efficiency and scalability of the Hadoop cluster with one master node and several slave nodes**

The purpose of the first evaluation is to compare the scalability of implementing the TVD- MRDL algorithms by increasing the number of nodes. During these experiments, two datasets shown in Table 4 are processed using a number of nodes.

Based on the results presented in Fig. 8, apart from the single node cluster (Hadoop in stand-alone mode), the system has near-linear scalability. However, in the analysis of the police descriptions, compared to that of the Hadoop cluster with the one slave node, the Hadoop cluster with seven slave nodes leads to an almost 60.87% reduction in the processing time. The reason for this improvement is that in the MapReduce program, the system automatically performs calculations among large clusters of machines in parallel, manages system failures, and schedules within-system communication, and in this way, it makes the use of network and disks easier and more effective. Thus, through the balanced distribution of computing load across nodes, the parallel execution is performed without interference and the processing time decreases. Besides, in the analysis of the surveillance camera images, the processing time reduces by70%. As stated in Section 5.1, in comparison with the sequential implementation, a cluster with one node reveals poorer performance, which is due to Hadoop overhead.

**5.2.2 Scalability regarding data volume**

This scenario is designed to estimate how the system scales change by increasing the volume of data. To this end, the entire cluster is used to process a number of police descriptions related to the violations and the relevant images of the traffic control centers. The results are presented in Fig. 9. The processing time is the same as that for a small data set (low-volume data) (because of cluster overloads). As expected, the processing time for the high-volume datasets (a greater number of data) is linear since the image datasets provided by the traffic control center are independently processed.

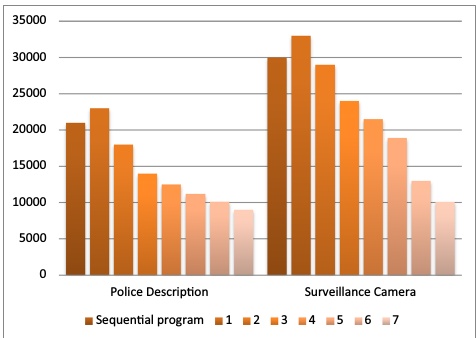


Fig. 8 The processing time using TVD-MRDL with regard to the number of nodes

IMAGE

Multimedia Tools and Applications

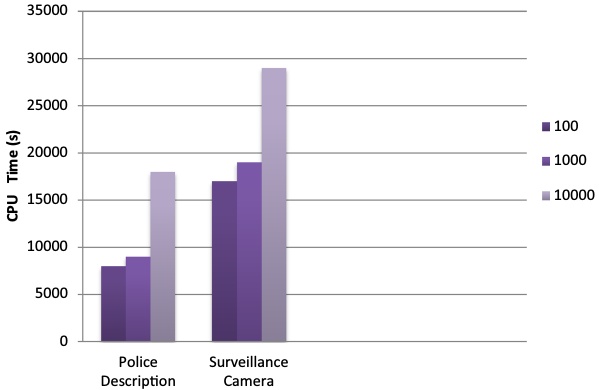


Fig. 9 The processing time using TVD-MRDL with regard to the number of datasets

**5.2.3 The comparison of the scalability of TVD-MRDL and the traditional system**

The purpose of this scenario is to examine the scalability of TVD-MRDL, the sequential program, and the traditional system for unsafe behavior detection considering the size of the data and the number of different slave nodes. To this end, the number of slave nodes varies from one to seven, and the processing time is computed for 2000, 4000, 6000, 8000 and 10,000 data derived from the police descriptions and the images of the cameras (Job). The results are shown in Fig. 10.

As seen in Fig. 10, the blue bars represent the processing time of the police descriptions and the red bars represent the processing time of the images obtained from the surveillance cameras. In this diagram, the processing time of three types of systems, called TS (traditional system), SP (sequential program) and MR (TVD-MRDL), with different numbers of slave nodes are compared. As observed in this figure, for a given amount of data, the processing time in the traditional system and the sequential program with different slave nodes is the same. For example, for 4000 textual and visual data with different numbers of slave nodes (varying from one to seven), the processing time of the traditional system is similar to that of the sequential program. The reason is that there is just one node in the sequential and traditional systems. However, the processing time differs for various jobs when different numbers of slave nodes are used; in other words, in each graph, as the number of slave nodes increases with regard to the volume of the data, the processing time decreases.

**5.3 Cost of Hadoop clusters**

**5.3.1 The comparison between the cost of resources of TVD-MRDL and the sequential program**

This scenario is designed to estimate how much cost of resources might be achieved in sequential program and TVD-MRDL during the analysis with virtual machines has been used

IMAGE

Multimedia Tools and Applications

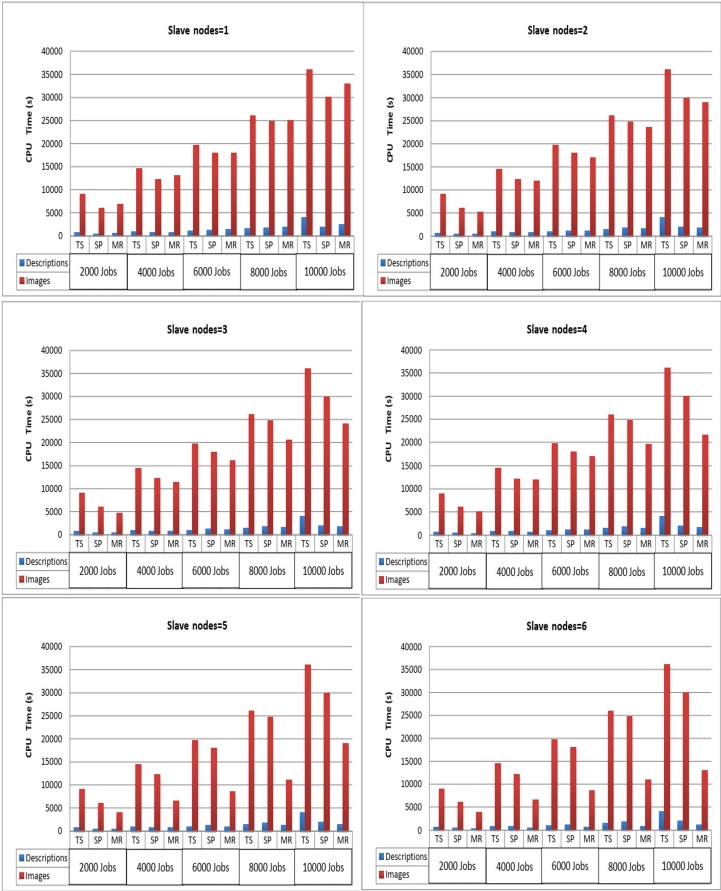


Fig. 10 The scalability of TVD-MRDL, sequential program and traditional system for unsafe behavior detection in terms of the volume of data and different numbers of slave nodes

to build up Hadoop clusters. It was also important in this analysis to compare the same amount and the same strong virtual machines within the Hadoop cluster. We used Hadoop version 2.7.1 on Linux Ubuntu 14.04 LTS, with the weakest category of VMware Workstation. This cluster has number of slave nodes (from 1 to 7), each with 7 core, 2 GB of RAM, 3 hard drives. This cluster has an estimated cost of about 800$ (Memory) and 1200$ (Hardware). Figure 11

shows the estimated costs of the Hadoop cluster consisting of slave nodes (from 1 to 7) and sequential program.

IMAGE

Multimedia Tools and Applications

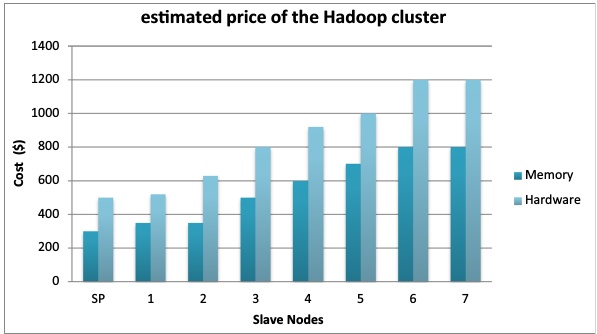


Fig. 11 Estimated costs of the Hadoop cluster

**5.3.2 Scale-up time of Hadoop slave nodes**

Our presented scenario is to scale up a cluster, which initially has one Hadoop Slave nodes, to seven Slave nodes. We examined the amount of time that elapsed between the fired command and the fully functional structure of the enlarged cluster. The experiments were repeated five times, and the final result is the average scale-up time to the given number of nodes (from 1 to 7). Figure 12 shows the diagram of the results, i.e. a kind of elasticity of the Hadoop cluster. The horizontal axis represents the number of nodes in the enlarged cluster and the vertical axis represents the elapsed time period during the fully functional state of the given Hadoop node. The current performance tests do not cover the effects of changes in the various parameters such as the number of Map or Reduce tasks. More than 190 parameters are available to control the behavior of a MapReduce job in Hadoop. As a conservative estimate, the settings of more than 25 of these parameters can have a significant impact on job performance.

**5.4 The comparison of the accuracy, precision, and recall of TVD-MRDL and traditional system in terms of data volume**

The purpose of this scenario is to explore the criteria for the detection of unsafe behaviors in different volumes of data in TVD-MRDL, sequential program (SP), and traditional system (TS). To explore the accuracy, precision, and recall of the proposed approach and compare them with those of the traditional system, different numbers of police descriptions and images of 2000, 5000, and 10,000 data (Job) are shown in Fig. 13.

In Fig. 13, the blue, red, and green bars represent the criteria of accuracy, precision, and recall, respectively. In this diagram, the traditional system is compared to the sequential program and MR program (TVD-MRDL). It is quite obvious that the values of these criteria in the sequential program and MR are the same due to the mechanization of the system. Moreover, the difference between the sequential and Programs is only related to processing time and the reduction of processing time. As seen in this figure, the values of all three criteria, namely, Accuracy, Precision, and Recall, in the MR system are far higher than those of the traditional ones.

IMAGE

Multimedia Tools and Applications

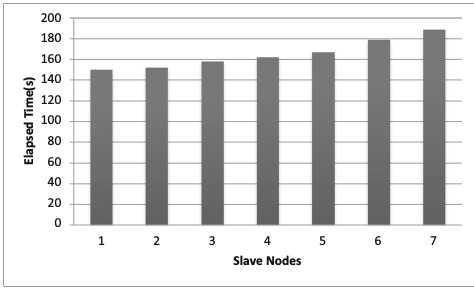


Fig. 12 Scale-up time of Hadoop slave nodes

**5.5 Communication overhead with regard to an increase in the number of nodes in Hadoop**

When the Hadoop is installed, SSH activation makes it possible for the Hadoop to link different nodes using RCP without any passwords. This abstraction in the TCP protocol is officially called the client protocol and the data node protocol. The data nodes (slave) send heartbeats (every three seconds) to the name of the node to make it realize that it is still active. As the number of nodes is big, approximately more slave and master nodes communicate with each other.

Increasing the number of nodes in a large volume of data can improve the performance of the system, which is deemed as its main advantage. However, the large number of nodes can increase communication costs (communication overhead time) and there is the probability of the failure of a number of slave nodes, which has a profound effect on system performance.

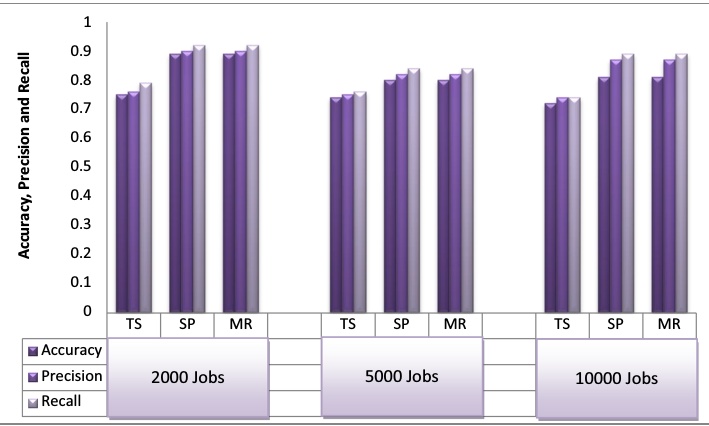


Fig. 13 Accuracy, Precision, and Recall of TVD-MRDL, sequential program, and traditional system for unsafe behavior detection in terms of data volume

IMAGE

Multimedia Tools and Applications

Figure 14 depicts the communication time (communication overhead-delay) and computational time (execution) for different slave nodes. The selected dataset is 100,000 unstructured textual data and 1000 multimedia data with a cluster of 1 to 7 slave nodes. Figure 14 shows that the execution time decreases with increasing the number of slave nodes while the communication time is low and almost constant. As observed in this figure, communication delay and information overhead are somehow noticeable for a greater number of slave nodes. Therefore, it can be stated that with increasing the number of nodes, there is a significant reduction in execution time and an increase in communication overhead, and vice versa.

**5.6 The comparison of efficiency and scalability in the Hadoop cluster using deep learning**

In order to investigate the TVD-MRDL system with a deep learning approach, a sample of the data gathered from the surveillance cameras is depicted in Fig. 15. As shown in Fig. 15, the input data obtained from the traffic surveillance cameras are first divided into several segments. In the analysis of surveillance camera images, the image fragmentation is done based on the effective factors which are derived from expert knowledge. The image fragmentation is used to find the location of the traffic signs, traffic lights, vehicle, nearby object, and nearby action. In the Map phase, a convolutional neural network is selected for each segmented image based on the type of function that will be used in the next phase. For instance, the first convolutional network would be able to identify the type of category pertaining to traffic signs or traffic lights. In the Reduce phase, via the combination of the results of the convolutional networks, the driver violation is detected and displayed.

Because of the high-dimension images, it is impractical to connect the neurons to all of the neurons behind them. Therefore, each neuron is connected only to a small area of the input image. The dimensions of this small area are big enough for a hyper parameter to be connected to them. This area is called Receptive Field. Table 5 shows the training time for different

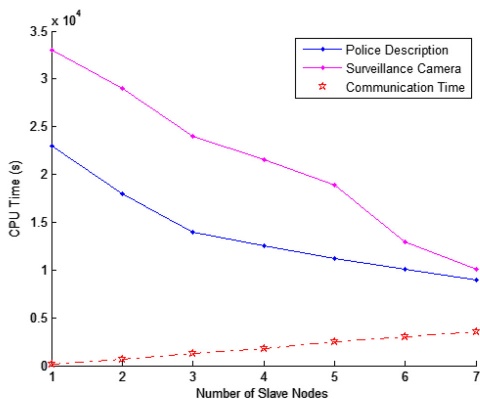


Fig. 14 Communication time (communication delay) and computation time obtained from TVD-MRDL for unstructured textual and multimedia data with respect to different numbers of slave nodes

IMAGE

Multimedia Tools and Applications

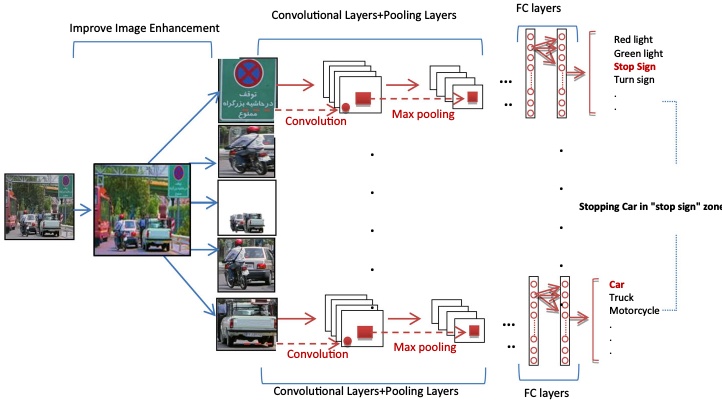


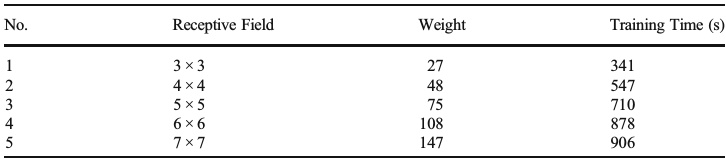
Fig. 15 A sample test data using the deep learning approach

receptive fields with different dimensions. For example, the second row in Table 5, implies that each neuron in the convolutional layer has a weighting factor of 48 for a 4 ×4 ×3 (length, width, and depth) receptive field in the input image. Therefore, the dot product of the weights and input images is performed, the results are passed through a nonlinear function, the connections are locally made with regard to the location, and the output is generated. For the training data obtained from the images in the unstructured police data with two slave nodes, the learning time of the 4 ×4 receptive field is 1189 units. As can be observed, the learning time increases as the area of the receptive field (of the model) increases. Therefore, it can be declared that the distributed violation detection algorithm requires more learning time, and that this time increases linearly with an increase in the area of the receptive field, which makes it suitable for large-scale learning.

Moreover, in order to compare the performance of the TVD-MRDL system in different conditions, the processing time of the surveillance camera images using the MapReduce programming model is shown in Fig. 5(b), the deep distributed convolutional neural network is proposed in Fig. 5(c), and a different number of slaves are also explored. The results achieved from deep learning and the detection functions in the MapReduce programming model displayed in Fig. 5(b) are given in Table 6.

In Table 6, Map Time and Reduce Time specify the processing times of the Map function and the Reduce function, respectively. Obviously, without the MapReduce technique, no time

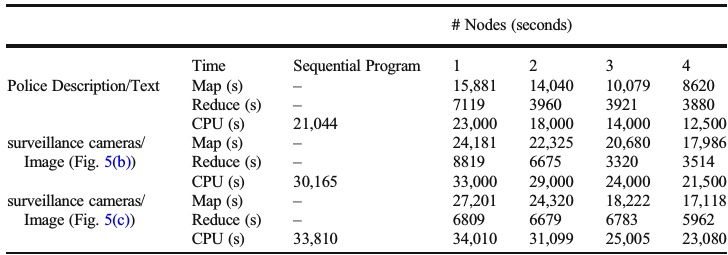
Table 5 The experimental results for different receptive fields



IMAGE

Multimedia Tools and Applications

Table 6 The MapReduce and processing times using the methods presented in Fig. 5(b) and Fig. 5(c)



would be available to process the Map and Reduce functions. CPU Time is the total Job processing time consisting of loading, shuffling, and etc. Compared to the sequential program, Hadoop with a single node required more time for large-scale data processing because of its high overhead. In addition, the processing time of the Map function decreases dramatically by increasing the number of nodes using deep learning. For example, in the processing of the textual data, the processing time of the program using deep learning for the training data of the Hadoop cluster with two nodes is equal to 31,099 units; however, the processing time reduces to 29,000 units when deep learning is not employed. By increasing the number of nodes to 4 using deep learning on each node, the processing time decreases to 23,080.

Based on the results reported in Table 6, it can be concluded that the CPU time for each node decreases as much as the number of slave’ rises due to its parallel execution, and over the course of training through deep learning, since the training time is also taken into account, the total time increases. The Map time is usually high owing to the distribution of the data on the slaves, and the CPU time is reported considering the MapReduce time. In addition, the MapReduce technique is more efficient when the number of samples increases. So, it is required to strike a balance between the number of slaves and the volume of the data set in order to obtain more efficient results.

**6Rerviewand Conlusions**

Due to the fact that the esthetic analysis process, especially for a large number of images, is substantially complex and time-consuming, in paper [52], an esthetic analytical system is propounded. In this system, Hadoop with the MapReduce technique is utilized to reduce the processing time of the images via the implementation of several slaves. Time Series Analysis and Forecast Analysis Using MapReduce Framework are used in paper [10]. In paper [10], the modeling of the climate data on a large data platform is carried out via the MapReduce approach. The results of the climate data analysis are obtained quickly by using the MapReduce technique. Also, in paper [30], a MapReduce algorithm is suggested for processing big data in meteorology (climate data). In paper [9], a Big Data clustering algorithm using the MapReduce-based Artificial Bee Colony (MR-ABC) is introduced, and the results indicate that the efficiency of the MapReduce-based clustering improves in comparison with the k- means clustering. In paper [60], a MapReduce-Based K-Means Clustering Algorithm is presented, and it is confirmed that it has the efficient capability of processing big datasets. Moreover, parallel K-PSO based on MapReduce is also investigated in paper [51]. MapReduce is also used for posterior probability clustering and relevance

Multimedia Tools and Applications

models for recommendation systems [50]. In paper [50], good scalability behavior is observed using distributed implementation on different nodes. In paper [14], an improved TF-IDF algorithm is introduced for retrieving Reuters news. The results of paper [14] indicate that the MapReduce-based TF- IDF function is improved for news categorization, word weighting, and clustering. Also, paper [3] presented a MapReduce-based SVM algorithm for the annotation (labeling) of images. The proposed distributed algorithm in article [3]not only displays considerable decrease in the training time by dividing the training datasets into smaller subsets and optimizing the divided subsets in a group of computers but also it has high accuracy and precision.

The MapReduce technique is employed in databases to optimize recurring queries [58] and data join operations [2], and a new recurring query model based on MapReduce and join operations is proposed. In paper [1], the boundary numbers of the data cubes are calculated using the MapReduce technique for lower-order computations. In medicine, the MapReduce technique is used for processing large data [13, 25]. A new data analysis framework is also available for smart cities [36], discussing the approaches to knowledge discovery in traditional systems versus those in big data systems. Student behavior detection based on the big data framework is explored in paper [12]. Paper [42] uses a wireless sensor network to monitor the level of air pollution in the city through big data. The purpose of article [57]was to collect the real-time tweets of football fans in the United States during five games at the FIFA World Cup 2014 using the Twitter API. Using emotion analysis,paper [57] examined the emotional responses of the US football fans in their tweets, in particular the sentimental changes after the goals were scored (by either their favorite teams or their opponents). In paper [32], a number of sensors are used to collect the required data, including water temperature, etc., and forecasting is done using the neural network. Because of the high volume of data received by the sensors, the data is deemed as big data. In the article [23], the Monte Carlo simulation-based forecast of traffic speed was done using big data history. In article [55], big data issued for pedestrian counting. Paper [55] simulated new data sources, such as ‘big data’, and computational analysis on pedestrians. A brief review of the relevant studies undertaken

on big data using the Hadoop framework and the MapReduce technique in various fields of study along with their advantages, disadvantages, and performance metrics are provided in Table 1.

In the area of traffic control, the following studies have been performed using deep learning. In paper [56], a new algorithm based on deep models is proposed to detect traffic signs. The results of implementing this algorithm in paper [56] indicate over 97% accuracy in detection. Paper [5]investigates several systems for detecting traffic signs via deep neural networks and analyzes the memory allocation, accuracy, and processing time of each system. In paper [7], deep learning is used for the traffic sign detection systems with approximately 100% precision. Also, in paper [6], deep learning is employed to spot traffic signs. The automatic detection of traffic lights through deep learning classification techniques based on GPS tracking during driving is discussed in paper [34]. Therefore, in most of the studies on detecting traffic signs, deep learning, as a different approach, is employed to enhance accuracy. However, the aforementioned articles rarely address the issue of processing time. To our knowledge, in the realm of Traffic control, deep learning has been mostly employed in the detection of traffic signs. In paper Asadianfam et al. [8] introduce a big data based platform for identifying the risky behaviors of vehicle drivers. This platform is used MAPE technology but is not used deep learning methods.The present study attempts to examine job processing times using deep learning along with the MapReduce technique over big traffic data including both surveillance camera data and police descriptions data hoping to identify unsafe behavior or drivers’ violation behavior. Moreover, there is a dearth of research on the

Multimedia Tools and Applications

MapReduce technique in the field of transportation and the analysis of the big traffic data. In the current study, the TVD-MRDL system, which is based on Hadoop, the MapReduce technique, and deep learning, is utilized in the hope of providing efficient solutions in this regard.

Advances in information technology have brought about exponential growth in data, making big data analysis a foregone conclusion. With the wide application of the Internet and communication technologies, the transport industry generates a large amount of real-time data that urgently needs to be analysed and processed. Undoubtedly, Hadoop has become one of the mainstream technologies for big data analytics and has been widely studied by academics as a cloud computing platform. Therefore, it is crucial to study the application of Hadoop in transport big data. Due to the complexity of the traffic big data problem, research on it using Hadoop big data technology has become a hot topic. In order to better understand the research of Hadoop in traffic big data, we have analysed a number of collected papers and identified eight application scenarios of Hadoop in traffic big data. We also summarised the development history and latest results of existing research in this area. In addition, we focused on the literature on Hadoop optimisation and identified the latest research advances in the field of transportation. Finally, we identify gaps in current research. Based on our review of the literature and bibliometric analysis, we draw the following conclusions. Real-time data plays a crucial role in various transportation big data applications such as traffic state identification, real-time traffic control, dynamic route guidance, and real-time bus scheduling. However, Hadoop has limitations in processing real-time data. Therefore, integrating Hadoop with other big data frameworks designed for real-time data processing (e.g., Apache Storm, Apache Flink, Apache Samza, and Kafka Streams) can provide an effective solution for real-time big data analytics in transport. Further research on the integration and development of these frameworks with transport big data can lead to new advances in the application of Hadoop big data technologies in the transport domain.

The basic idea behind big data is to deal with complex systems, and transport problems are a typical example of such systems. Traffic involves a wide range of interrelated factors, such as traffic flow, road conditions, driver behaviour and weather conditions, which are highly dynamic and rapidly changing, making it difficult to develop effective solutions. Therefore, an integrated approach to solving traffic problems is required that takes into account all relevant factors and their interdependencies. This approach uses big data analytics to gain insight into the system and develop data-driven solutions to reduce congestion, minimise accidents and improve overall transport efficiency. However, existing big data technologies, such as Hadoop, have limitations when dealing with relational data, especially when analysing multi-source and heterogeneous traffic data. Addressing this challenge requires integrating cross-modal, multi-technology, and cross-domain processing to enable multi-dimensional correlation of large data sets.The Hadoop Distributed File System (HDFS) plays an integral role in facilitating distributed storage within the Hadoop ecosystem. However, processing large datasets with HDFS requires strict adherence to software and hardware requirements. Fortunately, data compression techniques can alleviate some of these limitations by effectively reducing storage space requirements. Furthermore, the statistical similarities between data compression and data analysis processes mean that encoding and decoding data through artificial intelligence can improve data analysis and even replace some HDFS functionality. The transport industry could benefit enormously from this exciting prospect. However, it also presents significant challenges and requires interdisciplinary expertise in AI and traffic engineering to develop and implement efficient data compression and analysis methods. Bridging the gap between these two fields is critical to advancing the use of big data technologies in the transport sector.

Nowadays, more and more transportation systems are being controlled by intelligent machine vision programs. A very low error rate, high speed, very low maintenance cost, and many other advantages have made various industries increasingly adopt image processing and machine vision. Image monitoring systems are the main means of transportation system management. They have the advantage of providing visual information for decision making. In this paper, human resources were replaced with mechanized control using the MapReduce algorithm and deep learning. In this way, first, various mistakes caused by human error, such as visual errors, fatigue, etc., were eliminated from the system; second, the number of trained human resources required was reduced, and consequently, the associated costs were diminished. These results show that in comparison with the sequential program, Hadoop in stand-alone mode decreases the processing time of the high-volume data by more than 70%. Also, by increasing the number of slave nodes from 1 to 7 in the TVD-MRDL system, the processing time reduces by 60.87% and 70%, respectively. Given the disastrous impacts and ramifications of traffic

accidents, attempts to redress improper driving habits by revealing violations and traffic breaks can provide opportunities to control many non-social behaviors of drivers and avoid accidents and heavy casualties. The vehicle under study was the ordinary car and the tests were conducted on it. As future work, this approach is used for detection from unmanned aerial vehicles.

**References**

1. Afrati FN, Sharma S, Ullman JR, Ullman JD (2018) Computing marginals using MapReduce. J Comput Syst Sci 94:98–117

2. Afrati F, Stasinopoulos N, Ullman JD, Vassilakopoulos A (2018) Sharesskew: an algorithm to handle skew for joins in mapreduce. Inf Syst

3. Alham NK, Li M, Liu Y, Hammoud S (2011) A MapReduce-based distributed SVM algorithm for automatic image annotation. Comput Math Appl 62(7):2801–2811

4. Aoyama K (1997) “Next Generation Universal Traffic Management System (UTMS’21) in Japan,” in Intelligent Transportation System, 1997. ITSC’97., IEEE Conference on, pp. 649–654: IEEE

5. Arcos-García Á, Álvarez-García JA, Soria-Morillo LM (2018) Evaluation of deep neural networks for traffic sign detection systems. Neurocomputing 316:332–344

6. Arcos-García Á, Álvarez-García JA, Soria-Morillo LM (2018) Deep neural network for traffic sign recognition systems: an analysis of spatial transformers and stochastic optimisation methods. Neural Netw 99:158–165

7. Arcos-García Á, Soilán M, Álvarez-García JA, Riveiro B (2017) Exploiting synergies of mobile mapping sensors and deep learning for traffic sign recognition systems. Expert Syst Appl 89:286–295

8. Asadianfam S, Shamsi M, Rasouli Kenari A (2020) Big data platform of traffic violation detection system: identifying the risky behaviors of vehicle drivers. Multimedia Tools and Applications 79(33):24645–24684. https://doi.org/10.1007/s11042-020-09099-8

9. Banharnsakun A (2017) A MapReduce-based artificial bee colony for large-scale data clustering. Pattern Recogn Lett 93:78–84

10. Bendre M, Manthalkar R (2019) Time series decomposition and predictive analytics using MapReduce framework. Expert Syst Appl 116:108–120

Multimedia Tools and Applications

11. Bui-Minh T, Ghita O, Whelan PF, Hoang T, Truong VQ (2012) “Two algorithms for detection of mutually occluding traffic signs,” in Control, Automation and Information Sciences (ICCAIS), 2012 International Conference on, pp. 120–125: IEEE

12. Cantabella M, Martínez-España R, Ayuso B, Yáñez JA, Muñoz A (2019) Analysis of student behavior in learning management systems through a big data framework. Futur Gener Comput Syst 90:262–272

13. Cattaneo G, Giancarlo R, Petrillo UF, Roscigno G (2016) “MapReduce in computational biology via Hadoop and spark,” Encyclopedia of Bioinformatics and Computational Biology, pp. 1–9

14. Chen C-H (2017) Improved TFIDF in big news retrieval: an empirical study. Pattern Recogn Lett 93:113– 122

15. Chen M, Mao S, Liu Y (2014) Big data: A survey. Mobile networks and applications 19(2):171–209

16. Chen M, Mao S, Zhang Y, Leung VC (2014) Big data: related technologies, challenges and future prospects. Springer

17. De La Escalera A, Armingol JM, Pastor JM, Rodríguez FJ (2004) Visual sign information extraction and identification by deformable models for intelligent vehicles. IEEE Trans Intell Transp Syst 5(2):57–68

18. De Mauro A, Greco M, Grimaldi M, Ritala P (2018) Human resources for big data professions: a systematic classification of job roles and required skill sets. Inf Process Manag 54(5):807–817

19. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113

20. Elotmani S, El Hitmy M (2014) “A light traffic signs recognition system,” in Multimedia Computing and Systems (ICMCS), 2014 International Conference on, pp. 459–464: IEEE

21. Gantz J, Reinsel D (2011) Extracting value from chaos. IDC iview 1142(2011):1–12

22. Goodfellow I, Bengio Y, Courville A (2016) “Deep learning (adaptive computation and machine learning series),” Adaptive Computation and Machine Learning series, p. 800

23. Jeon S, Hong B (2016) Monte Carlo simulation-based traffic speed forecasting using historical big data. Futur Gener Comput Syst 65:182–195

24. Kasaei SHM, Kasaei SMM (2011) “Extraction and recognition of the vehicle license plate for passing under outside environment,” in Intelligence and Security Informatics Conference (EISIC), 2011 European, pp. 234–237: IEEE

25. Kouanou AT, Tchiotsop D, Kengne R, Tansaa ZD, Adele NM, Tchinda R (2018) “An optimal big data workflow for biomedical image analysis,” Informatics in Medicine Unlocked

26. Krishnan A, Lewis C, Day D (2009) “Vision system for identifying road signs using triangulation and bundle adjustment,” in Intelligent Transportation Systems. ITSC’09. 12th International IEEE Conference on, 2009, pp. 1–6: IEEE

27. Laney D (2001) 3D data management: Controlling data volume, velocity and variety. META group research note 6(70):1

28. Le TM, Liaw S-Y (2017) Effects of Pros and Cons of Applying Big Data Analytics to Consumers’ Responses in an E-Commerce Context. Sustainability 9(5):798

29. Lotfi E (2011) “Trajectory Clustering and Behaviour Retrieval from Traffic Surveillance Videos,” Majlesi Journal of Multimedia Processing, vol. 1, no. 2

Multimedia Tools and Applications

30. Manogaran G, Lopez D, Chilamkurti N (2018) In-mapper combiner based MapReduce algorithm for processing of big climate data. Futur Gener Comput Syst 86:433–445

31. McLauchlan P, Beymer D, Coifman B, Mali J (1997) “A real-time computer vision system for measuring traffic parameters,” in cvpr, p. 495: IEEE

32. Millie DF, Weckman GR, Young II WA, Ivey JE, Fries DP, Ardjmand E, Fahnenstiel GL (2013), “Coastal ‘big Data’and nature-inspired computation: prediction potentials, uncertainties, and knowledge derivation of neural networks for an algal metric,” Estuarine, Coastal and Shelf Science 125:57–67

33. Moghaddam AM, Ayati E (2014) Introducing a risk estimation index for drivers: a case of Iran. Saf Sci 62: 90–97

34. Munoz-Organero M, Ruiz-Blaquez R, Sánchez-Fernández L (2018) “Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on GPS traces while driving,” Computers, Environment and Urban Systems 68:1–8

35. Nguyen V, Kim H, Jun S, Boo K (2018) A study on real-time detection method of lane and vehicle for lane change assistant system using vision system on highway. Engineering science and technology, an international journal 21(5):822–833

36. Osman AMS (2018) “A novel big data analytics framework for smart cities,” Futur Gener Comput Syst

37. Park SH, Jung K, Hea JK, Kim HJ (1999) “Vision-based traffic surveillance system on the internet,” in Computational Intelligence and Multimedia Applications, 1999. ICCIMA’99. Proceedings. Third International Conference on, pp. 201–205: IEEE

38. Patterson J, Gibson A (2017) Deep Learning: A Practitioner’s Approach. “O’Reilly Media, Inc.”

39. Phung SL, Le MC, Bouzerdoum A (2016) Pedestrian lane detection in unstructured scenes for assistive navigation. Comput Vis Image Underst 149:186–196

40. Rahemi Z, Ajorpaz NM, Esfahani MS, Aghajani M (2017) Sensation-seeking and factors related to dangerous driving behaviors among Iranian drivers. Personal Individ Differ 116:314–318

41. Rakotonirainy A, Schroeter R, Soro A (2014) Three social car visions to improve driver behaviour. Pervasive and Mobile Computing 14:147–160

42. Rios LG (2014), “Big data infrastructure for analyzing data generated by wireless sensor

networks,” in Big Data (BigData Congress), 2014 IEEE International Congress on, pp. 816–823: IEEE

43. Sallah M, Sarah S, Hussin FA, Yusoff MZ (2011) “Road sign detection and recognition system for real-time embedded applications,”

44. Saptharishi M, Spence Oliver C, Diehl CP, Bhat KS, Dolan JM, Trebi-Ollennu A, Khosla PK (2002) Distributed surveillance and reconnaissance using multiple autonomous ATVs: CyberScout. IEEE Trans Robot Autom 18(5):826–836

45. Schmidhuber J (2015) Deep learning in neural networks: An overview. Neural Netw 61:85–117

46. Secundo G, Del Vecchio P, Dumay J, Passiante G (2017) Intellectual capital in the age of big data: establishing a research agenda. J Intellect Cap 18(2):242–261

47. Shvachko K, Kuang H, Radia S, Chansler R (2010) “The hadoop distributed file system,” in Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on, pp. 1–10: Ieee

48. Stauffer C, Grimson WEL (2000) Learning patterns of activity using real-time tracking. IEEE Trans Pattern Anal Mach Intell 22(8):747–757

Multimedia Tools and Applications

49. Tao D, Zhang R, Qu X (2017) The role of personality traits and driving experience in self-reported risky driving behaviors and accident risk among Chinese drivers. Accid Anal Prev 99:228–235

50. Valcarce D, Parapar J, Barreiro Á(2018) A MapReduce implementation of posterior probability clustering and relevance models for recommendation. Eng Appl Artif Intell 75:114–124

51. Wang J, Yuan D, Jiang M (2012) “Parallel K-PSO based on MapReduce,” in Communication Technology (ICCT), 2012 IEEE 14th International Conference on, pp. 1203–1208: IEEE

52. Wang W, Zhao W, Cai C, Huang J, Xu X, Li L (2015) An efficient image aesthetic analysis system using Hadoop. Signal Process Image Commun 39:499–508

53. White T (2012) Hadoop: The definitive guide. “O’Reilly Media, Inc.”

54. Yi S-C, Chen Y-C, Chang C-H (2015) A lane detection approach based on intelligent vision. Comput Electr Eng 42:23–29

55. Yin L, Cheng Q, Wang Z, Shao Z (2015) ‘Big data’for pedestrian volume: exploring the use of Google street view images for pedestrian counts. Appl Geogr 63:337–345

56. Yu Y, Li J, Wen C, Guan H, Luo H, Wang C (2016) Bag-of-visual-phrases and hierarchical deep models for traffic sign detection and recognition in mobile laser scanning data. ISPRS J Photogramm Remote Sens 113: 106–123

57. Yu Y, Wang X (2015) World cup 2014 in the twitter world: a big data analysis of sentiments in US sports fans’ tweets. Comput Hum Behav 48:392–400

58. Zhang B, Wang X, Zheng Z (2018) The optimization for recurring queries in big data analysis system with MapReduce. Futur Gener Comput Syst 87:549–556

59. Zhang W, Yoshida T, Tang X (2011) A comparative study of TF\* IDF, LSI and multi-words for text classification. Expert Syst Appl 38(3):2758–2765

60. Zhao W, Ma H, He Q (2009) “Parallel k-means clustering based on mapreduce,” in IEEE International Conference on Cloud Computing, pp. 674–679: Springer

61.Y. M. Z. Zhang, "Environmental benefits of bike sharing: A big data-based analysis," Applied Energy, vol. 220, pp. 296-301, 2018.

62. S. W. &. N. H. Elliot Fishman, "Bike Share: A Synthesis of the Literature Transport Reviews," no. 33:2,, pp. 148-165, 2013.

63 N. G. H. M. Christine Fricker, "Mean field analysis for inhomogeneous bike sharing systems.," Advances in Complex Systems, vol. 14, no. 03.2011, jul 2012.

64.R. &. R. T. A. Godavarthy, "Winter Bikesharing in US: User Willingness, andOperator’s Challenges and Best Practices.,"Sustainable Cities and Society, vol. 30.10.1016/j.scs.2017.02.006., 2017.

65.T. L. &. W. C. J. Hamilton, "Bicycle infrastructure and traffic congestion: Evidence from DC's Capital Bikeshare," Journal of Environmental Economics and Management, Elsevier,, vol. 87(C), pp. 72-93, 2018.

66.L. C. R. O. M. S. W. Y. Caggiani, "A modeling framework for the dynamic management of free-floating bike-sharing systems," Transportation Research Part C: Emerging Technologies, vol. 84, pp. 159-182, 2018.

67.J. Y. Z. a. X. X. Yuan, "Discovering regions of different functions in a city using human mobility and POIs," Proceedings of the 18th ACM SIGKDD international conference onKnowledge discovery and data mining, p.186194, 2012.

68.J. R. H. a. W. v. H. Schuijbroek, "Inventory RebalancingandVehicleRoutinginBikeSharing Systems," TepperSchool of Business., p. 1491., 2013.

Multimedia Tools and Applications

69. L. M. O. Caggiani, "A Modular Soft Computing based Method for Vehicles Repositioning in Bike-sharing Systems.," Procedia - Social and Behavioral Sciences, 2012.

70. X. ,. W. P. G. J. S. M. A. R. K. M. J. A. D. S. Han, "Bike sharing data analytics for silicon valley in USA," 2017 IEEE SmartWorld Ubiquitous Intelligence and Computing,Advanced and Trusted Computed, ScalableComputing and Communications, Cloud andBig Data Computing, Internet of People and

Smart City Innovation,SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI 2017 - , pp. 1-9, 2018.

71. F. Soriguera, V. Casado, and E. Jimnez, “A simulation model forpublic bike-sharing systems,” Transportation Research Procedia, vol. 33,pp. 139 – 146, 2018, xIII Conference on Transport Engineering,CIT2018.

72.A. Gandomi and M. Haider, “Beyond the hype: Big dataconcepts, methods, and analytics,” International Journal of InformationManagement, vol. 35, no. 2, pp. 137 – 144, 2015.

**Publisher**’**s note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations