Scalable Traffic Video Analytics using Hadoop MapReduce

Vaithilingam Anantha Natarajan
Department of Computer Science and Engineering
Annamalai University
Tamilnadu, India
Email: v.ananth.satyam@gmail.com

Subbaiyan Jothilakshmi
Department of Computer Science and Engineering
Annamalai University
Tamilnadu, India
Weisberg Division of Computer Science
Marshall University
West Virginia, USA
Email: jothi.sekar@gmail.com

Venkat N Gudivada
Weisberg Division of Computer Science
Marshall University
West Virginia, USA
Email: gudivada@marshall.edu

***Abstract*—Road traffic video analytics aims at using a number of techniques to achieve better traffic and road safety, control congestion and provide immediate care for accident victims. In this paper, we propose a near real-time traffic analytics system which can automatically detect road accidents from live video streams. The system alerts nearby hospitals and highway rescue teams when accidents occur. It also detects road congestion and broadcasts alternative route information to relevant commuters. We have implemented the system using Apache Hadoop. Analysis results are stored in Hive, which is a data warehouse built on top of Hadoop. This data includes overall traffic speed, traffic volume, and individual vehicle speed. Hive provides data summarization, query, and analysis. We have deployed and tested the system on a cluster computer.**

Keywords*—*Traffic Video Analytics; Road Accidents.

I. INTRODUCTION

The economy of a city greatly relies on its road network and it is important to monitor this infrastructure. Traffic jams, congestion, and accidents in city roads is a common problem in most major cities across the world. Road traffic accidents on the highways are increasing. The World Health Organization reports that by 2030 road traffic accidents will become the fifth leading cause of human death. The National Crime Records Bureau reports that every year in India more than 135,000 traffic collision-related deaths occur [1].

Real-time and historical data on road traffic is essential to effectively manage city road networks and minimize road traffic related deaths. This data can be used to ease traffic congestion by suitably programming traffic lights and suggesting alternative routes to drivers through instant messaging services. Also, historical traffic data is used to identify peak traffic hours, highly congested road intersections and accident-prone roadways.

Current generation traffic monitoring systems have the capability to capture and transmit real-time live traffic data, number of vehicles that pass through an intersection as a function of time intervals, and average speed of vehicles. The data grows in size quite rapidly and analysis entails high computational requirements. Furthermore, this data combined with Geographic Information Systems (GIS) and Global Positioning System (GPS) enable new possibilities that were not possible hitherto.

Most of the research in road traffic analytics is done in the context of developed countries where traffic follows strict lane discipline. Sensor devices are used to detect vehicles and measure road traffic variables. Sensor devices include magnetic loop sensors, speed guns, and video surveillance cameras. Magnetic loop detectors are used in large-scale for traffic monitoring [2]. Installation and maintenance of these sensor-based systems are both labor intensive and expensive. Moreover, drivers do not follow lane discipline in over 90% of the cities in the developing world. Furthermore, using sensors to detect vehicles and collect traffic data is prohibitively expensive for these cities.

Another approach to traffic monitoring and analysis is based on video image processing. In [3], an approach to vehicle detection using a rule-based reasoning is proposed. In another work [4], a Hidden Markov model (HMM)-based technique is proposed to detect accidents and other events at road intersections. Computer vision based techniques for detecting vehicles using corner features is discussed in [5]. Here again, lane abiding traffic as well as average vehicle speeds in the range 80 *km/h* to 110 *km/h* are assumed. Clearly, these approaches do not work in situations where there is no lane discipline and vehicle speed ranges vary greatly.

To circumvent the requirements of lane discipline and sensor arrays, we propose a data-driven approach to road traffic analytics using digital video. Most cities have digital video cameras installed in hundreds of locations primarily for monitoring crime and terrorist activities. They generate video data per day at the scale of terabytes. Issues involved include efficient and secure transmission and storage, processing and feature extraction, storage and retrieval of features, and performing analytics on feature data. Analytics reveal traffic patterns keyed to geographic location and time intervals, congestion and accident reports. History of feature data should be maintained to enable both descriptive and predictive analytics. The latter is critical for enhancing accuracy of traffic pattern forecasting, preventive maintenance, and proactive capacity

 ISBN: 978-1-61208-445-9

planning.

In this paper, we describe a traffic monitoring system that we developed which is suited for road traffic conditions in the developing world. Our system is based on Hadoop MapReduce framework and can capture, process, store, analyze, and retrieve video data at scale. Our system detects and tracks individual vehicles in the video frames and computes total number of vehicles that have passed through an intersection over a time interval. It also computes the speed of individual vehicles and average speed of vehicles. The system detects vehicle collisions which can be communicated to a nearby hospitals and highway rescue teams in real-time. Additional functionality of the system includes suggesting alternative routes to commuters when congestion is spotted on roadways.

The remainder of the paper is structured as follows. The traffic monitoring system is outlined in Section II. Architecture of the system is described in Section III. Section IV describes various processes involved in computing traffic analytics – video splitting technique using Apache Hadoop, vehicle detection using Haar classifiers and Support Vector Machine (SVM), and an algorithm for speed estimation. Experimental results and their analysis are presented in Section V. Finally, Section VI concludes the paper.

II. MapReduce based Video Processing

Our application uses Apache implementation of MapReduce programming paradigm, Hadoop, to chunk incoming video frames and decode them into a sequence file of image frames in parallel mode. The application can run on a local network or can be deployed on clouds that support Infrastructure as a Service (IaaS). The Hadoop Distributed File System (HDFS) is used for storing massive volumes of data across a number of physical disk clusters.

To enhance retrieval and processing speeds, the sequence files are split into smaller units known as *chunks* and distributed over HDFS nodes. When jobs are submitted to Hadoop, they are scheduled on machines that host the required data. Our system can be integrated with various traffic video capturing devices like Internet protocol (IP) cameras, closed-circuit television (CCTV), and other traffic video streaming systems.

Different video devices capture video in different file formats. A video file format can be considered as a logical container that wraps compressed video and audio. The results presented in [6] influenced the use of MapReduce framework for our system. The latter transcodes various video code formats into MPEG-4 video format.

The raw video streams are automatically partitioned into blocks. The blocks are of size 64 MB and are stored in cluster nodes. The distributed MapReduce video processing algorithm is illustrated in Figure 1. In the first mapping phase of MapReduce, the cluster nodes are assigned data from the frames of a single video [7]. The input video frames are read and vehicles are detected using a machine-learning algorithm. For each frame processed by a map, a corresponding output frame is produced and is indexed for subsequent efficient access. The index contains pairs of key values which are used to connect a frame identifier with the corresponding data.

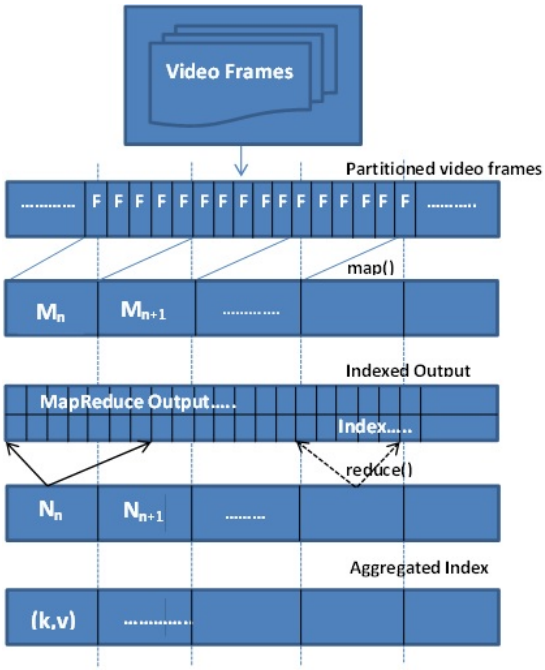

Fig. 1. MapReduce based video processing algorithm

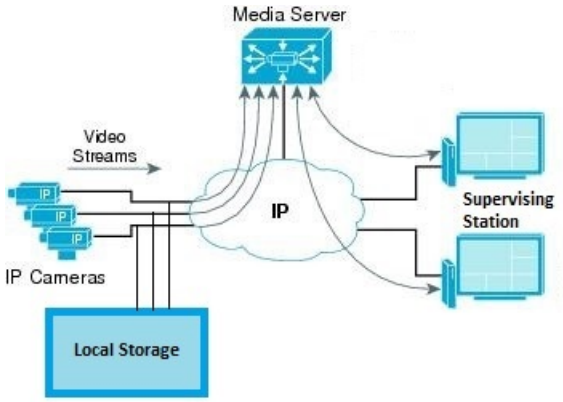

Fig. 2. System architecture

During the second phase, the index maps generated during the first phase are reduced into a single result map.

The information extracted by the MapReduce including vehicle counts, accident data and average vehicle speed are stored in Hive data warehouse for further analysis. Hive is part of the Apache Hadoop ecosystem and is optimized for analytics workloads.

III. System Architecture

The architecture of our system is shown in Figure 2. It is divided into four modules – video streams, IP wireless transmission, media server, and supervisor/control station.

The video recording module provides two functions. It streams the video to the control station. It also saves the video

 ISBN: 978-1-61208-445-9

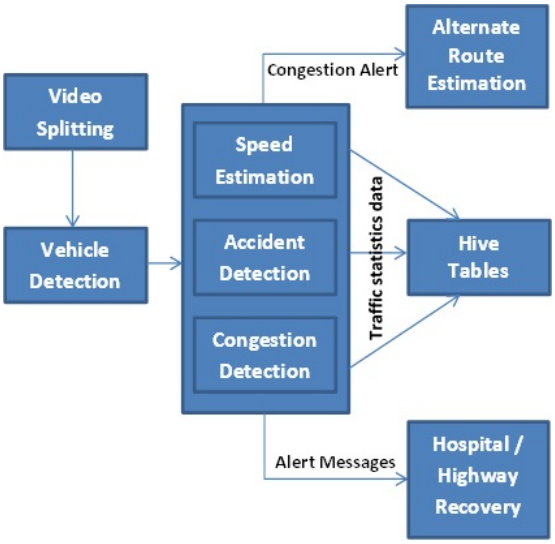

Fig. 3. Block diagram of video analytics process

in digital form in its local storage. The IP wireless transmission module routes the captured video from various spots in the city to the control station via the media server. The control station processes the live video streams and stores them in Hive data warehouse for further analysis.

The latitude and longitude of important places in and around city are stored in a Hive table. They are used to suggest alternative routes when road congestion occurs. The alternate routes are computed on-demand basis using the Google Map Directions application programming interface (API).

Our Hadoop implementation runs on a 3-node cluster. We have tested the application in three scenarios by varying the number of nodes in the cluster.

IV. MapReduce based Video Analytics

Hadoop does not have built-in capability to extract structured data from unstructured video. Shown in Figure 3 is the process our system uses for video analytics. After splitting the video into chunks, vehicles are detected from the chunks. Vehicle detection is not a trivial task and is performed in two stages. First, Haar classifiers are used for pre-detecting vehicles in the video frames [8]. This is a pre-processing step. In the second stage, a Support Vector Machine (SVM) is used to accurately detect the presence of vehicles.

The Haar classifier works fast with low-resolution images. However, SVM works slower due to algorithmic complexity involved in extracting complex features and the need for high-resolution detection window. Next, vehicle speeds are estimated, accidents and congestion are detected. All this structured data is written to Hive data warehouse. If congestion is detected, alternative route messages are sent to mobile phones of subscribed users. If an accident is detected, messages are sent to hospitals and highway rescue teams which are close by the accident site. These steps are discussed below in some detail.

A. Video Splitting

Generally in Hadoop the file is split into blocks of specified size (default is 64 MB). Each block is processed by one map process and map processes run in parallel. Larger block size offers several advantages. Video files are split into blocks in a way to avoid information loss. In other words, we do not want some video frames of an accident to go to one mapper process and remaining frames of the same accident go to another mapper.

We split video into blocks based on time units, which is specified by the users of the system. In the case of a single cluster mode, the video file is processed by one map process. For two- and three-node scenarios, the video file is processed two and three map processes, respectively. Our scalable video analytics approach is illustrated in Figure 4.

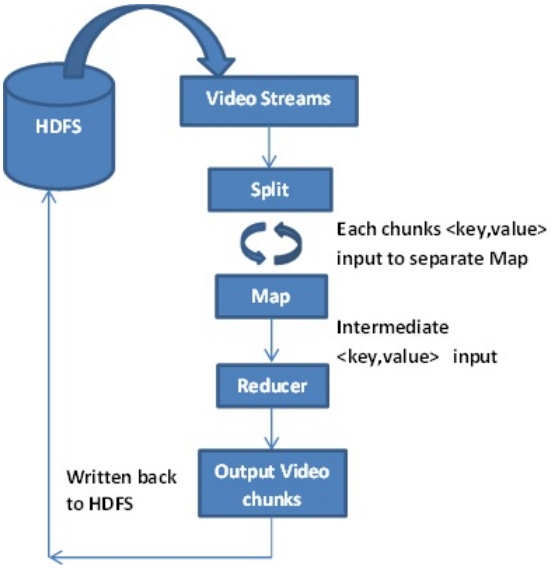

Fig. 4. Process flow for MapReduce based video processing

B. Vehicle Detection using Haar Classifier and SVM

As indicated in Section IV, vehicle detection is done using a two-step process. Haar classifiers' detection speed is high. The prediction accuracy is high with low false positives. In the second step, SVM uses Histogram of Gradients (HOG) features to further improve vehicle detection accuracy.

A Haar feature for vehicle contains a set of adjacent rectangles and the position of the rectangles is defined relative to the detection window that acts like a bounding box to the vehicle. To detect vehicles, a window is scanned over the input image, and for each subsection of the image Haar-like feature is computed. This difference is then compared to a learned threshold that separates non-objects from objects. The Haar-like features are combined into a classifier cascade to form a strong classifier.

Though vehicle detection using Haar-training is more difficult compared to SURF (Speeded Up Robust Features) [9], its robustness and execution speed compensate for this difficulty. We first train the classifier by providing both positive and negative examples. We have used 2,000 photos containing

 ISBN: 978-1-61208-445-9

vehicles (positive examples) and 2,500 background images without vehicles (negative examples) for the training step of the classifier.

A detector was used to identify Regions of Interest (ROI) in images by sliding a window across the image. The detector determines whether the detected object is present inside the window by using a cascade classifier. The size of the window is varied to detect objects at different scales, but keeping the aspect ratio constant. After coarse detection, the ROI is clipped from the original video frame and then the presence of the vehicle is verified using the SVM. If more than one vehicle is present in a video frame, all the ROIs are clipped and verified using the SVM.

C. Vehicle Speed Estimation

The speed of the moving object is computed using the displacement of centroids. To overcome the problem of perspective distortion, the camera calibration parameters are used to convert the pixel distance measured from the video frames into real world distance. Images represent the real world 3D coordinates as 2D points. Therefore, the camera calibration matrix must be known to convert the speed measurement calculated in pixels per second to the actual distance traveled by the object [10]. The accuracy of the estimated speed is calculated by comparing it to the average traffic speed.

RGB format video frames are converted into gray scale and a reference background image from video stream is generated. Then, the moving objects were extracted from the video frame by finding the difference between the current frame, I_t and the reference frame or previous frame, I_{t-1}. The frame-differencing algorithm is used to detect the motion of the detected vehicles.

When there is a change in the illumination, the reference background image is updated. The extracted vehicle moving regions in the image are converted into binary format. A series of morphological operations are performed on the binary image. Next, the objects contours are filled to remove holes inside object areas.

The speed of a vehicle is the ratio between the difference of object centroid at time t and $t + 1$ and the frame rate of the capturing device. The speed computed is in pixels per second and is converted to *km/h* using the camera calibration parameters.

Traffic congestion was detected when vehicles are traveling below the average speed for a period of time. Similarly, road accidents were detected using two measures. One is the difference between the centroids of the detected vehicles, which indicates that there may be a collision. The second one is when the vehicle speed falls close to zero. For generating rerouting messages to subscribers in case of congestion, current positions of users are calculated using the Geolocation API and an alternate route estimation is provided using Google Maps API.

V. Experiments and Results

For our experiments, we placed a handy cam just outside our lab building facing the roadside and collected videos during various lighting conditions. The camera was connected to a laptop and live video was streamed and transmitted to the MapReduce application running on a virtualization server through local campus network. The performance of the application is also tested on the road traffic videos collected from Internet sources.

Fig. 5. Result of the vehicle detection process

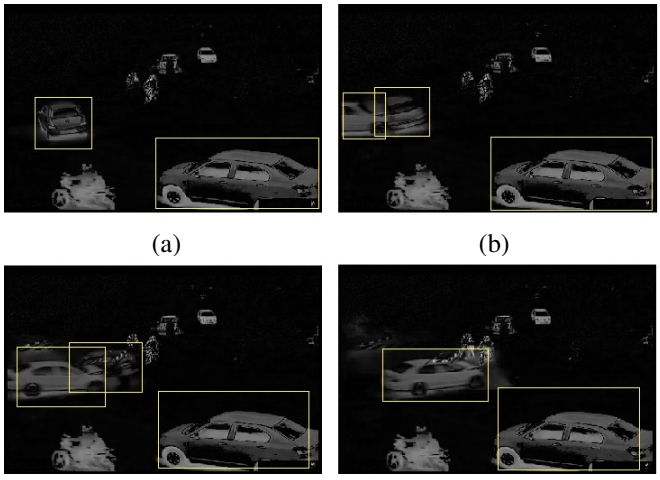

Fig. 6. (a) Single vehicle detected (b) & (c) Collision detected (d) The speed of the vehicles becomes almost zero

To detect vehicles, Haar classifier cascades have to be trained first. For training two image sets are needed – negative and positive examples. The location of vehicles within the positive images was given as input to the classifier. The SVM is then trained with HOG features extracted from the training data. The intermediate result of the vehicle analytics process in two consecutive video frames is shown in Figure 5.

Next, vehicle speed is estimated by finding the centroid of the vehicle. When there is a collision the centroid of the vehicles colliding with each other appears to be closer. This does not guarantee an accurate detection of the collision or accident since the same condition will prevail when the vehicles pass each other. When the distance between the centroids is less than a threshold value T_1, the speed of the vehicle is checked. If the speed is less than a threshold value T_2, an accident is confirmed. Figure 6 shows the output of the frame differencing process done to detect vehicle collision and speed estimation in the consecutive frames with an interval of three frames.

The result of the vehicle detection was verified manually by annotating a short video sequence and then comparing the result of the MapReduce application. The overall accuracy of

 ISBN: 978-1-61208-445-9

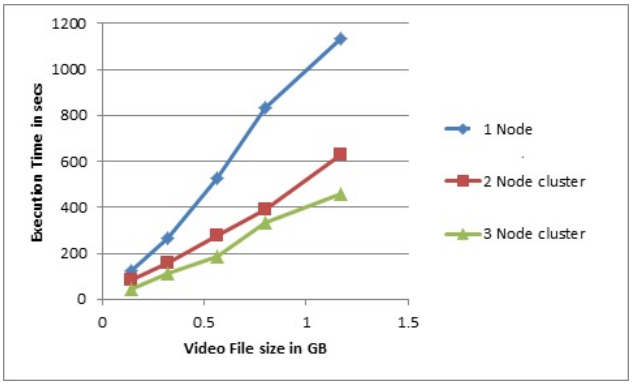

Fig. 7. Performance of MapReduce application

the vehicle detection process was found to be 88.6%. Certainly reducing the rate of false positives will increase the detection accuracy. This is achieved by limiting the region of the video frame that is analyzed.

The performance of our application is analyzed by running it on single and multiple nodes of a cluster. Figure 7 shows the performance of the application when run on one, two, and three nodes of a cluster. As expected, the time required to process video data decreases linearly with the increasing number of nodes. The scalability of our application is illustrated in Table I. When the application is run using three nodes, the processing time required to extract traffic statistics data is almost equal to the time taken by humans.

TABLE I. PERFORMANCE COMPARISON

Video file size in GB	Duration of the video file in hours	Execution time in hours		
		1 Node	2 Nodes	3 Nodes
5.60	19.20	63.60	38.40	22.10
4.80	15.30	46.20	30.10	18.16
3.43	11.15	33.80	21.90	13.30
1.17	3.80	18.15	9.89	6.05

VI. Conclusion and Future Work

In this paper, we presented an analytics solution for road traffic video using Apache MapReduce implementation, Hadoop. We have demonstrated the scalability of the system. In future, the system performance will be analyzed by incorporating more nodes. Live analysis of video data is a task operating on a stream of data. Hadoop is intended for batch processing of large volumes of data. To support real time stream computing, Storm will be considered in future instead of Hadoop. We plan to work on enhancing high-level event recognition and prediction as well as classifying vehicles. We will also investigate and validate the relationship between collision probability and safety.

Acknowledgment

S. Jothilakshmi is a postdoctoral researcher at Marshall University, USA. She is sponsored by the University Grants Commission of India under Raman Fellowship program.

References

[1] National Crime Records Bureau. Accidental deaths in India. Retrieved March 9, 2015. [Online]. Available: http://ncrb.nic.in/CD-ADSI-2012/accidental-deaths-11.pdf

[2] S.-Y. Cheung, S. Coleri, B. Dundar, S. Ganesh, C.-W. Tan, and P. Varaiya, "Traffic measurement and vehicle classification with a single magnetic sensor," *Transportation Research Record*, pp. 173–181, 2005.

[3] R. Cucchiara, M. Piccardi, and P. Mello, "Image analysis and rule-based reasoning for a traffic monitoring system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 119–130, Jun. 2000.

[4] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 1, no. 2, pp. 108–118, 2000.

[5] B. Coifman, D. Beymer, P. McLaughlin, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," in *Transportation Research Part C: Emerging Technologies*, 1998, vol. 6, no. 4, pp. 271–288.

[6] M. Kim, Y. Cui, S. Han, and H. Lee, "Towards efficient design and implementation of a hadoop-based distributed video transcoding system," *Multimedia and Ubiquitous Engineering*, vol. 8, no. 2, pp. 213–224, 2013.

[7] R. Schmidt and M. Rella, "An approach for processing large and non-uniform media objects on mapreduce-based clusters," *Lecture Notes in Computer Science*, vol. 7008, no. 2011, pp. 172–181, 2011.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. I–511–I–518.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[10] U. U. Sheikh and S. A. R. Abu-Bakar, "Three-dimensional pose estimation from two-dimensional monocular camera images for vehicle classification," in *Proc. of 6th WSEAS International Conference on Circuits,Systems, Electronics, Control and Signal Processing*, 2007, pp. 356–361.

 ISBN: 978-1-61208-445-9