Fuzzy Systems and Soft Computing
ISSN : 1819-4362

USING THE INTERNET OF THINGS TO DETECT CROWD PANIC

DEBASISH MOHAPATRA
Raajdhani Engineering College, Bhubaneswar
1debasishmohapatra@rec.ac.in

Abstract*— *Detecting crowd behavior is crucial for applications in smart cities, like those where people congregate for various occasions. But because of the internal states of the crowd and the surrounding surroundings, it is a difficult task. This research presents a novel framework for detecting crowd behavior based on several characteristics. Initially, we utilize a computer vision method founded on the scale invariant feature transform (SIFT) to categorize crowd behavior into two groups: panicked and normal. Then, in addition to the time stamp, we take into account several other environmental aspects, such as crowd coherency, social interaction, motion information, internal chaos level, randomness in crowd speed, crowd condition, crowd temporal history, and crowd vibration state. These parameters are then supplied to the deep learning model throughout the training phase.

Keywords—VANET; smart cities; crowd behavior; deep learning; Recurrent Neural Networks (RNN); Convolution Neural Networks (CNN); Invariant Feature Transform (SIFT)

I. INTRODUCTION

Smart cities are intended to ease the automated services for people inside their living areas. Recently, smart city services have been the subject of many research articles [1-4]. For example, one of the services reported in [2] is measuring the health of historical building where maintaining such building is challenge since it requires a frequent visits by administrators and technical staff to check their functionalities. Sensors are used to measure the building characteristics including humidity, temperature, and vibration. Such automated techniques for providing services that might require huge manpower and time invigorated the concept of smart cities. One more service towards smart cities is the waste management where it is a major issue for many cities. Sensors could be placed in garbage cans and when they are full, the system sends the garbage according to the shortest possible route. However, the most important and integral part of smart cities is crowd management with the increase of population. Different social events take place in different public places. Todetect abnormal behaviors in crowd is very important for the concept of smart cities as a whole.

Crowd monitoring for behavior detection is an important application for smart cities where the system detects abnormalities during crowd gatherings. This helps to identify any problem source and enable the authority to monitor it. Other well investigated services in smart city traffic monitoringinclude handling traffic congestion, smart signals, and vehicles routing. Cameras and sensors could be sources to track congested area in crowd and report it back to the authority with less effort compared to the traditional methods where police officers are asked to report any accident of congested crowded area.

For crowd analysis, smart city services may require the interaction between many elements on the roads, homes, and government buildings, and other infrastructures. Regardless of the provided services in the smart cities, effective mechanisms are required to route the collected information from one place to another; other competent mechanisms are mandatory to manage many of the heterogeneous devices as well as many of the newly produced sensors. This raises the concept of internet of things (IoT) and its role towards the implementation of smart cities for crowd analysis to detect abnormal behaviors.

IoT is a network that accomplishes the seamless integration among sub networks such as sensor networks for public places, vehicular networks, and mobile networks. The interaction among these networks allows ease of data exchange between their elements. For example, a mobile node can exchange data with a vehicle in seamless manner as they are on the same network. However, for a seamless integration among the IoT networks, an elegant model is compulsory. A few models are proposed in the literature including the ones in [5-8]. However, most of the existing models were concern about certain networks. For instance, references [9-11] were focusing on RFID connectivity to IoT environment. Other models are proposed in [12] and [13] for connecting mobile, vehicular and sensor networks in the IoT environment. In both articles, a gateway including access point is proposed as a mean to integrate these networks together. In addition, authors of [14] reported that IoT gateway deployment as a message ferry outperforms placing the gateway based on petro graphical area.

One of the convoluted networks in smart cities applications in general is the Vehicular Ad Hoc Network (VANET) [15]. VANET consists of three main components which are On Board Unit (OBU), Application Unit (AU), and RoadSide Unit (RSU). OBU is a device that is installed in a public place to pass the collected data from sensors or applications in public places to the other OBU in other places or to RSU. AU is a device resides in a public or crowd place that can be utilized for the crowd surrounding information. The final component isRSU, where it is mainly installed along the roads, intersection areas, or crowd public places. The main role is to collect the information that is passed from the OBU to a control center. Based on the transferred data via RSU, the authority in the control center can come up with any crowd related decision.

Based on the previous VANET facilities and new sensor capabilities, VANET could be one of the main players in IoT that could be exploited in many IoT services including the detection of abnormal crowd behavior in smart cities. Smart cars can identify the crowd congestion in order to facilitate rerouting in the smart city and it can also help to lower the occurrence of abnormal crowd behaviors. One of the challenges in VANET that contributes to the IoT is the detection of crowd behavior as a whole. Identifying crowd behavior could be used as early indicator of a dangerous situation that might lead to chaos in public places. In order to identify crowd behavior, sensors in public places and other units could be of help reporting different features/parameters used to do so. This paper investigates the detection of crowd behavior identification. The problem is not new; however, the previous work focuses on small set of parameters and use traditional methods for the same purpose.

Next section of the paper introduces the literature review to crowd behavior detection; proposed method is presented in Section III; results are elaborated in Section IV; discussion is presented in Section V and Section VI presents the conclusion.

II. Literature Review

Crowd behaviors have been significantly investigated in the literature including [15-22]. The importance of such investigations comes from the impact of such behavior on the people safety. Throughout this section, we summarize the state-of-the-art of crowd behaviors in some of the recent literatures. Table I presents the work done in this field recently.The goal of the studies has been classified in different categories. As can be seen, the table lists the used features, identification method, and the behaviors. The target of research done in [23-24] is to identify a particular crowd considering group of participants based on some features, while in [25-27],the main target is to identify the behavior of the crowd whetherthe crowd is normal or aggressive. Sometimes, the moderate style is used in exchange with normal style. Identificationmethodologies used are one of three categories which are neural networks [28-29], classification models [30-31], and statistical models [32-33].

The number of input parameters reported also in the Table I ranges from two to four. We argue that with increasing the number of input parameters to the identification method capturing all of the realistic information would increase the accuracy of crowd behavior detection. In fact, this is the main motivation behind the wok in this paper. Therefore, in our proposal, different parameters are used as input to the identification method namely crowd feature analysis, crowd coherency, social interaction, motion information, randomness in crowd speed, internal chaos level, crowd condition, crowd temporal history, and crowd vibration status along with time stamp. These inputs will be applied to deep learning identification method for crowd behavior identification.

TABLE. I. List of Published Work in the Field of Crowd behavior Identification

Ref	Features / Parameters	Identification Method	Behavior
[15]	- Crowd Speed - Exit Opening	Bayesian Probability	- Normal - Aggressive
[16]	- Speed and velocity - Average motion / Deceleration Profile - Turning speed vs. Radius of turn	support vector machine (SVM)	- Identify general behavior
[17]	- acceleration, - the speed, - Social interaction	neural network	- Crowd Scene analysis
[18]	- Chaotic behavior	Probabilistic ARX model	- Density estimation
[23]	- acceleration - randomness - the crowd speed - turning behavior	Decision Tree Clustering algorithm	- Identify coherent groups
[24]	- acceleration relative lane position	Statistical method of a Gaussian mixture model (GMM)	- Identify abnormal behavior
[25]	- Acceleration - Speed - position	fuzzy clustering algorithm	- Event planning
[26]	- position - velocity - Coherency	neural networks	- Crowd dynamics
[27]	- Feature integration	Bayesian network	- Crowd trajectories

III. Proposed Method

This section elaborates on the proposed VANET framework based on IoT architecture [15][34] for crowd analysis. The framework shows how VANET applications and protocols could perfectly fit the IoT architecture for crowd analysis. Fig. 1 shows the layered framework and the name of each layer. It basically consists of seven layers where the bottom layer is the data collection layer. Sensors, odometers, GPS information, etc. are sources of information in this layer. The second layer of the framework is the connectivity layer where different sensors might connect to each other. These sensors need to connect to RSU and RSU might need to connect to the data center. Therefore, all of the connectivity issues could be handled in this layer. This allows the collected information to be exchanged and forwarded to other components in the network until it reaches the data center helping out in decision making process. The collected raw datamight be repeated, unsuitable, noisy, and unformatted. Therefore, layer 3 is responsible for data transformation and cleanness. Layer 4 is responsible for data analysis and storage

since it is expected to have huge information increasing exponentially with time. This huge collected data needs to be aggregated to save the network bandwidth and removing the data redundancy. Many of the effective aggregation as well as fusion methods could be used in layer 5. Layer 6 is the application layer which involves many of the crowd applications as well as related data center applications. The decision making and process collaboration with other components of the system could be handled in layer 7.

Beside the vertical structure, edge software can be source for multiple layers as needed for crowd anomaly detection. For example, security aspects cannot be dedicated to one layer only. Particularly, security is needed at devices level; so they do not get hacked at the hardware level. It is also needed at application layer; so the software cannot be hacked as well. Another example is the scheduling for devices communication.It can be an important factor at devices level, so it can determine when to sense the data. It is also important at data transformation level; so it can be decided when to schedule thedata exchange for crowd monitoring. This allows us to see the model, Fig. 1, from two sides, vertical and horizontal shape. The vertical one has a pre-determined function for each layer and the horizontal one can be an implementation for cross layerfunctions.

Table II briefly compare the VANET model to the IoT models proposed in [15][34]. As can be seen in the Table II, the proposed model perfectly fit the IoT model. For our work related to crowd behavior identification problem, this model can be extended to encode features and connectivity across multiple nodes in the network.

Nowadays, crowd behavior detection is the main focus of industry and significant progress has been made in this regard. In addition, the concept of IoT enforces many of the automation to take place. However, the behavior of crowd itself is not taking that much attention although it is very important for people safety and for the surrounding environment such as pedestrians. However, crowd behavior identification depends on many of the factors. Some of these factors were taking into consideration in the previous works but due to the complexity of the problem, some others were ignored.

Fig. 1. VANET Framework based IoT Architecture.

TABLE. II. IOT LAYERS AND THEIR FUNCTIONS

Layer No.	Layer name	Function	Elements in VANET
1	Physical Device and Controller	Collect and sense the data from the surrounding environment	Sensors, odometer
2	Connectivity	Setup the connectivity between the network elements.	Zig bee, Bluetooth
3	Edge Computing	Exchange the collected data via the connected components	Public place unit, Sensor
4	Data Accumulation	Store the collected data in a dedicate data center after it travel though the connected devices	Data center
5	Data Abstraction	Abstract the stored data to ease the arrival of a decision in later layer	Artificial intelligent software
6	Application	Report the abstract to the user via application	Any mobile application
7	Collaboration and Process	The needed action based on the collected data can be processed in this stage.	People, business, sign in the places.

Here, we tend to take different parameters as input for accurate decision making about the crowd behavior including crowd coherency, social interaction, motion information, randomness in crowd speed, internal chaos level, crowd condition, crowd temporal history, and crowd vibration status along with time stamp. These parameters are easy to be captured by sensors in the data collection layer. In addition, the crowd behavior decision could be one of two behaviors namely aggressive and normal. Here, we extend the definition of aggressive crowd that is driven by abnormal motion patterns. Based on these concepts, investigating a well-designed solution to classify crowd behavior is a challenging problem. Part of the challenge is to capture data such as computer vision based features out of a video stream. In fact, this parameter requires accurate image processing algorithm as well as it might take long processing time. In the next section, we will explore the details of the proposed methodologies.

Following the VANET model presented in Fig. 1, the crowd behavior identification could be handled based on the same layers. For instance, the raw data is collected through thedata collection layer, then transformed to a suitable format in the transformation layer followed by analysis and storage in thenext layer, decisions have to aggregated in the aggregation layer to be passed to the sensor applications layer. Finally, the data is shared using the collaboration process and through the connectivity layer.

For the paper to be self-content, this section starts by brief description to the used methodologies. The following subsection explains the SIFT algorithm. The next subsection briefly explain two of the deep learning neural networks namely Recurrent Neural Networks (RNN) and Convolution Neural Networks (CNN).

To detect crowd behavior, we compute scale invariant feature transform (SIFT) [35][36] from each video frame of the

crowd video. SIFT based frame matching is a foundation step of many challenging issues in the field of computer vision, including object and situation identification. Besides that, SIFT features have several important aspects that make them significant for associating various frames of the crowd behavior. It is worth noticing that the SIFT features do not change by changing other factors including scale, rotation, and illumination. In addition to that, these features are highly unique, which allows a single feature to be correctly associated with the relevant and matching feature in the consecutive video frame. SIFT features computation from a video frame occurs in four steps which are scale space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor.

Scale-space kernel is mainly the Gaussian function. Therefore, the scale space of a frame of a video is the function L(x, y, σ) that it is generated from the convolution of a variable-scale Gaussian, G(x, y, σ), with the frame, I(x, y) as given in equations (1) and (2).

$$L(x, y,\sigma) = G(x, y,\sigma) * I(x, y) \tag{1}$$

$$G(x, y,) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2}$$

The scale-space extrema in the difference-of-Gaussian function is convolved with the video frame, D(x, y, σ), that is calculated from the difference of two nearby scales separated by a constant multiplicative factor k as provided in equation (3).

$$D(x, y,\sigma) = (G(x, y, k\sigma) - G(x, y,\sigma)) * I(x, y)$$

$$= L(x, y, k\sigma) - L(x, y,\sigma) \tag{3}$$

To find the local maxima and minima in the difference frame D(x, y, σ), each feature point is compared to its eight neighbors in the same frame. It is detected and identified only

if it is larger than all of these neighbors or smaller than all of them. The computation of this operation is not heavy because most similar features will be identified after the first few operations. The important characteristic is to consider the iteration of sampling in the video frame and scale domains that is needed to effectively identify the extrema. To this end, there is no minimum spacing of features that will choose all the extrema because they can be in near vicinity with high probability. Therefore, we must investigate a formulation that trades off efficiency and completeness. In is important to note that the extrema that are in near vicinity are susceptible to small noise of the video frame.

Additionally, the total number of detected features increases with increased sampling of scales and the total number of correct matches also increases. The featurematching with high probability depends substantially on the amount of accurately matched feature points. Hence, it is important to exploit a larger number of scale samples and features. Nevertheless, the load of calculations also increases with this criteria. To this end, the scale space difference of Gaussian function has a large number of extrema and we can identify the most reliable and effective subset even with a coarse sampling of scales.

Preprocessing the video frame before extrema detection significantly ignores the high magnitude frequencies in spatial domain. Hence, the streaming crowd video frame can be magnified to engender more features than were present in the original video frame. Therefore, the size of the crowd video frame is significantly increased by utilizing linear interpolation technique before establishing the first level of the pyramid. The other operations could have been carried out by considering various sets of subpixel-offset kernels on the original video frame. In fact, increasing the size of the video frame presents improved implementation.

Once a keypoint sample is detected by comparing a pixel to its neighbors, the next step is to carry out a matching process in the neighborhood region. In this way, the technique reject points that have low contrast or improperly localized along an edge. For reliability, ignoring keypoints with low contrast does not suffice. The difference-of-Gaussian procedure will present huge magnitude on edges. Therefore, the process is sensitive to small amount of noises in the neighborhood regions. An improperly detected peak in the difference of Gaussian function will have a large principal curvature which can be

calculated from a 2 x 2 Hessian matrix computed at the location and scale of the keypoint as given below:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{D}_{xx} & \boldsymbol{D}_{xy} \\ \boldsymbol{D}_{xy} & \boldsymbol{D}_{yy} \end{bmatrix} \tag{4}$$

The magnitude H is formulated by subtracting neighboring features or keypoints. For each video frame, L(x, y) is calculated using pixel differences as given in equations (5) and (6).

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \tag{5}$$

$$\theta(x, y) = \tan^{-1}((L(x, y+1) - L(x, y-1)) / (L(x+1) - L(x-1, y))) \tag{6}$$

A direction histogram is engendered from the gradient calculations of features within an area around the detected feature in the center. Once the keypoints on the consecutive frames are calculated, we determine the distances among the corresponding keypoints. If the distances of the corresponding keypoints are in a specific range and there are a set of such keypoints, the frame is considered as angry. Otherwise it shows the normal behavior of the driver. In Fig. 2, two frames are presented showing normal and panic crowd behavior from benchmark UMN dataset [37]. Our approach correctly detected the abnormal emotion of the crowd. If there a less than 10 SIFT features matching on the consecutive frames, we consider it panic behavior.

Normal behavior

(b) Panic behavior

After recognizing partially the behavior of the crowd, we feed it as a parameter along with other parameters to Deep Learning model. It is worth noticing that Deep learning is the novel and significant trend in machine learning. It pledges general, powerful, and fast machine learning, moving us one step closer to artificial intelligence. In the Deep learning, input is passed through several non-linearities before being output. In comparison to Deep learning, traditional learning algorithms e.g., decision trees and SVMs and Naive Bayes are shallow. Deep learning excels on challenging problems where the inputs are not a few parameters but instead are parameter values in the form of images. In addition to ability of the deep learning to handle nonlinear data, deep networks also have other capabilities which set them apart from other traditional machine learning models. For example, we can modify them indifferent ways to fit to our problem.

The earlier important progress in Deep Learning was Deep Belief Networks [38-39] to pretrain deep networks. This method investigated that pretraining each layer is better for initial weights instead of random initialization. For example, Deep Belief Networks based on Restricted Boltzmann Machines [40], and Deep Autoencoders based on Autoencoders [41].

Autoencoders is driving more interest since it could be used as a way to pretrain neural networks. In fact, training neural networks is very difficult due to the reasons that the magnitudes of gradients in the lower layers and in higher layersare different. Moreover, the curvature of the objective functionis difficult for stochastic gradient descent to locate the local optimum. Also Deep networks are multiple parameters oriented. That means they can remember training data but do not generalize well. The pretraining process cope with the aforementioned problems. In the pretraining step, a sequence of shallow autoencoders is trained layer by layer. The last layeris trained exploiting the supervised data. Finally, backpropagation is used to fine-tune the entire network using the supervised data. The significant progress in Deep Learningis the use of convolutional neural networks to obtain a paramount improvement. Actually the unsupervised learning inthe autoencoders is less relevant when a lot of labeled data are available. In the convolutional neural networks, a technique called locality structures reduces the number of connections. Also weight sharing is used to reduce the number of connections. A few parameters in the model are constrained tobe equal to each other in the weight sharing. The convolution networks also come with another layer known as the max- pooling layer. In this layer the max value of a selected set of output neurons is calculated from the convolutional layer and itis used as input to higher layers. In the convolution neural networks, explicit computation is performed with all the weights in the forward pass. In the backward pass, the gradient for all the weights is computed based on equation (7).

$$\frac{\partial J}{\partial w_1}, \ldots\ldots, \frac{\partial J}{\partial w_g} \tag{7}$$

$$w_1 = w_1 - \alpha\left(\frac{\partial J}{\partial w_1} + \frac{\partial J}{\partial w_4} + \frac{\partial J}{\partial w_7}\right),$$

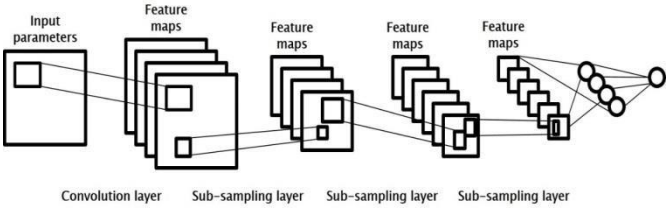
$$w_4 = w_4 - \alpha\left(\frac{\partial J}{\partial w_1} + \frac{\partial J}{\partial w_4} + \frac{\partial J}{\partial w_7}\right),$$

$$w_7 = w_7 - \alpha\left(\frac{\partial J}{\partial w_1} + \frac{\partial J}{\partial w_4} + \frac{\partial J}{\partial w_7}\right), \tag{8}$$

For the max-pooling layer, in the forward pass, the layer that renders the max value is remembered, so that in the backward pass, the gradient for that layer is computed only. InFig. 3 below, the diagram of the convolutional neural network is depicted that shows the layered architecture.

The input to the CNN may have multiple channels. Therefore, the CNN architecture can be modified to work with such input. In this modification, it is important to have a filter that processes multiple channels input. However, the weights are not shared across different channel. Each set of output produced by each filter is called a map. It is also possible to have a CNN architecture with multiple maps. To take this into account, the CNN can be modified to have multiple filters per location. In fact, the input to the CNN is generally two-dimensional. The CNN architecture cope with two-dimensional inputs. For this purpose, each filter has two dimensions. In case the inputs have many channels, then each filter is essentially three dimensional: row-column-channel.

It is also worth to notice that the CNN architecture considers input with fixed size. However, inputs may have many sizes. To deal with that, it is typical to crop the inputs at the center and convert all inputs to the desired size. In fact, many state-of-the-art CNN are sequences of processing blockswhere each block is a combination of convolution, max pooling, and local contrast normalization. The variable size inputs are challenging in some cases. For example, predicting the stock market of any company. One possible solution to dealwith the variable-sized input problem is to use convolutional neural network where the max pooling is applied to all of the output of the filters. However, it does not fix the problem. If the input is variable-sized, the output is fix-sized. The problemwith this situation is that the CNN is invariant to translation. Considering such a large max-pooling, losing position information is inevitable. In the field of image processing and computer vision, translation invariance is acceptable since the output of a system should be invariant to translation. However,in the stock prediction model, this is an unwanted property, because we want to make use of the precise temporal information. The solution to deal with variable-sized inputs is to use a Recurrent Neural Network (RNN).



In the RNN, there are usually three sets of parameters: the input to hidden weights (W), the hidden to hidden weights (U), and the hidden to label weight (V). To this end, all the W's are shared, all the U's are shared and all the V 's are shared. It is due to this sharing property, that the RNN is suitable forvariable-sized inputs. Considering these notations, the hidden states are iteratively calculated as shown in equation (9).

$$f(x) = Vh_T$$
$$h_t = \sigma(Uh_{t-1} + W_{x_t}),$$
$$\cdots\cdots$$
$$h_o = \sigma(Wx_o) \quad (9)$$

The cost function can be minimized then to get the appropriate weights. To calculate the gradient of the RNN, the backpropagation algorithm can be used which is called Backpropagation through time (BPTT). While computing the gradient for the RNN, it could be either large or very small. Therefore, the entire training process will converge slowly. To improve the speed of the training process, it is suggested to truncate the gradient at certain values.

Based on the proposed parameters for crowd behavior identification, we believe that a deep learning technique is suitable to analyze the crowd behaviors. However, deep learning has been widely used in many of the image classifications as well as audio and showed a successful results over different datasets. It is also designed in such way to accommodate certain type of inputs. Therefore, our solution in this paper works in stages, starting by the collected data as shown in Fig. 4.

The input of the designed system is real crowd videos along with other parameters coming from sensors. Videos are framed and the status of the crowd is captured at each second. However, this raw information has to be preprocessed to have equal weights and meaningful information to the deep leaning. Part of the preprocessing is to have weights for each type of parameter. Some other statistics are applied on the collected data for each parameter such as Max, Min, Mean, Median, Variance, and Standard Deviation.

These statistical information expresses the changes in the frame information. Since the collected data will be huge to be used with the deep learning, the crowd behavior can be captured through multiple frames where the collected frames are divided into overlapped segments. Each segment is divided into a number of slices. Again, these slices are overlapped to avoid losing any information during processing. These slices are also considered as deep learning widows.

Another step before providing the input to the selected deep learning model is the fitting and normalization over the training data. The fitting is done by collecting the statistics (mean/stdev) from the training data. Then the testing dataset is also normalized using the statistics calculated from the training dataset. The paper utilizes two different deep learning techniques and their variants which are Convolutional neural networks (CNN) [42] and Recurrent neural networks (RNN) [43]. The dataset is adapted to work with both models during the processing phase. For instance, RNN is better working with sequence learning, the input is formatted accordingly.

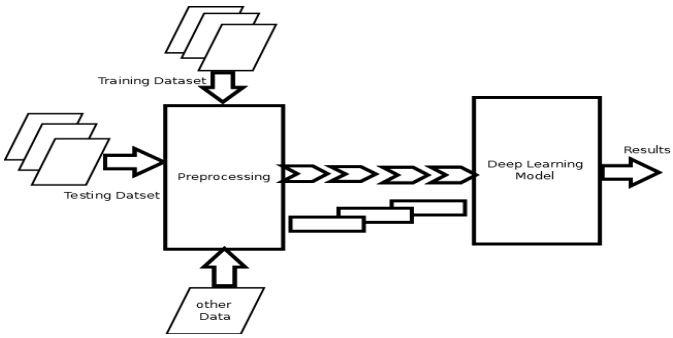


Fig. 4. Solution Process.

IV. RESULTS

This section shows some of our test cases as a proof of concept to the utilization of deep learning with this large number of input parameters. First, in order to have all of the suggested parameters in hand, we consider two popular benchmark datasets which are called UMN [37] and UCSD [44] for panic detection and non-pedestrian entities detection in crowded scenes. Our criteria in measuring the performance of our approach are accuracy, precision, recall, and F1 score. F1 score could be computed by equation (10):

$$F = 2. \frac{1}{\frac{1}{Recall} + \frac{1}{Precsion}} = 2.\frac{Precsio\ .Recall}{Precsion + Recall} \quad (10)$$

We divide the video from each dataset into segments and slices. A frame consists of multiple parameters and statistical values. These data are preprocessed to be normalized and fitted according to the requirements of the deep learning model. The training dataset is 70% of the overall dataset and 30% is the testing dataset. For performance measure Conventional Neural Networks (CNN) with/without pooling layers, Recurrent neural networks (RNN), Pre-trained Recurrent neural networks (PretrainRNN) and Stacked Recurrent neural networks (StackedRNN) are examined. For RNN and StackedRNN, the input data is formatted to have one frame per row. Therefore, each row will be having different values ending with the correct label for training data. The frame information within a segment could be averaged to work on segments instead of frames. However, our experiments are done on frames instead for more accurate results. Rectified linear unit (relu) as an activation function and (softmax) are appended to the recurrentlayer. Two layer StackedRNN (100 neuron each) are developed for the testing purpose as well. For CNN, the input format is different in terms of number of rows and columns. The numberof rows in this case is 48 +8 row representing the main parameters and the generated statistics. However, the number of columns (frames per segment) is 128. Again, 1000 iterationsare set Softmax last layer. CNN network is designed with six layers; three are convolution pooling and the other three are fully connected layers.

Table III shows the results collected for UMN dataset [37] from running the different methods using the training and test parts of the same dataset. We also extract other parameters from the dataset including crowd coherency, social interaction, motion information, randomness in crowd speed, internal chaos level, crowd condition, crowd temporal history, and crowd vibration status along with time stamp. These parameters are

extracted manually with the help of 10 participants. As can be seen from the Table III, CNN with pooling layers overall performance is better than CNN without pooling layers. At the same time, RNN results, in general, is much better than CNN. However, the StackedRNN seems to be the best in terms of Accuracy, Precision, Recall, and F1 score. It was able to reach 76% accuracy. Based on our observations to the results, although the current dataset is representative but we believe that StackedRNN could perform more accurate with larger datasets.

Table IV shows the results collected for UCSD dataset [44] from running the different methods using the training and test parts of the same dataset. Again, we extracted other parameters from this dataset including crowd coherency, social interaction, motion information, randomness in crowd speed, internal chaos level, crowd condition, crowd temporal history, and crowd vibration status along with time stamp. As can be seen again from the Table IV, CNN with pooling layers overall performance is better than CNN without pooling layers. At the same time, RNN results, in general, is much better than CNN. However, the StackedRNN seems to be the best in terms of Accuracy, Precision, Recall, and F1 score. It was able to reach 75% accuracy. Based on our observations to the results, although the current dataset is representative but we believe that StackedRNN could perform more accurate with larger datasets.

We also present some results in comparison with other reference methods including spatio-temporal anomaly model (STA) [45], abnormal crowd behavior model (ACB) [46], and anomalous trajectory detection (ATD) [47]. The results are averaged over both the datasets and presented in Table V. As can be seen, our method outperforms all the reference methods.

TABLE. III. DEEP LEARNING METHODS PERFORMANCE FOR UMN DATASET

Method	Accuracy	Precision	Recall	F1 Score
CNN without pooling layers	0.65	0.54	0.49	0.52
CNN with pooling layers	0.71	0.48	0.54	0.56
RNN	0,67	0.52	0.46	0.50
PretrainRNN	0.69	0.67	0.47	0.53
StackedRNN	**0.76**	**0.71**	**0.60**	**0.66**

TABLE. IV. DEEP LEARNING METHODS PERFORMANCE FOR UCSD DATASET.

Method	Accuracy	Precision	Recall	F1 Score
CNN without pooling layers	0.48	0.47	0.55	0.51
CNN with pooling layers	0.57	0.58	0.49	0.52
RNN	0,60	0.61	0.56	0.61
PretrainRNN	0.65	0.66	0.66	0.63
StackedRNN	**0.75**	**0.75**	**0.67**	**0.69**

TABLE. V. DEEP LEARNING METHODS PERFORMANCE FOR BOTH THE DATASETS

Method	Accuracy	Precision	Recall	F1 Score
STA [45]	0.71	0.66	0.58	0.53
ACB [46]	0.67	0.61	0.51	0.51
ATD [47]	0,72	0.67	0.61	0.62
StackedRNN	**0.75**	**0.73**	**0.63**	**0.67**

Considering these results, we present overall analysis. First, the large number of parameters might affect the decision accuracy. At the same time, not taking all of the parameters into consideration might lead to wrong decision. So, it is a tradeoff between the two cases. Second, some of the parameters might be more effective than others. Part of our future work is to look into these problems and find an elegant solutions. For instance, fuzzy logic control could handle the uncertainty in the collected data and it might be utilized as wellto rank the input parameters. Other extension to this work includes collecting big data related to real world crowd condition and data from sensors installed in different public places. Thus it would enable us to consolidate the current framework with more informative and useful information.

V. DISCUSSION

We have presented a new method for crowd behavior analysis method. We detect and identify panic situations and non-pedestrian entities. It is worth mentioning here that many methods have proposed previously for the same problem as we discussed in the literature review. However, those methods suffer from various problems including lack of generalization capabilities. Moreover, our proposed method is invariant to different key challenges as we mentioned in the introduction section. We carried out detail experimental analysis on two benchmark datasets which are considered very challenging for the same problem in the field of computer vision. Inspired by the concept of the Internet of Things, our method is enriched with robustness to deal with the difficult problem of crowd behavior analysis. In the experimental assessment, we used different performance metrics including accuracy, precision, recall, and F1 Score. Our method showed very performance considering both datasets and performance metrics. In fact, our work can be further extended to many other crowd behavior detection due to its generalization capabilities.

VI. CONCLUSION

In this paper, we have explored a new research direction in crowd behavior detection in smart cities using a novel and a comprehensive framework. Initially, SIFT features are considered to detect crowd behaviors. However, SIFT features are not used as standalone input. To make the framework robust, a number of other parameters are taken into account from the surroundings. Then a deep learning model is trained using the generated training data that detects the crowd behavior in the testing phase. CNN with pooling and without pooling, RNN, pretrained RNN, and Stacked RNN are used as deep learning methodologies. These methods are examined and their performances are validated. The results are promising especially with StackedRNN.

In our future work, we will develop novel deep learning architectures to further enhance the capabilities of our crowd behavior analysis method.

REFERENCES

[1] S. Adam, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. "A simple neural network module for relational reasoning." In Advances in neural information processing systems, pp. 4967-4976. 2017.

[2] C. John M., and T. J. Hastie. "Statistical models." In Statistical Models in S, pp. 13-44. Routledge, 2017.

[3] Building IoT Together by Cisco , [online] https://www.cisco.com/ web/offer/emear/38586/images/Presentations/P11.pdf

[4] W. Xiangqian, Y. Tang, and W. Bu. "Offline text-independent writer identification based on scale invariant feature transform." IEEE Transactions on Information Forensics and Security 9, no. 3 (2014): 526-536.

[5] Lowe, G. "SIFT-the scale invariant feature transform." Int. J 2 (2004): 91-110.

[6] Unusual crowd activity dataset of university of minnesota, available from http: //mha.cs.umn.edu/movies/crowd-activity-all.avi.

[7] H. Geoffrey E. "Deep belief networks." Scholarpedia 4, no. 5 (2009): 5947.

[8] M. Abdel-rahman, G. E. Dahl, and G. Hinton. "Acoustic modeling usingdeep belief networks." IEEE transactions on audio, speech, and languageprocessing 20, no. 1 (2011): 14-22.

[9] N. Vinod, and G. E. Hinton. "Rectified linear units improve restricted boltzmann machines." In Proceedings of the 27th international conference on machine learning (ICML-10), pp. 807-814. 2010.

[10] S. Suvash, A. K. Menon, S. Sanner, and L. Xie. "Autorec: Autoencoders meet collaborative filtering." In Proceedings of the 24th international conference on World Wide Web, pp. 111-112. 2015.

[11] V. Andrea, and K. Lenc. "Matconvnet: Convolutional neural networks for matlab." In Proceedings of the 23rd ACM international conference on Multimedia, pp. 689-692. 2015.

[12] G. Alex, A. R. Mohamed, and G. Hinton. "Speech recognition with deep recurrent neural networks." In 2013 IEEE international conference on acoustics, speech and signal processing, pp. 6645-6649. IEEE, 2013.

[13] Mahadevan V, Li W, Bhalodia V, Vasconcelos N (2010) Anomaly detection in crowded scenes. In: IEEEconference on computer vision and pattern recognition (CVPR), pp 1–8.