Road User Abnormal Trajectory Detection using a Deep Autoencoder

Pankaj Roy and Guillaume-Alexandre Bilodeau

September 4, 2018

Road User Abnormal Trajectory Detection using a Deep Autoencoder

Pankaj Raj Roy and Guillaume-Alexandre Bilodeau

LITIV lab., Dept. of computer & software eng.
Polytechnique Montréal
{pankaj-raj.roy,gabilodeau}@polymtl.ca

Abstract. In this paper, we focus on the development of a method that detects abnormal trajectories of road users at traffic intersections. The main difficulty with this is the fact that there are very few abnormal data and the normal ones are insufficient for the training of any kinds of machine learning model. To tackle these problems, we proposed the solution of using a deep autoencoder network trained solely through augmented data considered as normal. By generating artificial abnormal trajectories, our method is tested on four different outdoor urban users scenes and performs better compared to some classical outlier detection methods.

Keywords: Deep autoencoder · Unsupervised learning · Data augmentation · Abnormal trajectory detection.

1 Introduction

Abnormal event detection has been an intriguing research subject for many years, namely because of the fact that the definition of an abnormal event can be very unclear. The classification of abnormal events can be a challenging task when the number of possible abnormalities can easily exceed our knowledge of abnormal behaviors. Also, it is usually a very tedious job to obtain abnormal data. There can be many different possibilities of abnormalities and some of them can even be subjective. In order to tackle this problem, many authors suggest the hypothesis that all the abnormalities are outliers of the normality distribution. By taking that into account, one can notice that abnormalities are context/scene dependent and that a uniform definition of abnormal events cannot be generalized for all kinds of scenarios other than the assumption of these being the opposite of normal events.

The goal of this paper is to detect abnormal trajectories at road intersections where the objects of interest can be identified as pedestrians, cyclists or vehicles. The problem with trajectory data at intersections is that it does not follow any particular probabilistic rules. Therefore, the best way for classifying them is to use statistical or machine learning approaches that can learn the normal trajectories in an unsupervised manner and be trained to detect outliers, which can be classified as abnormal trajectories. Various statistical approaches can solve the issue of trajectory anomaly detection. But, this problem becomes challenging when the dataset is not large enough for trajectory data classification. In this work, we want to devise a method that can use a small number of trajectory samples and that gives the best precision on the classification of normal and abnormal trajectories compared to other outliers detection methods.

To solve the abnormal event detection problem, our contribution is to propose a deep autoencoder model coupled with a data augmentation method. This allows us to encode information about normal trajectories while removing irrelevant information. Results show that our proposed method outperforms classical methods such as one-class support vector machine (SVM).

2 Related work

In the work of Mousavi et al. [7], the detection of abnormal crowd behavior is based on Histograms of Oriented Tracklets (HOT), the Latent Dirichlet Allocation (LDA) and an SVM that is used for the classification of frames. A spatio-temporal model based on background subtraction in the form of a co-occurrence matrix is proposed in [1] for detecting objects following suspicious paths. This method cannot detect abnormal events overlapping normal paths. The Mixture of Dynamic Textures (MDT), in [6], is used for the anomaly detection in crowded scenes by modeling the appearance and the representation of dynamics of the scenes. This method is not designed to interpret trajectories. In [2], an interaction energy potentials method is proposed. An SVM is used in order to classify the abnormal activities based on the standard bag-of-word representation of each video clip. In [5], detection is implemented using a dictionary of normal patterns represented by the training features from a region in a video, which are the 3D gradient features of a spatial-temporal cubes of five frames. This method cannot be used to interpret trajectories. The covariance matrix for optical flow based feature is applied in [13]. In [12], the authors demonstrated how a Convolutional Neural Network (CNN), a Long Short Term Memory (LSTM) and an SVM can be learned from very few videos and used for detecting abnormal events. Even though this method achieves 95 % accuracy, the abnormal activity detection is only limited to binary classification and is not applicable to abnormal trajectory. The convolutional auto-encoder (CAE), proposed in [10], learns the signature of normal events with the training data. This method takes the frames, the appearance features extracted via Canny edge detector and the motion features from optical flow in the input of the CAE model and outputs the regularization of reconstruction errors (RRE) for each frame.

Abnormal event detection is fundamentally an outlier detection problem. Therefore, any classical outlier detection approach can be used. In [11], a One-Class SVM algorithm is proposed for unlabeled data, which can be used for the novelty detection. This latter form of detection is basically an outlier detector that does not require any abnormal data during the training process of the classification model [9]. In [4], the Isolation Forest method isolates anomalies instead of profiling normal points and also works well with large datasets.

3 Proposed Method

Before applying any statistical or machine learning methods for the detection of outliers, the input data is processed in order to extract its trajectories. Unlike the standard classification problem where each frame of a video represents a sample, in the case of trajectory, multiple frames are needed to define a trajectory sample. Even for thousands of frames, the number of trajectory samples extracted will be insufficient for training properly any statistical or machine learning methods, especially Neural Networks (NN) based methods. Therefore, it is necessary to apply data augmentation for increasing the number of trajectory samples. The general idea behind our abnormal trajectory detection is the following (see Figure 1). First, tracking results are used to extract trajectories assumed to be normal. Then, data augmentation techniques are used in order to increase the number of normal trajectories. A deep autoencoder (DAE) is then trained to learn normal trajectories and classification can be applied on new data.

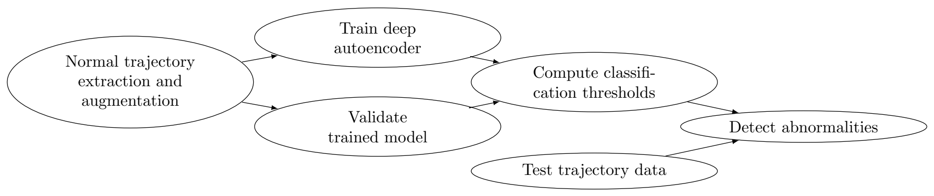


Fig. 1. The main steps of our abnormal trajectory detection method.

3.1 Background on Autoencoders and Deep Autoencoders

The simplest form of autoencoder (AE), called Vanilla Autoencoder (VAE), can be used for the abnormal trajectory detection. Regular AEs are made out of fully-connected NNs that are configured to reproduce the inputs to the outputs of the network. They follow an unsupervised learning method which does not require input data to be labeled. First, this type of NNs compresses the original input into a latent-space representation, which is the space that lies in the bottleneck (hidden) layer, also known as Compressed Feature Vector (CFV). This latter form of representation is produced by an encoding function representing the encoder. Then, the decoder side of this NN reconstructs the CFV input into the output resembling the original input of the encoder side. Figure 2 shows the simplest form of an AE, which is basically made out of three layers, one input, one hidden layer (CFV) and one reconstructed output.

The advantage of utilizing this kind of NNs is that it can be used for learning the normality of the input data. By using only normal data for training of the AE network, the encoder and the decoder are adapted for the normal data and will produce a reconstructed data where the Mean Square Error (MSE) with the

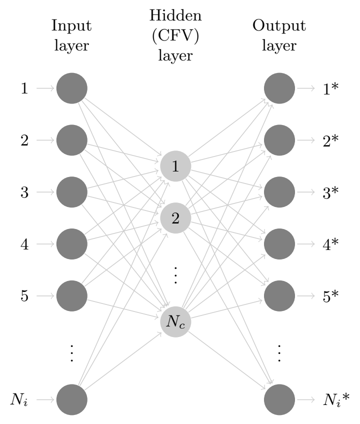


Fig. 2. Diagram showing a Vanilla Autoencoder.

original one varies within some specific defined range, which will be elaborated in the next subsection. When testing the trained AE with abnormal input data, it will produce a reconstruction error (RMSE) exceeding the threshold value calculated from the trained and validated RMSE values.

Deep Autoencoder (DAE) Even though the VAE can be used for the detection of anomalies, intuitively, it is not enough for learning the complex nature of normal trajectories in road intersections. By adding multiple hidden layers in the AE network, the model is able to learn a more complex feature representation, which can be useful for classifying more realistic abnormal trajectories.

3.2 Abnormal Event Detection with a Deep Autoencoder

Algorithms 1 and 2 summarize the main steps required for the training of the DAE model and for detecting abnormalities using the trained network. During the training process of the DAE model, there are two types of validation samples: one that is used internally through cross-validation during the fitting of the networks with the input training set, called va_{cv} samples, and the other called va_e is used externally to validate the scoring of the normality of normal input data. Note that the cross-validation data is necessary during the training process of the network in order to prevent over-fitting.

The scores S are based of the MSE between normalized original z and reconstructed output $\hat{z}$ for each sample, as shown in equation 1.

$$S = \mathrm{MSE}\left(z, \hat{z}\right) \tag{1}$$

The threshold value τ is learned from the data and is used for separating normal from abnormal data. It is determined through equation 2, where STD

Algorithm 1 Training of the Deep Autoencoder Model.

```
Input: normal data $D_n$

1. Shuffle the samples of $D_n$.
2. Split the shuffled samples of $D_n$ into training $tr$ and validation sets $va$.
3. Build the DAE model.
4. Apply normalization scaling technique to the training $tr$ and validation sets $va$ samples.
5. Fit the model with $tr$ samples by using $P_{va_{cv}}$ % of it as the cross-validation set $va_{cv}$ in order to avoid over-fitting of the model.
6. Get the scores $S_{tr}$ of the fitted model with $tr$ samples.
7. Get the scores $S_{va}$ of the fitted model with $va$ samples.
8. Compute the threshold $\tau$ with the training and validation scores ($S_{tr}$ and $S_{va}$).
```

Algorithm 2 Detection of Abnormal Trajectories using Trained Autoencoder.

```
Input: Normal and abnormal data $D_{an}$, trained DAE model $i$
Output: classification decisions

1. Apply normalization scaling method to the test trajectory samples in $D_{an}$ between 0 and 1 ($z_{scaled}$).
2. Open the saved trained model $i$ and the corresponding computed threshold value $\tau$.
3. Get the scores $S_z$ of the trained model with sample $z$ in $D_{an}$.
4. Detect the abnormalities using $S_z$ and the previously computed threshold $\tau$ and output the results.
```

stands for standard deviation, and, S_{tr} and S_{va} are the scores for the training and the validation sets respectively.

$$\tau = \text{mean}\left(S_{tr}\right) + \text{mean}\left(S_{va}\right) + 3 \times \left(\text{STD}\left(S_{tr}\right) + \text{STD}\left(S_{va}\right)\right) \tag{2}$$

Then, the following rule applies for the classification C of a trajectory sample z:

$$C(z) = \begin{cases} \text{Normal}, & \text{if } S_z \leq \tau \\ \text{Abnormal}, & \text{otherwise}, \end{cases} \tag{3}$$

where S_z is the score obtained by applying the trained DAE model to the trajectory sample z.

4 Experiments

4.1 Deep Autoencoder Implementation Details

The table 1 presents the values of hyper-parameters that resulted in the best convergence of the deep autoencoder network. The implementation of the DAE was done using Keras Python package.

Table 1. Hyper-parameters used for the training of the DAE.

Parameter	Value	Definition
H_i	125	Input trajectory sample size
H_{h_1}	128	Number of units in the first hidden layer
H_{h_2}	64	Number of units in the second hidden layer
H_{h_3}	32	Number of units in the third hidden layer
H_{h_4}	16	Number of units in the fourth hidden layer
H_c	8	Number of units in the CFV layer
N_{batch}	128	Batch size used each time
N_{epoch}	100	Number of epochs
optimiser	RMSprop	RMSProp optimizer
σ	0.001	Default learning rate of RMSProp optimizer
loss	MSE	Mean Squared Error

Notice that the first hidden layer in the encoder side of the DAE has a number of units which is slightly greater than the input size. This configuration helps to transform the input size into a number that is a power of two. In fact, by having large layer size in the first hidden layer, the network learns a more detail representation of the input data. Then, by having decreasing unit sizes of power of two in the subsequent layers, the CFV layer gets a more abstract representations of the normal trajectory samples, which can help to detect more complex level of outliers. Also, note that the hidden layers use ReLu activation and the output layer use Sigmoid.

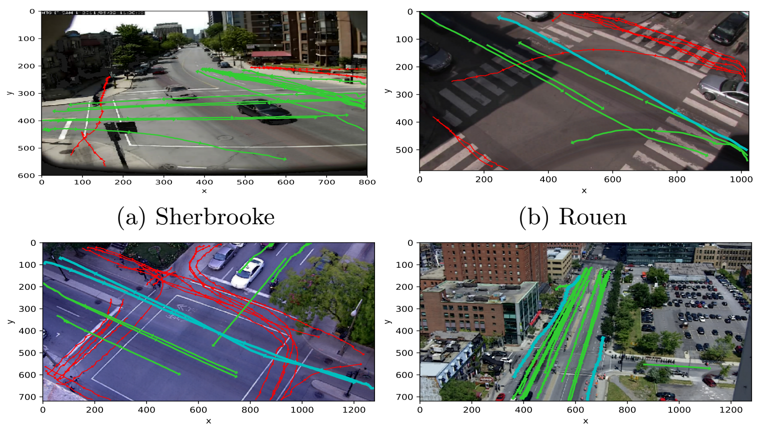
Before feeding the input data into the network, it is scaled with a min and max scaler of 0 and 1. We apply a normalization scaling technique, because the input trajectory data does not follow any specific probabilistic distribution. Also, by scaling the input data before feeding it to the DAE network, we avoid the possibility of having the knock-on effect on the ability to learn the normality of the training data.

4.2 Experimental Protocol

We applied N iterations of Repeated random sub-sampling validation for validating our model, because the shuffling of the input normal data, combined with the splitting into training and validation sets, has an impact on the accuracy of the normality/abnormality detection. Therefore, by doing N iterations, we can determine the average of our method's performance to assess its power of generalization and get the best model giving the highest values of True Positive (TP) and True Negative (TN), and the lowest values of False Positive (FP) and False Negative (FN). We compared our method with OC-SVM and IF that are implemented in `sklearn` Python package [8].

4.3 Input Data, Data Augmentation and Processing

We used the Urban Tracker Dataset [3] that contains four different types of scenarios involving pedestrians, vehicles (including cars and large vehicles) and bikes. Figure 3 shows all the original trajectories of the four scenarios, all of which are considered as ”normal” trajectories.



(a) Sherbrooke (b) Rouen

(c) St-Marc (d) René-Lévesque

Fig. 3. Original annotated trajectories of the Urban Tracker Dataset. Red, green and blue represent pedestrians, cars and bikes respectively.

The Sherbrooke video contains the trajectories of 15 cars and 5 pedestrians. The Rouen video includes 4 vehicles, 1 bike and 11 pedestrians. For the St-Marc video, 1000 frames were chosen, with 7 vehicles, 2 bicycles and 19 pedestrians. Finally, the René-Lévesque video includes 29 vehicles and 2 bikes. We used the ground truth locations of the moving objects for every frames of the urban intersection videos. These locations are composed of bounding box positions. Algorithm 3 describes the fundamental steps for the extraction and augmentation of the trajectories.

Note that we choose to generate trajectory samples by applying a sliding-window approach in which the trajectory window, composed of 31 positions, slides the complete trajectory of an object with a stride of 10 frames. This is done to learn the continuity of the trajectory and therefore prevents the network to learn the wrong representation of trajectory coordinates.

4.4 Abnormal Trajectory Generation

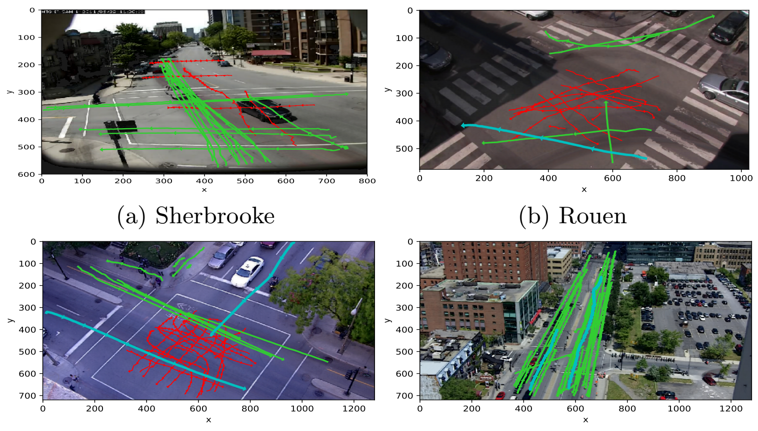
For testing the trained model, abnormal trajectory data is needed. Abnormal events can be a different trajectory path for a car, or a car following the path of pedestrians. For demonstrating the validity of our method, we have generated two types of abnormal data, one which consists of straight lines with constant velocities and the other, more realistic abnormal trajectories inspired by the real normal ones. The realistic ones are resulting from the transformation of the

Algorithm 3 Trajectory Extraction and Augmentation.

Input: bounding boxes
Output: trajectory samples

1. Extract the positions x and y corresponding to the center of the bounding box positions and its related velocities v_x and v_y of the object n.
2. Generate 50 augmented trajectories by randomly generating positions x_a and y_a around the real ones x and y and its related velocities v_{x_a} and v_{y_a}.
3. Decompose the extracted and augmented trajectories into sub trajectory frames composed of 31 positions and velocities.
   - Stretch the complete trajectory of the object n in order for it to be decomposable according to the specific sliding stride value.
4. Add the appropriate label depending on the type of the object. The label "0" identifies a pedestrian, "1" is for a car and "2" specifies a bike.
5. Flatten all the sub trajectories in the following Packed format (125 elements): $[\text{label}, x_1, y_1, v_{x_1}, v_{y_1}, x_2, y_2, v_{x_2}, v_{y_2}, ..., x_{31}, y_{31}, v_{x_{31}}, v_{y_{31}}]$.

original trajectories, which keeps the same degree of fluctuation in the positions and the velocities. The figures 4 shows some of the generated abnormalities.



(a) Sherbrooke (b) Rouen

(c) St-Marc (d) René-Lévesque

Fig. 4. Generated realistic abnormal trajectories. Red, green and blue represent pedestrians, cars and bikes respectively.

4.5 Experimental Results

Table 2 presents the results obtained by applying the trained models on the normal and abnormal trajectories of the dataset. The true positive rate (TPR) and the false positive rate (FPR) are defined as the detected normality/abnormality from normal/abnormal and abnormal/normal samples respectively. Here, we only considered the realistic version of the generated abnormal data. Also, note that the data augmentation technique is applied before training each of these models.

Table 2. Results obtained by applying the trained model on trajectory samples. Boldface values indicate the best results. Status: label indicating normal/abnormal samples, Size: number of samples, OC-SVM: One-class SVM, IF: Isolation forest, VAE: Vanilla AE, DEA: Deep AE.

			Method (%)							
			OC-SVM		IF		VAE		DAE (ours)	
Data	Status	Size	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
Sherb.	normal	20606	90	83	89	82	99	**98**	99	**20**
	abnormal	406	17	10	18	11	**2**	1	**80**	1
Rouen	normal	11884	90	87	90	91	**100**	**96**	**100**	**15**
	abnormal	234	13	10	9	10	**4**	**0**	**85**	**0**
St-Marc	normal	40139	90	82	90	79	**99**	99	**99**	**68**
	abnormal	789	18	10	21	10	1	**1**	**32**	**1**
Rene-L.	normal	45341	90	80	89	80	**99**	99	**99**	**61**
	abnormal	891	20	10	20	10	1	**1**	**39**	**1**

Globally, our method outperforms others in terms of TPR and FPR in both normal and abnormal samples. Specifically, DAE gives the smallest values of FPR compared to others. Also, the results show clearly that DAE distinguishes better the normal and abnormal data compared to VAE. Therefore, it is necessary to have multiple layers in AE network in order for it to avoid getting over-fitted by the training normal data.

5 Conclusion

In this paper, we studied the detection of abnormal trajectories in common traffic scenarios. Considering the hypothesis of abnormalities behaving as outliers, we have proposed a method with a DAE which uses only normal data in the training process. We also applied an automated data augmentation technique for increasing the number of training samples. By generating interactively abnormal realistic trajectories, our method, compared to others like OC-SVM, IF and VAE, yielded the best performance in terms of TPR and FPR of the normal/abnormal detection in most videos.

Acknowledgement

This research was supported by a grant from IVADO funding program for fundamental research projects.

References

1. Benezeth, Y., Jodoin, P.M., Saligrama, V., Rosenberger, C.: Abnormal events detection based on spatio-temporal co-occurences. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2458–2465 (June 2009). https://doi.org/10.1109/CVPR.2009.5206686
2. Cui, X., Liu, Q., Gao, M., Metaxas, D.N.: Abnormal detection using interaction energy potentials. In: CVPR 2011. pp. 3161–3167 (June 2011). https://doi.org/10.1109/CVPR.2011.5995558
3. Jodoin, J.P., Bilodeau, G.A., Saunier, N.: Urban tracker: Multiple object tracking in urban mixed traffic. In: IEEE Winter Conference on Applications of Computer Vision. pp. 885–892 (March 2014). https://doi.org/10.1109/WACV.2014.6836010
4. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining. pp. 413–422 (Dec 2008). https://doi.org/10.1109/ICDM.2008.17
5. Lu, C., Shi, J., Jia, J.: Abnormal event detection at 150 fps in matlab. In: 2013 IEEE International Conference on Computer Vision. pp. 2720–2727 (Dec 2013). https://doi.org/10.1109/ICCV.2013.338
6. Mahadevan, V., Li, W., Bhalodia, V., Vasconcelos, N.: Anomaly detection in crowded scenes. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1975–1981 (June 2010). https://doi.org/10.1109/CVPR.2010.5539872
7. Mousavi, H., Mohammadi, S., Perina, A., Chellali, R., Murino, V.: Analyzing tracklets for the detection of abnormal crowd behavior. In: 2015 IEEE Winter Conference on Applications of Computer Vision. pp. 148–155 (Jan 2015). https://doi.org/10.1109/WACV.2015.27
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
9. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. Signal Processing **99**, 215 – 249 (2014). https://doi.org/https://doi.org/10.1016/j.sigpro.2013.12.026, http://www.sciencedirect.com/science/article/pii/S016516841300515X
10. Ribeiro, M., Lazzaretti, A.E., Lopes, H.S.: A study of deep convolutional auto-encoders for anomaly detection in videos. Pattern Recognition Letters **105**, 13 – 22 (2018). https://doi.org/https://doi.org/10.1016/j.patrec.2017.07.016, http://www.sciencedirect.com/science/article/pii/S0167865517302489, machine Learning and Applications in Artificial Intelligence
11. Schölkopf, B., Platt, J.C., Shawe-Taylor, J.C., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (Jul 2001). https://doi.org/10.1162/089976601750264965, https://doi.org/10.1162/089976601750264965
12. Vignesh, K., Yadav, G., Sethi, A.: Abnormal event detection on bmtt-pets 2017 surveillance challenge. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2161–2168 (July 2017). https://doi.org/10.1109/CVPRW.2017.268
13. Wang, T., Qiao, M., Zhu, A., Niu, Y., Li, C., Snoussi, H.: Abnormal event detection via covariance matrix for optical flow based feature. Multimedia Tools and Applications (Nov 2017). https://doi.org/10.1007/s11042-017-5309-2, https://doi.org/10.1007/s11042-017-5309-2