Data Transformer for Anomalous Trajectory Detection

Hsuan-Jen Pan and Wen-Jiin Tsai

Abstract—Anomaly detection is an important task in many traffic applications. Methods based on convolutional neural networks reach state-of-the-art accuracy; however, they typically rely on supervised training with large labeled data and the trained network is only applicable to the intersection that the training data are collected from. Considering that anomaly data are generally hard to obtain, we present data transformation methods for converting data obtained from one intersection to other intersections to mitigate the effort of training data collection. We demonstrate our methods on the task of anomalous trajectory detection and leverage an unsupervised method that require only normal trajectories for network training. We proposed a General model and a Universal model for our transformation methods. The General model focuses on saving data collection effort; while the Universal model aims at training a universal network for being used by other intersections. We evaluated our methods on the dataset with trajectories collected from GTA V virtual world. The experimental results show that with significant reduction in data collecting and network training efforts, our methods still can achieve state-of-the-art accuracy for anomalous trajectory detection.

***Keywords*—**Anomaly detection, trajectory, data transformation, variational auto-encoder.

I. Introduction

ANOMALY detection is the task designed to find out the abnormal data in a large volumes of data, where the data can be objects, classes, or events. Detection of anomalous trajectories has recently attracted extensive research attention due to that it is a fundamental building block to applications such as traffic violation detection [1, 2, 3], traffic accident detection [4, 5], abnormal crowd behavior detection [6], etc. Trajectory anomaly detection is a challenging task due to the fact that the "anomaly" is usually hard to define. In general, an event is considered to be an "anomaly" when it occurs rarely, or unexpected. Although some anomalous trajectory detection methods have been proposed [4, 5, 7], many of them are based upon the trajectory data from global positioning systems (GPS) and are not suitable for intersection surveillance. We dedicated this paper on the vehicle trajectories generated from the video captured by the street cameras.

With the advance in deep learning technology, Convolutional Neural Networks (CNNs) have been widely used in many fields, such as image classification, object detection, segmentation, etc. Recent researches also employed them in anomaly detection. Anomaly detection networks can be divided into two categories: supervised and unsupervised. The supervised methods such as classifier [8] aims to construct a classification model to distinguish the outlier. The unsupervised methods, like auto-encoder [9], only need to collect normal data. The auto-encoder method utilizes encoder and decoder to reconstruct the input data. If the input data is similar to the training data, the reconstructed data from the decoder will be almost identical to the input. Otherwise, the difference between input and the reconstructed data will be huge. Therefore, the difference can be used to distinguish whether the input is anomaly data or not. Kumaran *et al*. took advantages of the above two methods by proposed a method called hybrid CNNVAE [4]. It integrates both the classifier and the auto-encoder in one module and train them together. Sabokrou *et al*. transplanted the concept of generative adversarial network. Regarding auto-encoder as a generator, classifier as a discriminator, and proposed a method called Adversarially learned one-class classifier for novelty detection [1]. No matter supervised or unsupervised methods, they all rely on the availability of sufficient data. However, most trajectory datasets have very limited data, especially the anomaly data. The problem of limited training data is exacerbated by wide variations in the trajectories across different intersections. This means that a network that was trained by the data collected from one intersection cannot be used directly to test for another intersection. Therefore, many approaches evaluated their performance on the datasets where training and testing trajectories are all collected from the same intersection [4, 5, 7]. Namely, to apply their models to many different intersections, we need to do training data collection and model training for each of these intersections. Obviously, this requires considerable effort and is time consuming, making them hard for practical use.

We address the problem of limited training data by trajectory transformation. That is, the trajectories collected from one intersection are transformed to simulate the trajectories at another intersection. Trajectory transformation is a challenging task for several reasons. Firstly, intersections exhibit intrinsic structure variations. The number of roads across the intersection, the angles between every two intersecting roads, and the number of lanes on the roads can vary, resulting in the fact that the trajectories across different intersections can appear with different geometries. Secondly, data capturing process exhibits substantial variations. The heights, distances, and shooting angles across street cameras can be different. This means that the captured trajectories can appear with different lengths, shapes, orientations and positions on the images.

This paper leverages an unsupervised anomaly detection model which is trained by normal trajectories only, namely, no data labelling is required. To reduce the effort of data collection and model training for each intersection, we also proposed a

General model and a Universal model for data transformation. The General model allows training data to be shared by similar intersections and two transformation methods, one-for-all and group-wise, are proposed. The Universal model allows the trained network to be shared by intersections and two transformation methods, bird's eye view and on-line group-wise are proposed. The experimental result shows that both General model and Universal model can achieve their goals at a very low degradation in the detection accuracy. The main contributions of this paper are summarized as follows.

- To our best knowledge, this is the first paper that proposed data transformation methods in the context of anomalous trajectory detection.
- It proposed the concept of using multiple transformation matrixes to cope with the variation problem in intersections.
- This paper presents several transformation methods. The evaluation results show that the group-wise transformation which uses multiple matrixes performed the best.
- The proposed Universal model shows the concept that sharing a single trained network with intersections can be achieved by applying transformation in the testing stage
- It built a dataset called *GTA-InterSec* for performance evaluation. The dataset contains trajectories collected from four intersections in the virtual world.

II. Proposed Method

A. Overview of the Method

In this paper, three anomalous trajectory detection models are proposed, which are *Basic model*, *General model*, and *Universal model*. The Basic model is similar to most existing methods where each intersection needs to collect training data for training its own network. The General model is designed to mitigate the effort of collecting training data for each individual intersection. For this purpose, the trajectories collected from one intersection are converted to imitate the trajectories on other intersections such that these intersections can use them to train their networks without collecting training data again. In the General model, two transformation methods are proposed: *one-for-all* and *group-wise* transformations, , which are detailed in section 3.C. As for the Universal model, it is designed to reduce the burden not only for data collection, but also for network training. Different from General model where each intersection needs its own network model for training and testing, the Universal model aims at building a single network model shared by different intersections. To achieve this goal, each testing trajectory is on-line transformed to imitate training trajectories before it is fed to the network. Two transformation methods are proposed for the Universal model: *bird's eye-view* and *on-line group-wise* transformations, which are detailed in section 3.D. Table I summarizes the comparison of the three models.

TABLE I
COMPARISON OF THREE DIFFERENT MODELS

	Intersection 1	Intersection 2	Intersection 3
Basic model	- Collect training data - Training - Testing	- Collect training data - Training - Testing	- Collect training data - Training - Testing
General model	- Training - Testing	- Training - Testing	- Training - Testing
Universal model	- Testing	- Testing	- Testing

B. Basic Model

The Basic model consists of three components: Multiple Object Tracking (MOT) network, Location-time representation, and Variational Auto-Encoder (VAE) network, as depicted in Fig. 1. The MOT network takes a video or a sequence of frames as input, detects vehicles in each frame, associate vehicles between successive frames to form each track, and finally output a sequence of location information for each track. Without loss of generality, any solution that can produce tracking information for the input video can also be used in our Basic model.

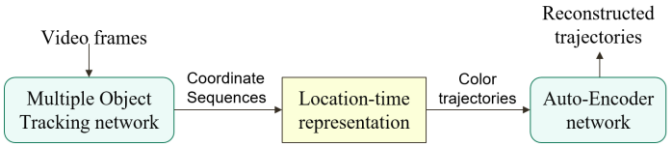
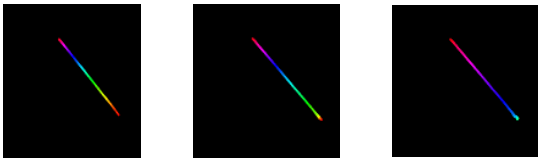


Fig. 1 Block diagram of the Basic model

The sequences of locations output from the MOT network not only consists of spatial information but also temporal information. To fully utilize both of them, we adopt the method proposed by Kumaran et al. [7] to represent the trajectory of each track. The method converts the sequence of locations to the spatial coordinates on images and uses color to convey temporal information. The coloring rule is defined as follows.

$$Hue(x_t,y_t) = t / T$$

where T is the total time of trajectory, x_t, y_t represents the vehicle location at time step t. Both T and t are calculated in terms of frames along temporal direction. *Hue (x, y)* is the color that the pixel at (x, y) will be presented. In other words, it uses different colors to represent for the time series. The color starts from "red" at the beginning of the trajectory and, through the changes of *Hue*, it indicates different time points. Some examples are depicted in Fig.2, where constant color-change along the trajectory means that the vehicle is in equal speed; slow color-change means the vehicle is in acceleration; while the fast color-change means the vehicle is slowing down or even stops. Each track is represented on a single image.


(a) Constant speed (b) Acceleration (c) slow down (or stop)
Fig. 2 Example of location-time representation

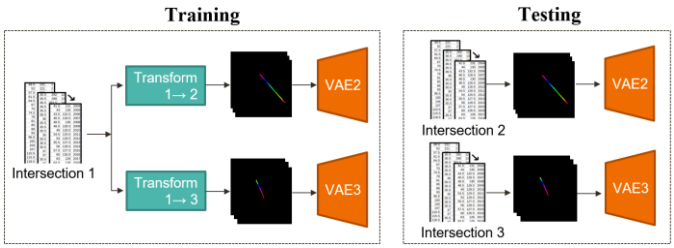
After location-time representation, the trajectories are fed into a network for anomaly detection. To reduce the effort of

data labeling for training, our Basic model leverages an unsupervised method, variational auto-encoder (VAE) network that requires only normal trajectories for training. The VAE is similar to the one in [7], which consists of two parts: encoder and decoder. The encoder takes a trajectory τ as input and produces a vector z in latent space, whereas the decoder aims to reconstruct the τ based on the corresponding z. For a trajectory τ, the loss function l_τ is defined by log-likelihood and the Kullback-Leibler Divergence (KLD) between the input and the reconstructed trajectories. Once trained, a threshold is used for the VAE to detect anomaly. If the reconstruction error is lower than the threshold, the input trajectory is regarded as a normal trajectory; otherwise, it is an anomaly. The VAE in [7] used the average loss during training as the threshold. However, since long trajectories tend to have high loss, we modified their loss function by taking trajectory lengths into considerations. The new loss l_τ* is defined as l_τ*=l_τ /$L(\tau)$, where $L(\tau)$ is the length of trajectory τ.

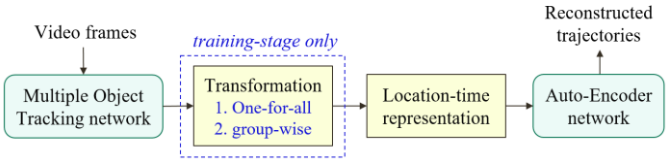
It is worth mentioning that since the VAE reconstruction error is used as the anomaly detection criteria, the to-be-tested trajectory must be very similar to some of the trajectories that the VAE has learned so that it can be reconstructed well with low loss. This indicates that each intersection should collect its own trajectories for both training and testing because the variations in the intersection structures and capturing process make the trajectories coming from different intersections look different and hard to be shared.

C. General Model

The General model aims at eliminating the needs that each intersection should collect its training data. For this purpose, transformation methods are proposed to convert the training data from one intersection to others. Fig. 3(a) shows a simple scenario of using General model, where the training data are collected from intersection1. After different transformations, the data are used to train VAE2 and VAE3, respectively. Then, VAE2 and VAE3 can be used detect anomaly for intersection2 and intersection3, respectively. As observed in this case, there is no need to collect training data from intersection2 and intersection3. General model is similar to Basic model except that it has a transformation block in between MOT network and space-time representation, as depicted in Fig. 3(b). The transformation is only applied in the training stage and two methods are proposed: *1. one-for-all transformation* and 2. *group-wise transformation*.



(a) A simple scenario



(b) Block diagram

Fig. 3 The proposed General model

One-for-all transformation. In the proposed One-for-all transformation, two intersections (say A and B) are related by a Homography, meaning that we can use a Homography matrix (H) to transform any point from intersection-A to the corresponding point in the intersection-B. The One-for-all transformation uses a single matrix to transform all the training trajectories at intersection-A to simulate the trajectories at intersection-B. The transformation formula is represented as follows.

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

where (x1, y1), and (x2, y2) are the coordinates of the points in intersection1 and intersection2, respectively. H is a 3x3 matrix that has 9 unknown numbers but with 8 degrees of freedom. Hence, it needs at least 4 pairs of points to estimate the H. The more corresponding points are used, the more precise H can be obtained. The selection of the corresponding points depends on the structure of the intersection. For a *n*-way intersection with *m* lanes on each road, we simply choose *m*×*n* end-points (one for each lane) as the corresponding points. As an example in Fig.4 where both intersection-A and intersection-B are four-way intersections, 8 end-points on each intersection are selected. With selected pairs of corresponding points, the transformation matrix *H* can be solved and this single matrix is then used to transform all the training trajectories at intersection-A to intersection-B. In this way, intersection-B can train its network by using the transformed data directly without the need to collect training data again.

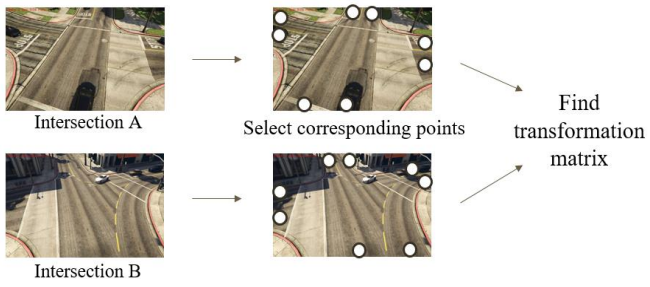


Fig. 4 Example of one-for-all transformation

Off-line group-wise transformation. Intersections exhibit variations with respect to *n*, the number of roads meeting at the intersection. Even with the same *n*, the angles between intersecting roads can vary. The diversity in topologies makes it hard to transform all the trajectories from one intersection to the other perfectly by using one single transformation matrix. To overcome the problem, *group-wise transformation* was proposed, which utilizes multiple transformation matrixes to

achieve the goal. For group-wise transformation, training trajectories are divided into groups such that those with similar locations and shapes fall into the same group and each group has its own transformation matrix, as depicted in Fig. 5. *K-means clustering* is an unsupervised method and is adopted for grouping. To use *k*-means algorithm, each trajectory is represented by the coordinates of its two-end points, Euclidean distance is adopted, and *k* is selected according to the structure of the intersection. For a typical 4-way intersection with 2 lanes on each way, *k* is set to 12 such that straight, turn-right curve, turn-left curve trajectories on each way will fall into different categories. Fig. 5 shows an example with 12 categories of trajectories for a 4-way intersection. To find the homography matrix, for each group we first choose a pair of trajectories, one at the source intersection and the other at the target, in order to determine the corresponding points. For the source intersection, each group uses the trajectory closest to its group center as the representative; while for the target intersection, we need to collect one trajectory for each group as the representative. Then, for each pair of representative trajectories, we follow the rules below to select corresponding points.

If the pair of trajectories is straight, we simply select their start-points and end-points as the corresponding points. If the pair of trajectories is curve, we select five pair of points as the initial corresponding points which include the start point (P1), end point (P2), the vertex at the maximum curvature (P3), the middle point between P1 and P3, and the middle point between P3 and P2. However, due to the diversity among different intersection structures, five pairs of corresponding points may not be enough to find a good transformation matrix for curving trajectories. To trade-off the performance against the time consumed, a metric is used to evaluate the transformation matrix. If the matrix is not good enough, increases the number of pairs by finding the points near the middle of every two neighboring selected points. A new matrix is then calculated again and evaluated. The above process repeat until a good transformation matrix is found. Fig.6 shows an example, where the circles stands for the initial five corresponding points and the triangles indicates the positions when more points are needed. We use cross-entropy between the transformed source trajectory and the corresponding target trajectory as the evaluation metric to judge the goodness of the transformation matrix.

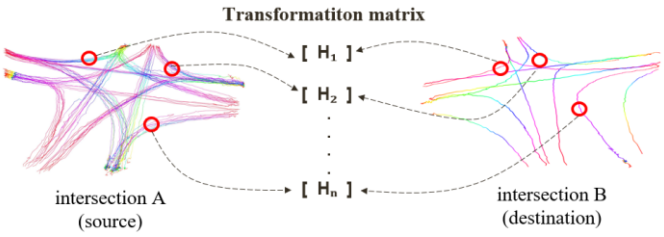


Fig. 5 Off-line group-wise transformation

Fig. 6 Corresponding point selection

D.Universal Model

In General model, even though training data can be shared by similar intersections, each intersection still needs to train its own network for anomalous trajectory detection. To mitigate the efforts of network training, the proposed Universal model aimed at training a single VAE network that can be used by all the similar intersections for anomaly detection. To fulfill this goal, we move the transformation process from training stage to the testing stage. A simple scenario of using Universal model is shown in Fig.7, where training data are collected from intersection1 and trained on VAE_U. At testing stage, on-line transformations are required by intersection2 and intersection3 before feeding their trajectories to VAE_U for anomaly detection. Two on-line transformation methods are proposed, which are *bird's eye-view* and *on-line group-wise transformation*.

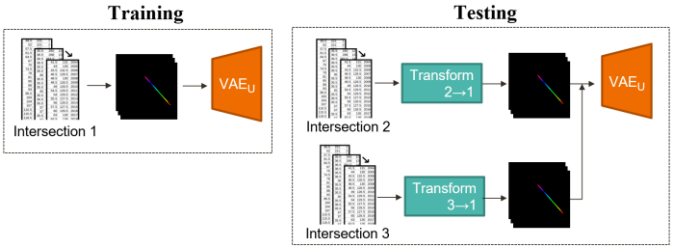


Fig. 7 A simple scenario of using Universal model

Bird's eye-view transformation. The Bird's eye-view transformation aims at eliminating the diversity of shooting angles, heights, and distances across different street cameras by converting all the trajectories to a common bird's eye-view, no matter for training or testing. For this purpose, each intersection requires a matrix to do on-line transformation before testing. For a 4-way intersection, we simply select the four cross-line points as the corresponding points to derive the transformation matrix. An example of the corresponding point selection is illustrated in Fig. 8.

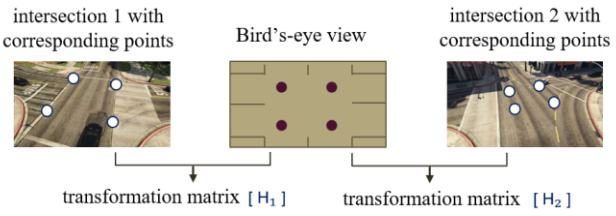


Fig. 8 Example of Bird's-eye view transformation

On-line group-wise transformation. Similar to the off-line group-wise transformation in the General model, the on-line group-wise transformation also utilizes multiple matrixes, one

for each group. The difference between them is that the off-line method is applied in the training stage; while the on-line one is applied in the testing stage. For the on-line group-wise method, a single network is trained using the training data collected from one intersection without any transformation. The training data are clustered using k-means to find the representative trajectory of each group. For any intersection that need anomaly detection, it needs to collect one trajectory sample for each group to derive transformation matrix. The process of deriving the matrix is identical to that used in the off-line method, except that the transformation source and target are interchanged. In the testing time, we have to classify which group the testing trajectory belongs to. Then, transform this testing trajectory using the corresponding transformation matrix. As a result, we can check if this trajectory is an anomaly by using the single VAE model.

III. EXPERIMENTS

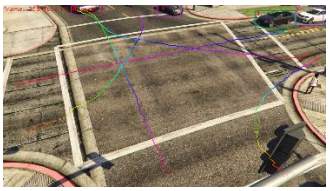
In this section, the proposed anomalous trajectory detection are evaluated. The dataset we used is described first. Then the implementation details and experimental results are presented.

A. GTA-InterSec Dataset

The dataset we used for trajectory anomaly detection is called *GTA-InterSec* which was constructed by ourselves using Grand Theft Auto V (GTA V). The GTA is a virtual world imitating the scenario of San Andreas. To simulate the surveillance cameras at intersections, we fix our view on top of a streetlamp in GTA to capture traffic information at the intersections. We also drive a car in GTA to generate anomalous trajectories. The dataset consists of the videos captured from three different 4-way intersections and one 3-way (T-type) intersection. Fig. 9 shows the snapshots of the intersections. The first intersection contains three video sequences of resolution 1920x1080. Each video is about 840 seconds. It includes 676 normal and 45 anomalous trajectories. We divide the normal trajectories into two parts: 608 trajectories for training and the remaining 68 trajectories for testing. Anomalous trajectories are all for testing. The second intersection contains two video sequences of resolution 1920x1080. Each is about 360 seconds. It includes 67 normal and 45 anomalous trajectories, all for testing. The third intersection contains two video sequences of resolution 1920x1080. Each is about 480 seconds. It includes 84 normal and 51 anomalous trajectories, all for testing. The three-way intersection contains two video sequences of resolution 1920x1080. Each is about 380 seconds. It includes 91 normal and 70 anomalous trajectories, all for testing.


intersection 1 (4-way)


intersection 2 (4-way)


intersection 3 (4-way)


intersection 4 (3-way)

Fig. 9 Snapshot of the four intersections in TGA-InterSec dataset

TABLE II
GTA-INTERSEC DATASET

Intersection	Training		Testing	
	Normal	Anomalous	Normal	Anomalous
Intersection 1	608	-	68	45
Intersection 2	-	-	67	45
Intersection 3	-	-	84	51
Intersection 4	-	-	91	70

B. Implementation Details

With the video captured from GTA, we use a multi-object tracking network, RTMOT [10], to generate sequences of vehicle locations for each track. Since RTMOT was designed for human tracking, we modified its anchor-box sizes for vehicles and re-train it on GTA-InterSec dataset. We labeled 3000 frames of vehicle bounding boxes in GTA-InterSec, 2400 frames for training, and 600 frames for validation. The RTMOT was trained for 30 epochs, with batch size=4 and learning rate of 0.001. The optimizer is SGD. To generate trajectory data, we set confidence threshold to 0.5, Non-maximum Suppression (NMS) threshold to 0.4, and Intersection over Uion (IOU) threshold to 0.5. After training, the RTMOT was fed with all the videos in GTA-InterSec dataset to generate trajectories for the four intersections. Since RTMOT is not a perfect model that can produce tracking information without errors, incomplete trajectories may occur due to tracking target loss and found again. We employ interpolation to make the trajectories smooth and completed. Each trajectory is represented by a sequence of coordinates of the tracked vehicles. The trajectories with the number of coordinates less than 30 are discarded. Since the performance of multiple object tracking is not our concern, any solution that can produce tracking information can also be used.

Before feeding trajectories into the VAE network, each of them is presented on an image of size 120*120. We follow the location-time representation method proposed in [7] to do this as described in the Basic model. The trajectory coordinates are first plotted on the image of size 1920*1080 and then are resized to 120*120.

For the trajectory anomaly detection, the VAE network proposed in [7] was adopted as a reference. We implemented it by using PyTorch. The experiments were executed on a computer with Ubuntu 18.04 64-bits, Intel® Core™ i7-4790 CPU @ 3.60GHz × 8, 32 GB RAM, and one NVIDIA GeForce RTX 2080 Ti GPU. The VAE was trained for 500 epochs with the batch size 20, and the learning rate 0.0005. The optimizer is Adam.

The training data consists of 608 normal trajectories. In order to increase the amount of data for training, data augmentation was applied. The training data was augmented by translation

and scaling. For translation, the trajectories are shifting along x-axis direction for +/- 2 pixels and y-axis direction for +/- 4 pixels. For scaling, we rescale the trajectories in 10 random lengths with at least 30 coordinates in each trajectory.

C. Experimental results

The experiments focused on evaluating the performance of data transformation methods. The General model consists of *one-for-all transformation* (method A) and *category-wise transformation* (method B), while the University model consists of *bird's eye-view transformation* (method C) and *on-line category-wise transformation* (method D). The four methods use 608 trajectories from intersection 1 for training, and 521 trajectories from four intersections for testing, where 113 trajectories are from intersection 1, 112 trajectories from intersection 2, 135 trajectories from intersection 3 and 161 trajectories from intersection 4, as listed in Table II. Since the Basic model does not adopt any transformation, it was evaluated only on intersection 1 where the training data come from. Besides, transformation methods A and C are based on the similarity of intersection structures to find corresponding points. Since the intersection 4 is a 3-way intersection which is different from the intersection 1 which is 4-way, we did not evaluate methods A and C on intersection 4. The detection results are shown in two tables: normal trajectory detection are shown in Table III, while anomalous trajectory detection are in Table IV.

As expected, Basic model on intersection1 exhibits the best performance. In these two tables, it achieves 100% accuracy for both normal and anomalous trajectory detection. The result is due to that for intersection 1, both training and testing data come from the same intersection and thus no transformation has been applied. With appropriate training data augmentation and threshold selected, the VAE can distinguish anomaly from normal trajectories very well. As for the General and Universal models on intersections 2~4, they seem to have large variations in performance. In Table II, methods A and C performed much worse than methods B and D. This is due to that both A and C use one single matrix which cannot meet large variations among all the trajectories. The differences between intersection1 and every other intersection made it hard to transform training trajectories to fit target intersections very well. Group-wise transformations such as methods B and D, however, partition the trajectories with large differences into different groups and then uses different matrixes for transformation. Namely, only similar trajectories will share the same matrix and this greatly solves the variation problem among intersection structures. One thing worth mentioning is that both methods B and D work well for intersection 4 even though it is a 3-way intersection. This demonstrates the robustness of group-wise transformation in handling the structure difference between intersections.

For the comparison between the two category-wise methods, the result in Table II shows that method B performed better than method D. Method D obtained the accuracy rates of 69.6%, 73.8% and 76.9%, which are much lower than 86.5%, 77.3% and 82.4% obtained by method B. The main difference between them is that method B transforms trajectories from intersection 1 to intersections 2~4 at the training stage, while method D transforms trajectories from intersections 2~4 to intersection 1 at the testing stage. Doing transformation at testing stage means that method D needs to know the category of each testing trajectory so that the proper matrix can be applied. However, inappropriate grouping happened sometimes and it caused the wrong matrix to be used, leading to improper transformation and inaccurate prediction. As for Method B, even though improper grouping also happened sometimes, the effects of wrong matrix and improper transformation can be mitigated by abundant training data because the transformation is applied in the training stage. As a result, the network still can be trained robustly enough to do prediction correctly.

Anomalous trajectory detection results are shown in Table III, where transformation methods A, B, C, and D did not show much difference in their accuracy rates. However, the overall anomaly detection performance shown in Table III is better than normal trajectory detection result in Table II, no matter which transformation method was adopted. The reason is that normal trajectories might become anomaly if the transformation did not perform well enough. However, improper transformation is not likely to make anomalous trajectories becoming normal ones, but to make them another form of anomaly. As a consequence, the anomaly detection performance was not much affected by the quality of the transformation that was adopted.

TABLE III
NORMAL TRAJECTORY DETECTION RESULTS

Accuracy ratio	**Basic Model**	**General Model**		**Universal Model**	
		A	B	C	D
Intersection1	68 / 68 (100%)	-	-	-	-
Intersection2	-	18 / 67 (26.8%)	58 / 67 (86.5%)	21 / 56 (37.5%)	46 / 66 (69.6%)
Intersection3	-	23 / 84 (27.3%)	65 / 84 (77.3%)	54 / 83 (65%)	63 / 84 (73.8%)
Intersection4	-	-	75/91 (82.4%)	-	70/91 (76.9%)

TABLE IV
ANOMALOUS TRAJECTORY DETECTION RESULTS

Accuracy ratio	**Basic Model**	**General Model**		**Universal Model**	
		A	B	C	D
Intersection1	45 / 45 (100%)	-	-	-	-
Intersection2	-	44 / 45 (97.7%)	44 / 45 (97.7%)	45 / 45 (100%)	43 / 45 (95.5%)
Intersection3	-	50 / 51 (98%)	51 / 51 (100%)	51 / 51 (100%)	51 / 51 (100%)
Intersection4	-	-	67/70 (95.7%)	-	70/70 (100%)

In general, the straight trajectories are easy to be classified correctly, while the curve trajectories such as turning right and turning left are not. Fig. 10 gives some examples where the input are normal trajectories and the corresponding trajectories reconstructed by VAE networks with different transformation methods are shown. The trajectories with label T means that they were classified correctly, while the ones with label F means they were not. The failed cases tend to be curve or short trajectories. The third column of method D shows a failed case resulted from improper grouping and hence incorrect matrix

adopted.

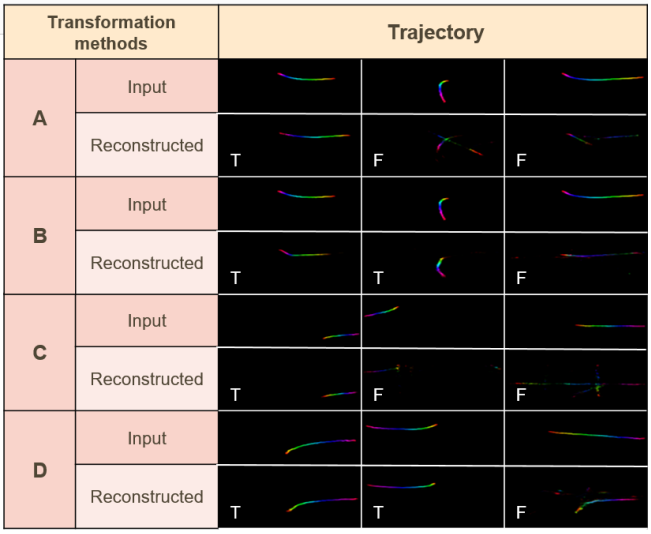


Fig. 10 Examples of trajectory reconstruction

IV. Conclusion

In this paper, we demonstrate that with a well-trained VAE network, anomalous trajectories can be detected with very high accuracy, as the basic model shows in the experimental results. However, it is hard to deploy the system to many intersections because a well-trained network requires sufficient trajectories for training and it costs a lot of efforts if we need to collect training data from each intersection. With data transformation, however, it is possible that the data collected from one intersection can be used by many other intersections. The General model is proposed for this purpose. To further reduce the effort of network training, the Universal model is proposed which allows the trained network to be shared by different intersections. There is a trade-off between the efforts and the prediction accuracy when choosing the model. The Universal model with on-line group-wise transformation can achieve a good result with minimum effort and therefore is the best choice if the efforts and the time consumed are the first priority in considerations. However, if detection accuracy is much more important, then General model with group-wise transformation will be the best.

References

[1] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli. Adversarially learned one-class classifier for novelty detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3379–3388, 2018.

[2] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun. Learning discriminative reconstructions for unsupervised outlier removal. In Proceedings of the IEEE International Conference on Computer Vision, pages 1511–1519, 2015.

[3] D. Nguyen, R. Vadaine, G. Hajduch, René Garello, and R. Fablet. 2019. GeoTrackNet-A Maritime Anomaly Detector using Probabilistic Neural Network Representation of AIS Tracks and A Contrario Detection. CoRR (2019).

[4] S. K. Kumaran, D. P. Dogra, P. P. Roy, and A. Mitra. Video trajectory classification and anomaly detection using hybrid CNNVAE. arXiv: 1812.07203 [cs], pp 1-9, 2018.

[5] Y. Yao, M. Xu, Y. Wang, D. J Crandall, and E. M Atkins. Unsupervised traffic accident detection in first-person videos. IROS, 2019.

[6] M. Sabokrou, M. Fayyaz, M. Fathy, and R. Klette. Deepcascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes. IEEE Transactions on Image Processing, 26(4):1992–2004, 2017.

[7] D. Zhang, L. Nan, Z. H. Zhou, C. Chao, S. Lin, and S. Li. iBAT: Detecting anomalous taxi trajectories from GPS traces. Proc. Int. Conf. Ubiquitous Comput., 2011, pp. 99–108.

[8] A. B. Gardner, A. M. Krieger, G. Vachtsevanos, and B. Litt. One-class novelty detection for seizure analysis from intracranial eeg. JMLR, 7(Jun):1025–1044, 2006.

[9] J. An and S. Cho. 2015. Variational Autoencoder based Anomaly Detection using Reconstruction Probability. Technical Report. SNU Data Mining Center. 1–18 pages.

[10] Z. Wang, L. Zheng, Y. Liu, and S. Wang. Towards real-time multiobject tracking. arXiv:1909.12605, 2019.