A Denoising Hybrid Model for Anomaly Detection in Trajectory Sequences

Maria Liatsikou
Information Technologies Institute, CERTH
Thessaloniki, Greece
maria_liatsikou@iti.gr

Symeon Papadopoulos
Information Technologies Institute, CERTH
Thessaloniki, Greece
papadop@iti.gr

Lazaros Apostolidis
Information Technologies Institute, CERTH
Thessaloniki, Greece
laaposto@iti.gr

Ioannis Kompatsiaris
Information Technologies Institute, CERTH
Thessaloniki, Greece
ikom@iti.gr

ABSTRACT

The advances of mobile technologies have led to massive amounts of trajectory data, i.e. data about tracked routes of moving objects. The detection of anomalies in trajectory data is an evolving research domain, which has applications in traffic management and in public safety, but also in climate research and in animal habit analysis. It is a challenging task due to the existence of non-linear spatiotemporal dependencies. In this work we propose the combination of deep learning techniques with a traditional outlier detection methodology for detecting anomalous trajectories. Two variants of denoising sequential autoencoder architectures are applied for unsupervised anomaly detection in vehicle trajectories, an LSTM-based autoencoder and a Sequence to Sequence LSTM autoencoder. A weighted distance-based loss function is optimized during the training phase, taking into account the impact of trajectory length on the detection outcome. We propose a hybrid architecture for the detection phase, which combines each of the autoencoders with the Local Outlier Factor algorithm – a density–based anomaly detection method – in order to detect anomalies. Our models are evaluated on different variants of synthetic anomalies generated by our dataset. The results indicate a clear performance advantage of our approach compared to competitive baselines.

KEYWORDS

Trajectory Analysis, Anomaly Detection, Denoising Autoencoder, Recurrent Neural Network, Long Short-Term Memory Network, LSTM, Sequence to Sequence

1 INTRODUCTION

The rapid advances in Global Positioning Systems (GPS), combined with those in computation and storage systems allow the large-scale collection and analysis of the digital traces obtained from GPS devices. The knowledge extracted from vehicle or human trajectory data can offer important insights in the fields of traffic monitoring and management, public safety and surveillance. A relevant research domain, which has lately been attracting interest, is anomaly detection in trajectory data.

An anomalous trajectory is one that appears to be different compared to others with respect to some kind of similarity [3]. It is really difficult to uniformly define abnormality, as it is usually context dependent [25]. For instance, vehicle trajectory outliers may happen because of anomalies in traffic, caused by various events, like protests, accidents, physical disasters or because of errors in the GPS devices or of unexpected drivers' behaviors.

In this work, we propose the use of deep learning based unsupervised outlier detection and its combination with a density-based outlier detection algorithm, during the detection phase. Two types of autoencoder architectures are applied, both variants of LSTM-based networks. The aim is the detection of abnormal trajectories, defining them as the ones that the autoencoder fails to reconstruct. Our models are trained using real-world data, including both normal and abnormal trajectories. Since there is no ground truth in our task, in order to evaluate our models on unseen data and compare their performance, we generate different types of abnormal data based on our test set and incorporate them in it. Anomalous trajectories are detected based on their reconstruction errors.

The proposed methods can address the problems of mixed distribution densities in a dataset and the dependencies on suitable similarity metrics. These are common drawbacks of traditional outlier detection approaches, in which user-defined parameters, like distance or density parameters, affect the outcome and are not easily defined for sequence data. In general, deep learning models have several advantages over other methods for outlier detection in sequences as they can effectively address the challenges of feature extraction, high dimensionality and non-linearity [18]. Moreover, there is no need to explicitly describe a normal pattern for a trajectory and to define the type of the anomaly (e.g. a trajectory with not so many neighbors or with small density).

Instead, the proposed method detects trajectories that differ from the normal patterns in a rather abstract manner. In specific, the model is trained to reconstruct most of the trajectories included in a dataset, while it fails on the more irregular ones, which are classified as anomalies.

What is more, the applied architectures are capable of capturing the temporal dependencies in the trajectory data as they are both based on recurrent neural networks (RNNs).

Several recent works have applied neural autoencoders in trajectory outlier detection tasks [4, 9, 21, 25]. They either use feed forward or sequential autoencoders in order to detect outliers in trajectory data. In these works a threshold is set on the errors' values for classifying a trajectory as outlier or not, or a qualitative analysis is conducted on the trajectories with the highest reconstruction errors. Moreover, the autoencoders are trained minimizing a standard loss function, which does not account for the trajectory length.

In this work we make the following contributions:

- We propose a hybrid architecture that combines a *sequential denoising autoencoder* with a *density-based outlier detection model* for unsupervised anomaly detection in trajectories. We apply the Local Outlier Factor (LOF) algorithm on the autoencoder's reconstruction errors of unseen data. In scoring outlier detection techniques an anomaly score is assigned to each instance and the final output is a ranked list of all instances with respect to their scores: those in the highest ranks are detected as anomalies.
- We train two variants of sequential denoising autoencoders as a first step in this hybrid setting, by minimizing a *Haversine distance-based weighted loss function*, effectively accounting for the overall length of each trajectory.
- We test on several anomaly detection tasks with synthetic data, showing important gains in performance using the proposed models against competitive baselines.

2 RELATED WORK

Most trajectory anomaly detection methods rely on *distance*, *density*, *historical similarity* or *classification* [3, 11]. Distance-based methods detect as outliers trajectories with insufficient number of neighbors according to a similarity measure. They detect global outliers and their results highly depend on the suitability of the chosen similarity metric. Methods relying on density classify a trajectory as an outlier if its density is lower than a threshold. They can detect local abnormalities in datasets with varying densities, but are computationally expensive. *Historical similarity* approaches are applied for detecting temporal outliers based on the calculation of changes in historical similarity between data points. *Classification-based* methods include the detection of anomalous trajectories using a motion-classifier (a classifier applied on trajectories represented as sequences of motion features) in a supervised setting, and also machine learning models, either supervised or unsupervised. The latter include Isolation Forests, One-Class SVMs and deep learning architectures, like autoencoders, which have remarkable performance in sequential outlier detection. In this section we present examples of each of these approaches.

A *density-based* algorithm is used by Fontes et al. [10] for trajectory anomaly detection between regions of interest. Spatiotemporal outliers are detected among sub-trajectories based on the concepts of neighborhood, 'standard' sub-trajectory (the neighborhood of which has a minimum density) and synchronization. Although spatiotemporal aspects are considered, this approach refers to trajectories between specific regions, and its outcomes are strongly related to user-specified parameters.

TRAOD [15] is a hybrid (*distance/density-based*) partition-and-detect algorithm for trajectory outlier detection. After a trajectory is partitioned, outliers are detected in its parts using distance and density metrics, so that anomalies are detected both in dense and sparse areas. A trajectory is considered as anomalous if its outlying parts are more than a threshold. Though TRAOD alleviates the problem of local density, its outcome depends on user-defined parameters and it only considers spatial data, without capturing the temporal sequence.

Methods using *historical similarity* between points detect temporal, rather than spatial, outliers. Li et al. [17] propose a framework for detecting temporal outliers in road segments. For every segment and time step, a 'temporal neighborhood vector' is defined, consisting of aggregated historical similarities between specific variables in relation to all other segments. An outlier is detected if there is a drastic change in the similarity values. Though related to our task, this method analyses temporal outliers in road network links rather than outliers in trajectories.

Classification-based methods refer to outlier detection with motion-classifiers or to machine learning classification models. In *ROAM* [16] – a supervised trajectory anomaly detection algorithm – trajectories are represented as pattern fragments, which are transformed to a feature space of spatiotemporal attributes. A hierarchical rule-based classifier is applied for classifying the trajectories in one of the two classes (supervised setting).

iBAT [29] is a lazy isolation technique for trajectory outlier detection. A grid is used to split the studied area, and all trajectories that share the same starting/ending grid cells are grouped together to detect the anomalous trajectories among them. Thus, this method focuses on discovering anomalous trajectories in specific regions. Furthermore, trajectories are represented as sequences of discrete variables (cell id), which results in a loss of spatial information. One Class SVM clustering is applied on a dataset of trajectories, extracted from video sequences in the work of Piciarelli et al.[23]. The aim is to detect the region in the feature space that encloses the normal data and excludes the anomalous ones. To address the problem of correctly choosing the number of acceptable anomalies – unknown in unlabeled data – the authors suggest a technique for removing the outliers in the training set. They leverage geometric characteristics of the feature space of the model. After clustering the trajectories, the outliers are detected using geometric criteria in the model's feature space, without taking account of temporal dependencies.

More recent works have started applying deep learning methods for detecting outliers in trajectory data. Roy & Bilodeau [25] work on abnormal event detection by analyzing fixed length trajectories in road intersections. A deep feed forward autoencoder is trained on normal data, learning to reconstruct the input trajectories from its latent-space representation. A threshold value based on the reconstruction error in training/validation sets is used for detecting outlying trajectories. A feed forward autoencoder is also used in the work of Olive et al. [21] to discover outlying (fixed-length) trajectory patterns in flight routes. Trajectories with high reconstruction errors are then analyzed to find possible associations with urgent situations. Though feed forward neural networks have shown good potential in anomaly detection tasks, they fail at capturing temporal dependencies.

Instead, recurrent neural networks learn from sequential data and can be effectively applied in outlier detection tasks with a temporal dimension. They also have the advantage of capturing the context of the entire trajectory. This makes them more effective than methods based on trajectory segmentation [18]. Di Mauro & Ferilli [9] deploy a Seq2Seq architecture based on an LSTM autoencoder in an unsupervised setting in order to detect anomalous vehicle routes. The routes are represented as sequences of gate ids, which ignores the spatial information, as in [29]. A Sequence to Sequence real-time outlier detector in human trajectories is proposed by Bouritsas et al. [4] aiming at discovering suspicious behavior in video surveillance footage. Focusing on localization of anomaly detection, each trajectory is partitioned in sub-trajectories, represented as sequences of coordinates. Synthetic data are used for evaluation and the training is applied only on normal instances.

In this work, instead of applying one of the aforementioned methods for trajectory anomaly detection, we propose a hybrid approach, a combination of deep learning and density-based anomaly detection techniques, while taking into consideration

the length of each trajectory during the training phase. The performance of the proposed architecture is tested on synthetic data, generated to account for different patterns of anomalies.

3 METHODS

3.1 Models

Autoencoders are used as the basis for unsupervised outlier detection method. Given an input sequence of vectors of spatial coordinates (longitude, latitude) $T_i = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, the autoencoder attempts to reproduce this input, by minimizing a loss function. This loss function refers to the error of the reconstructed outputs compared to the original input sequences. The instances that the model failed to reconstruct successfully suffer from high reconstruction error and are detected as anomalies. In our work, we employ a Long Short-Term Memory (LSTM) network [13] as the basis for our models due to its ability to capture temporal dependencies in the data.

The autoencoder consists of two components – the encoder and the decoder. During the encoding phase, the sequence is compressed into a latent vector of lower dimension, which carries all its spatiotemporal information in a compressed format (bottleneck architecture). The decoder then reconstructs the input using its latent representation. Both the encoder and the decoder may consist of one or more layers. Dense layers are added on top of the last decoder layer and produce the final reconstruction for the m timesteps.

In our work, we use two variants of denoising LSTM autoencoders [28], which attempt to output the clean input data from a partially corrupted input. As a result, the model learns representations which are robust to noise. This is accomplished by adding to the input data a Gaussian noise function with zero mean and unit variance.

LSTM: In our first variant, the last layer of the encoder returns the encoder's output, which is copied m times (equal to the number of the sequence timesteps) before being fed to the decoder. Similar model architectures have been used in anomaly detection tasks in other areas of research, such as natural language processing [27]. The decoder's architecture is the same as the encoder's and dense layers added on the top produce the final output.

SEQ: Our second variant of autoencoders is a Sequence to Sequence network (Seq2Seq) [6, 26], which has shown good results in the area of machine translation. LSTM layers comprise both the encoder and the decoder. The encoder sequentially reads each element of the sequence and encodes it in a hidden state, updated at every timestep. Instead of copying the encoder's output, as with the LSTM autoencoder, this hidden state is used as the initial state of the decoder, which is trained to generate the output sequence step by step. We train the model using teacher forcing, so that the actual value at each timestep is used as input to the decoder. The final reconstructed sequence is output by dense layers. In our work, both architectures (LSTM, SEQ) consist of one encoder, one decoder and one dense (output) layer.

Traditionally the loss function minimized during training is the mean squared error (MSE) between predicted and actual (location) sequences. Here, we propose a weighted MSE loss function. The weight variable is introduced in order to penalize the long trajectories more, as they tend to have higher reconstruction error, regardless of their abnormality. The equation of the weighted loss function is:

$$L_{HVR} = \frac{1}{2|I|} \sum_i \sum_t [(x_i(t) - x_i^r(t))^2 + (y_i(t) - y_i^r(t))^2] * w_i \quad (1)$$

where i is a specific instance from a total number of instances $|I|$, t denotes a certain timestep within a trajectory route, $\{x_i(t), y_i(t)\}$ is the actual and $\{x_i^r(t), y_i^r(t)\}$ is the reconstructed location (longitude, latitude) of the trajectory i at timestep t. The weight factor w_i is given by the log Haversine distance covered in the (actual) trajectory. For two given points (α, β) it is given by:

$$w_{\alpha,\beta} = 2 \arcsin \sqrt{\sin^2 \frac{(y_\alpha - y_\beta)}{2} + \cos(y_\alpha) \cos(y_\beta) \sin^2 \frac{(x_\alpha - x_\beta)}{2}} \quad (2)$$

where x_k and y_k refer to the longitude and latitude of the point k in radians, respectively. In our experiments, we employ both models (LSTM, SEQ) with both loss functions (MSE, HVR).

3.2 Anomaly detection methods

After training the models, typically an anomaly threshold is set in order to classify a trajectory as normal or not [2, 4, 20, 25]. We employ two different methods to detect outliers in unseen trajectories, both operating on the sequences of their reconstruction errors (i.e. those derived by LSTM/SEQ models, as presented in the previous subsection). The evaluation is conducted using a test dataset and synthetic anomalous data.

AVG: The average values of errors are calculated on the sequences in the test set and are used as a proxy for measuring the abnormality in a sequence of locations visited in a trajectory. The test instances are then sorted based on these values, with the most anomalous trajectories being placed at the highest ranks.

LOF: Instead of working with averages, we propose a hybrid architecture, combining each autoencoder with the Local Outlier Factor algorithm [5] (LOF). LOF is a density-based outlier detection algorithm, which performs well even in case of different outlying densities and detects both local and global anomalies. Anomalies are detected by comparing their local densities with the average local densities of their k nearest neighbors. The algorithm contains three steps:

(1) The k nearest neighbors of each record are found, using a distance metric (in our case, Euclidean distance on the reconstruction errors of LSTM or SEQ models, trained using MSE or HVR).
(2) The Local Reachability Density is computed for each record p, using its k nearest neighbors NN_k, according to the equation:

$$LRD_k = 1/\frac{\sum_{o \epsilon NN_k(p)} rd_k(p, o)}{|NN_k(p))|} \quad (3)$$

where $rd_k(p, o)$ is the reachability distance of the object p with respect to object o.
(3) The Local Outlier Factor score of a record is computed as the ratio of the average local densities of its neighbors to its local density. The equation for LOF score is:

$$LOF(p) = \frac{\sum_{o \epsilon NN_k(p)} \frac{LRD_k(o)}{LRD_k(p)}}{|NN_k(p)|} \quad (4)$$

Finally, LOF scores close to 1 are considered to be associated with normal records, as their densities are similar to those of their neighbors, while outliers have higher score values. The

LOF algorithm is applied on the sequences of the reconstruction errors, predicted by the autoencoder. The test instances are then sorted based on their LOF scores and outliers are detected as in the AVG method.

4 EXPERIMENTS

4.1 Data

Dataset: We make use of the dataset released in the Prediction Challenge of ECML PKDD 2015[1] [19], which contains 1,710,671 taxi trips made by all 442 taxis in the city of Porto, Portugal during the time period from 01/07/2013 to 30/06/2014. This dataset has been used in the past for related tasks, such as destination prediction [8, 24] and travel time prediction [12].

The location of each trip is sampled every 15 seconds; we sub-sample this signal so that we have one timestep per minute, in order to test the performance of our models in low sampling rates. Furthermore, we also observe that there are some duplicates, which are removed from the dataset. In order to keep out trajectories with very short length, we exclude trajectories for which the sum of distances between the first, the middle and the last point is less than 500 meters. We used only three points of each trajectory instead of all of its points for computational reasons. Trips with very low duration (lower than three minutes) or very high duration (higher than two hours) are also excluded. Since the autoencoders require the trajectories to have specific number of points, we choose this number to be nine, so that 75% of the total number of trajectories has more points. As a result, we keep trajectories with more than nine points and keep the first nine of them. We note that keeping the nine first points of each trajectory may result in excluding some anomalies, which require more of them to be expressed. However, we choose this percentage (75% of all trajectories), so that we take into account a large proportion of the total number of trajectories, while the number of points is large enough for the task of detecting anomalous trajectories to be meaningful. We end up with 1,218,657 sequences of nine spatial coordinates (longitude, latitude). The features of the dataset (longitude, latitude) are standardized to zero mean and unit variance, according to the equation:

$$\tilde{x} = \frac{x - \mu(x)}{\sigma(x)} \tag{5}$$

where $\mu(x)$ and $\sigma(x)$ are the average and variance of all values of each feature, respectively.

We use 80% of the trajectories of each of the 12 months of the time period for training. From the remaining 20%, 10% is used in order to define the end of the training process to avoid overfitting (validation set) and 10% for evaluating our models (test set).

Synthetic Examples of Anomalies: We are interested in capturing different types of anomalies in the trips made by the taxis in our dataset. Due to the lack of labels in our test set, we generate synthetic data by altering a small portion (1%) of the trajectories in the test set. We therefore aim at capturing clear cases of anomalous trips, as well as anomalies that correspond to certain driving patterns. Thus, we generate artificial anomalies corresponding to (a) noisy (distorted) trips (**DSTRT**) and (b) cyclic routes (**CYCLE**). Table 1 summarises the two different approaches, their variants and the exact pattern we have used to generate such synthetic trajectories. In the experiments that are discussed next, we employ each of these patterns independently in our test set.

[1] http://www.geolink.pt/ecmlpkdd2015-challenge/

Table 1: Overview of our methods for generating synthetic anomalies in the test set. The 'Pattern' column refers to the pattern we use to alter the locations of the actual trip [0, ..., 8].

Method	Variant	Description	Pattern
DSTRT	DSTRT_c	Complete noise.	[0,8,1,7,2,6,3,5,4]
	DSTRT_p	Light noise.	[0,1,2,4,3,5,6,7,8]
CYCLE	CYCLE_c	Perform the same cycle twice and then continue with the rest of the trip.	[0,1,2,3,0,1,2,3,4]
	CYCLE_b	Go back-and-forth all the time.	[3,4,5,4,3,4,5,4,3]

4.2 Model Comparison

Task Definition: Our goal is to detect anomalous driving patterns by analysing the trajectories. To this end, we first compare the ability of our four variants to reconstruct the trajectories in the test set against a non-sequential model ('FF', see next paragraph). We then conduct our main evaluation task.
Anomaly Detection Task: We form this task in a rank-based manner. We rank the trajectories based on their anomaly factor, as derived by the models, aiming at ranking higher the (artificially generated) anomalous trajectories compared to the 'normal' ones. We compare the performance of our sequential models against baseline methods, as detailed next.

Finally, we compare two of our model variants in a qualitative way, to get insights on the advantages of using the Haversine-weighted loss function compared to standard MSE, as well as the hybrid (+LOF) approach operating on the reconstruction errors compared to considering the average error.

Models: We contrast the two models defined in section 3.1, each trained by minimising (a) the Mean Squared Error (MSE) and (b) the Haversine-weighted (HVR) loss function. We denote the models as $\{\textbf{LSTM}_m, \textbf{SEQ}_m\}$ when they use the MSE loss function and as $\{\textbf{LSTM}_h, \textbf{SEQ}_h\}$ when they use the HVR loss function respectively. We further compare their performance by ranking the trajectories based on the AVG or LOF method, as detailed in section 3.2. All versions of our models have two hidden layers of 16 units, followed by one dense layer for the final prediction, since using these parameters offered the best performance in terms of reconstruction error for all cases.

To test the importance of the temporal component in our models for the *Anomaly Detection* task, we compare them against a denoising autoencoder implemented via a Feed-Forward Neural Network with two hidden layers (**FF**) and a Local Outlier Factor (**LOF**) algorithm trained on the raw trajectory data instead of the sequences of reconstruction errors derived by one of our models. Both of these baseline models treat the different timesteps as different features, ignoring their temporal dimension. FF is trained in the same data and using the same noise as our models. LOF is trained straight away on the test set to detect the anomalies and we select the number of neighbors (trials: [10,50,100,500,1000]) on the basis of its performance on the test set, as to provide it with a strong advantage against our models. Our sequential models as well as the FF baseline are trained using the Adam optimizer [14], an L2 regularization norm equal to 0.1, a learning rate equal to 0.0001 and a batch size equal to 512. Finally, we provide the evaluation metrics (see next subsection) of a naïve random ranking model (**NRR**), by averaging the performance after 10K experiments with random scores on the test set. For the

case of the *Trajectory Reconstruction* subtask, we also compare our models against FF.

We implement neural network architectures based on the Keras library [7] and the LOF algorithm using the Scikit-learn library [22], both in Python.

Anomaly Scores. In the *Anomaly Detection* task, once our models (LSTM_m, LSTM_h, SEQ_m, SEQ_h) and the FF baseline make predictions on the test set[2], we measure the MSE between the model predictions and the actual location of each timestep of every trip in the test set. We then rank the trips of the test set by employing any of the methods described in section 3.2. For LOF, we rank trips in the test set based on their LOF scores in order to identify (i.e. rank higher) the synthetic ones.

Evaluation. We make use of three evaluation metrics that are suited for ranking tasks and widely used in related work. In specific, we use two common classification-based metrics, adjusted for the needs of ranking tasks: **Recall at k** ($r@k$) measures the portion of the anomalous trips that are present in the top-k trips ranked by our model; **Precision at k** ($p@k$) measures the portion of the top-k trips ranked by our model that are indeed anomalous. We finally calculate the **F1 Measure** at k as our main evaluation metric, defined as follows:

$$F1 = \frac{2 \cdot r@k \cdot p@k}{r@k + p@k} \tag{6}$$

We set k to be equal to 5% of the size of our test set. This value is larger than the percentage of the artificial anomalies we aim at detecting (1%), since we expect that there will be more (non artificial) anomalies in our test set. Nevertheless, we further experiment with different values of k (range: [0.1%-50%]).

To help with the reproducibility of our experiments, we make available the implementation of the proposed methods and the conducted experiments on GitHub[3].

5 RESULTS

In this section we present the results from our models' comparison. The reconstruction errors of our sequential models – a metric of their ability to accurately reconstruct the trajectories – is compared to the FF model error in our test set. Then, we present the comparison between the performance of our hybrid LOF models against the AVG method (described in section 3.2). The impact of using a Haversine-weighted loss function is also examined next. Finally, we present our empirical insights by comparing the anomalies detected in our test set by SEQ_m with AVG against SEQ_h with LOF, to test the advantages of the latter approach.

5.1 Base Task – Trajectory Reconstruction

The mean absolute and mean squared error in the test set are computed as metrics of the reconstruction error. The results are shown in Table 2. The two sequential models show better reconstruction ability than the FF model, demonstrating the advantage of models that consider the temporal dimension of the trajectories. We also find that the incorporation of the Haversine-weighted (HVR) loss function further improves the results consistently across the two models (SEQ, LSTM).

[2]Note that the 1% of the trips in the test set have been altered according to one of the four patterns described in section 4.1.

[3]https://github.com/marialiatsikou/BMDA_anomaly_detection

Table 2: Mean Absolute and Mean Squared Error of our four models and the FF baseline. Performance is measured via the reconstruction errors in the test set.

	MAE	MSE
FF	0.00200	$8.726 \cdot 10^{-6}$
LSTM_m	**0.00130**	$4.112 \cdot 10^{-6}$
SEQ_m	0.00156	$4.968 \cdot 10^{-6}$
LSTM_h	**0.00130**	$3.908 \cdot 10^{-6}$
SEQ_h	0.00144	$4.286 \cdot 10^{-6}$

5.2 Main Task – Anomaly Detection

Table 3 shows the results of our models and baselines on the anomaly detection task across all experiments (i.e. across all synthetic datasets, as summarised in Table 1). In the following, we examine the effects of the different components of our models for the anomaly detection task.

Anomaly Detection Methods: Our sequential models outperform the FF and LOF baseline models in almost all cases. The application of LOF on the sequences of reconstruction errors in all cases of synthetic datasets improves the performance of all four sequential autoencoders compared to the AVG method, in which the ranking of trajectories is determined by the average error values. The relative improvements in F1 measure (averaged over the corresponding values in the four datasets) of LSTM_m and LSTM_h models are 18.3% and 15.1% respectively. The corresponding numbers for Seq_m and Seq_h are much higher (50.2% and 33.2% respectively). The hybrid approach gives much higher F1 score values even in the predictions made by the FF model (i.e., comparing FF+AVG vs FF+LOF). It is worth noting that FF with LOF outperforms the other models in the case of CYCLE_b, while its performance in this case is much lower compared to all other models (except for NRR) in the case of AVG method. These findings show the clear and consistent benefit we gain via the hybrid architecture proposed in this work.

Loss Functions: We examine the performance of the models with respect to the loss functions used. The Haversine-weighted loss function (HVR) outperforms Mean Squared Error loss (MSE) with respect to F1 measure in 13 out of total 16 cases shown in Table 3 (the cases refer to the predictions of the two Sequential models for the four different datasets). The relative gains in performance by using the Haversine-weighted loss are 1.32% and 4.14% for the LSTM and SEQ models, respectively.

To further examine the performance of the four variants (the SEQ/LSTM models (a) trained using the MSE/HVR loss function and (b) using the AVG/LOF method for their final ranking), Figures 1, 2 show the variation of F1 measure over different threshold values of k for LSTM and SEQ model respectively. It becomes clear that our hybrid approach combined with the Haversine weighted loss function (HVR+LOF) gives better results for LSTM and SEQ in most cases, providing a strong boost in performance compared to the common MSE+AVG approach.

5.3 Qualitative Analysis

As a final step, we employ SEQ_m with AVG method and SEQ_h with LOF method to rank all the trajectories of our original test set. As opposed to the quantitative analysis conducted thus far on the artificially generated anomalies, here we are interested in gaining qualitative insights on the detected anomalies by each

Table 3: Evaluation (in %) of all models in all runs for the *Anomaly Detection* task for k=5%. Blue cells indicate the best-performing anomaly detection method (AVG vs LOF) for each model, per synthetic dataset. Green cells indicate the best performing model per synthetic dataset. Bold scores show the best-performing loss (MSE vs HVR) when comparing the two variants of each of our sequential models.

		DSTRT				CYCLE			
		$DSTRT_c$		$DSTRT_p$		$CYCLE_c$		$CYCLE_b$	
		AVG	LOF	AVG	LOF	AVG	LOF	AVG	LOF
Precision	NRR	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	LOF	–	17.23	–	2.69	–	4.43	–	4.43
	FF	10.98	16.43	1.74	3.53	0.94	3.58	1.72	9.29
	$LSTM_m$	16.72	18.22	3.02	5.15	3.68	4.41	4.91	5.73
	SEQ_m	15.08	18.12	2.48	4.94	2.48	5.22	3.15	6.53
	$LSTM_h$	16.90	18.38	3.10	4.73	4.00	4.56	5.12	5.86
	SEQ_h	16.02	18.78	2.79	5.05	3.33	5.10	3.76	5.55
Recall	NRR	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
	LOF	–	86.21	–	13.46	–	22.17	–	22.17
	FF	54.93	82.18	8.70	17.65	4.68	17.90	8.62	46.47
	$LSTM_m$	83.66	91.13	15.11	25.78	18.39	22.09	24.55	28.65
	SEQ_m	75.45	90.64	12.40	24.71	12.40	26.11	15.76	32.68
	$LSTM_h$	84.56	91.95	15.52	23.65	20.03	22.82	25.62	29.31
	SEQ_h	80.13	93.92	13.96	25.29	16.67	25.53	18.80	27.75
F1 Measure	NRR	1.67	1.67	1.67	1.67	1.67	1.67	1.67	1.67
	LOF	–	28.72	–	4.49	–	7.39	–	7.39
	FF	18.30	27.38	2.90	5.88	1.56	5.96	2.87	15.48
	$LSTM_m$	27.88	30.37	5.03	**8.59**	6.13	7.36	8.18	9.55
	SEQ_m	25.14	30.20	4.13	8.23	4.13	**8.70**	5.25	**10.89**
	$LSTM_h$	**28.18**	**30.64**	**5.17**	7.88	**6.67**	**7.60**	**8.54**	**9.77**
	SEQ_h	**26.70**	**31.30**	**4.65**	**8.43**	**5.55**	8.51	**6.26**	9.25

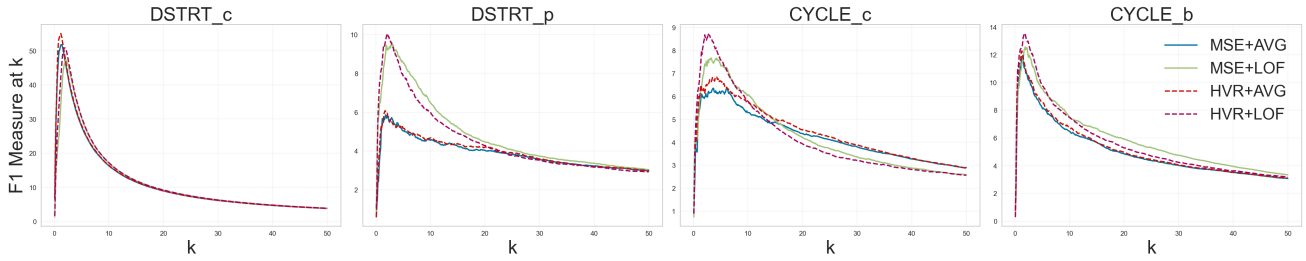

Figure 1: F1 Measure for different values of k (x-axis) for LSTM model.

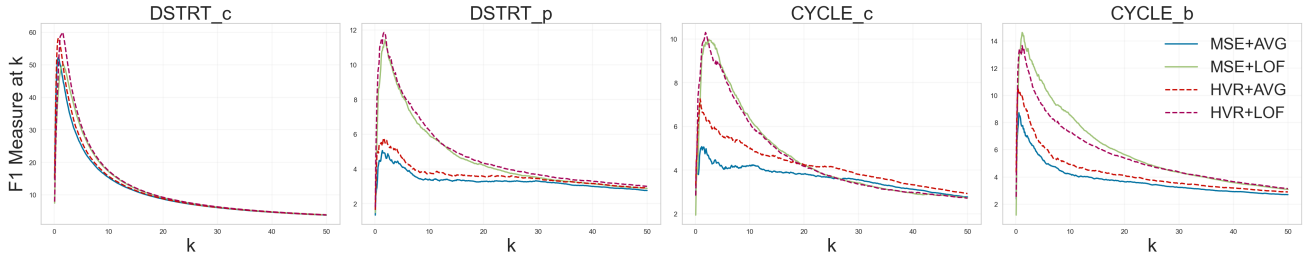

Figure 2: F1 Measure for different values of k (x-axis) for SEQ model.

of these methods on our actual test set. We obtained the two sets of the top-0.1% (i.e., most anomalous) of the trajectories detected by each of the two models in the test set and focused on those that are predicted by one model only (i.e., we excluded the intersection of the two sets).

Two annotators labelled these trajectories as 'normal' or 'abnormal' (or 'NA', if they could not tell). To familiarise themselves with the task, the annotators (one author, one PhD graduate with no domain knowledge) were first provided with examples of normal and abnormal trajectories in the test set and were then

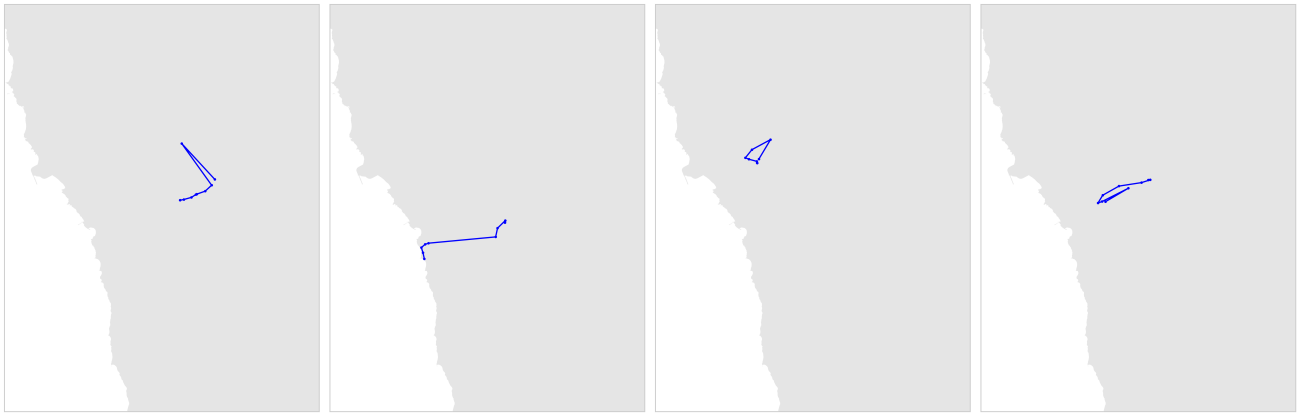

Figure 3: The types of anomalies detected by the SEQ_h with LOF method. The patterns from left to right are: GPS device error, unreasonably long covered distance, cyclic route, back and forth movement.

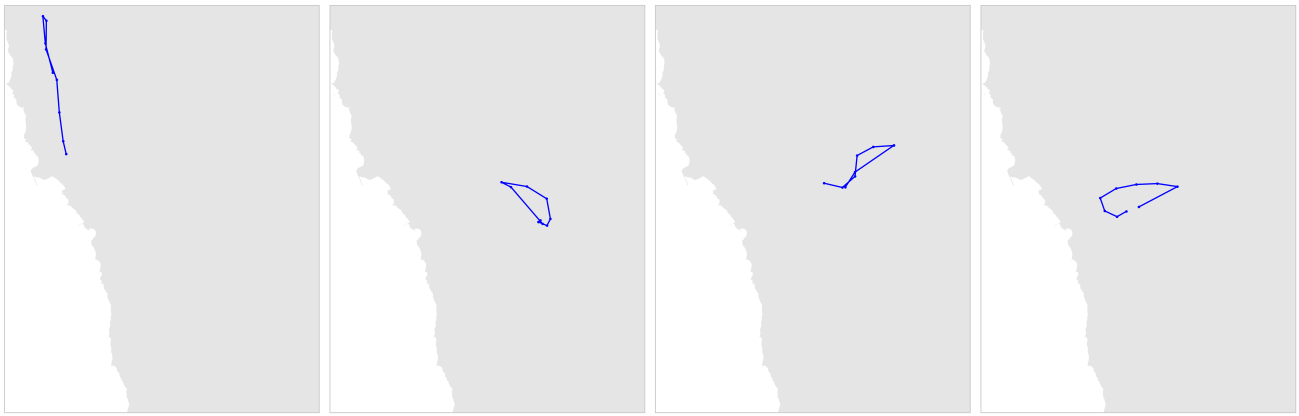

Figure 4: The types of anomalies detected by the SEQ_m with AVG method. The patterns from left to right are: One back and forth movement and three routes with cyclic parts.

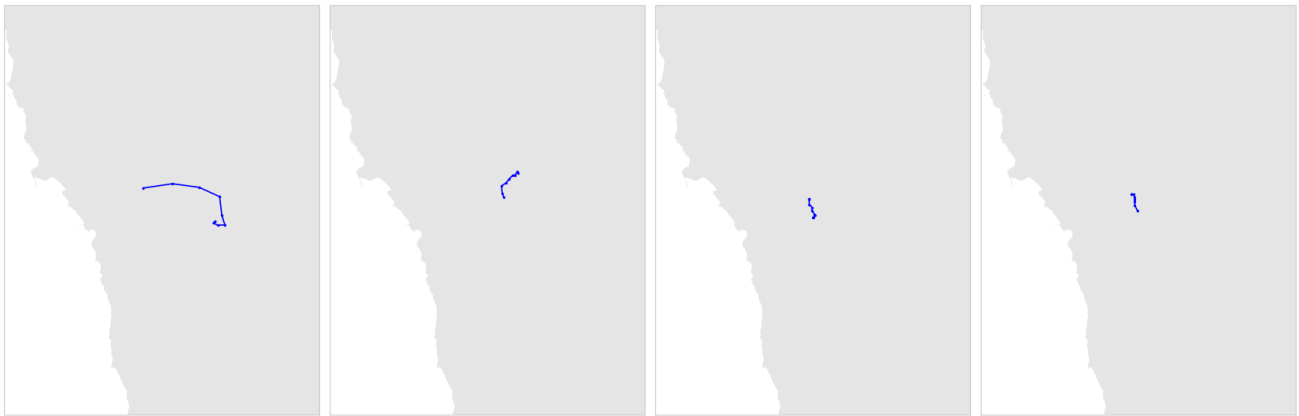

Figure 5: Four typical trajectories classified as normal by the SEQ_h with LOF method.

asked to label the trajectories in our dataset of interest in a similar manner. There was an agreement on 73/86 cases, with the inter-annotator agreement in Cohen's kappa terms being 0.72 (i.e. 'substantial agreement'). For SEQ_m with the AVG method, 4 trajectories out of the 42 (accuracy=9.5%) for which the annotations coincide (excluding 'NA'), were labelled as 'abnormal'; for SEQ_h with the LOF approach the anomalous trajectories were 15 out of 29 (accuracy=51.7%), certifying the superiority of our hybrid model.

In Figure 3 four patterns of anomalies detected by the SEQ_h with LOF method are displayed. The first corresponds to error of the GPS device, while the rest possibly refer to driving patterns: unreasonably long covered distance, a cyclic trip and a trip including back and forth movement. For comparison purposes we

present the four trajectories annotated as anomalies by both annotators for SEQ_m with the AVG method in Figure 4. It is obvious that the types of anomalies detected with this approach are more restricted and they refer to a trip which includes back and forth movement and three trips with cyclic parts. Finally, some typical trajectories that our hybrid model (SEQ_h with LOF method) classified as normal are displayed in Figure 5. The findings of this qualitative analysis show that the proposed approach can capture different types of anomalies, despite not being tailored to any of them explicitly. Annotating a larger dataset of trajectories based on their pattern of anomaly could lead to further insights on the performance of our models in future work.

6 CONCLUSION

In this work we introduce a hybrid architecture, which combines a sequential denoising autoencoder with a density-based outlier detection algorithm, for trajectory anomaly detection. We propose two variants of sequential models, a denoising LSTM autoencoder and a denoising Seq2Seq autoencoder. They are both trained by optimizing a Haversine distance-based weighted loss function, in order to take into account the overall distance covered in a trajectory. During the detection phase, the Local Outlier Factor algorithm is applied on the sequences of reconstruction errors of unseen trajectory data. The evaluation is conducted on four different types of synthetic data, generated by altering a small portion of the trajectories of our test set. We show that the hybrid architectures outperform other anomaly detection methods. Regarding the proposed loss function, results for both detection methods (AVG and LOF) are in most cases improved by the introduction of the weighted loss function.

In the future, we intend to apply our methodology in datasets of different trajectory length and type, including bike sharing data from the i-CHANGE platform [1], and to collaborate with domain experts to obtain annotations for anomalous trajectories. We also intend to test other variants of loss functions during training and also test transfer learning approaches across different datasets and domains as well as to try out different deep learning architectures.

ACKNOWLEDGMENTS

This research was co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code:T1EDK-04582).

REFERENCES

[1] Lazaros Apostolidis, Symeon Papadopoulos, Maria Liatsikou, Ioannis Fyrogenis, Efthymis Papadopoulos, George Keikoglou, Konstantinos Alexiou, Nasos Chondros, Ioannis Kompatsiaris, and Ioannis Politis. 2020. i-CHANGE: A Platform for Managing Dockless Bike Sharing Systems. In *International Conference on Computational Science and Its Applications*. Springer, 851–867.

[2] Javed Ashraf, Asim D Bakhshi, Nour Moustafa, Hasnat Khurshid, Abdullah Javed, and Amin Beheshti. 2020. Novel Deep Learning-Enabled LSTM Autoencoder Architecture for Discovering Anomalous Events From Intelligent Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems* (2020).

[3] Kiran Bhowmick and Meera Narvekar. 2018. Trajectory Outlier Detection for Traffic Events: A Survey. In *Intelligent Computing and Information and Communication*. Springer Singapore, Singapore, 37–46.

[4] Giorgos Bouritsas, Stelios Daveas, Antonios Danelakis, and Stelios CA Thomopoulos. 2019. Automated Real-time Anomaly Detection in Human Trajectories using Sequence to Sequence Networks. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 1–8.

[5] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 93–104.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[7] François Chollet et al. 2015. Keras. https://keras.io.

[8] Alexandre De Brébisson, Étienne Simon, Alex Auvolat, Pascal Vincent, and Yoshua Bengio. 2015. Artificial neural networks applied to taxi destination prediction. In *Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge-Volume 1526*. 40–51.

[9] Nicola Di Mauro and Stefano Ferilli. 2018. Unsupervised LSTMs-based Learning for Anomaly Detection in Highway Traffic Data. In *International Symposium on Methodologies for Intelligent Systems*. Springer, 281–290.

[10] Vitor Cunha Fontes, Lucas Andre de Alencar, Chiara Renso, Vania Bogorny, and It Pisa. 2013. Discovering Trajectory Outliers between Regions of Interest. In *GeoInfo*. Citeseer, 49–60.

[11] Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. 2013. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering* 26, 9 (2013), 2250–2267.

[12] Thomas Hoch. 2015. An ensemble learning approach for the Kaggle taxi travel time prediction challenge. In *Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge-Volume 1526*. 52–62.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory outlier detection: A partition-and-detect framework. In *2008 IEEE 24th International Conference on Data Engineering*. IEEE, 140–149.

[16] Xiaolei Li, Jiawei Han, Sangkyum Kim, and Hector Gonzalez. 2007. Roam: Rule-and motif-based anomaly detection in massive moving object data sets. In *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM, 273–284.

[17] Xiaolei Li, Zhenhui Li, Jiawei Han, and Jae-Gil Lee. 2009. Temporal outlier detection in vehicle traffic data. In *2009 IEEE 25th International Conference on Data Engineering*. IEEE, 1319–1322.

[18] Weining Lu, Yu Cheng, Cao Xiao, Shiyu Chang, Shuai Huang, Bin Liang, and Thomas Huang. 2017. Unsupervised sequential outlier detection with deep architectures. *IEEE transactions on image processing* 26, 9 (2017), 4321–4330.

[19] Joao Mendes-Moreira and Luis Moreira-Matias. 2015. On learning from taxi-GPS traces. In *Proceedings of the 2015th International Conference on ECML PKDD Discovery Challenge-Volume 1526*. CEUR-WS. org, 37–39.

[20] Ali H Mirza and Selin Cosan. 2018. Computer network intrusion detection using sequential LSTM neural networks autoencoders. In *2018 26th signal processing and communications applications conference (SIU)*. IEEE, 1–4.

[21] Xavier Olive, Jeremy Grignard, Thomas Dubot, and Julie Saint-Lot. 2018. Detecting Controllers' Actions in Past Mode S Data by Autoencoder-Based Anomaly Detection.

[22] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

[23] Claudio Piciarelli, Christian Micheloni, and Gian Luca Foresti. 2008. Trajectory-based anomalous event detection. *IEEE Transactions on Circuits and Systems for video Technology* 18, 11 (2008), 1544–1554.

[24] Alberto Rossi, Gianni Barlacchi, Monica Bianchini, and Bruno Lepri. 2019. Modelling Taxi Drivers' Behaviour for the Next Destination Prediction. *IEEE Transactions on Intelligent Transportation Systems* (2019).

[25] Pankaj Raj Roy and Guillaume-Alexandre Bilodeau. 2018. Road user abnormal trajectory detection using a deep autoencoder. In *International Symposium on Visual Computing*. Springer, 748–757.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[27] Adam Tsakalidis and Maria Liakata. 2020. Sequential Modelling of the Evolution of Word Representations for Semantic Change Detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 8485–8497.

[28] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research* 11, 12 (2010).

[29] Daqing Zhang, Nan Li, Zhi-Hua Zhou, Chao Chen, Lin Sun, and Shijian Li. 2011. iBAT: detecting anomalous taxi trajectories from GPS traces. In *Proceedings of the 13th international conference on Ubiquitous computing*. 99–108.