Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on GPS traces while driving

Mario Munoz-Organero, M., Ramona Ruiz-Blaquez, Luis Sánchez-Fernández

Telematics Engineering Department and UC3M-BS Institute of Financial Big Data, Carlos III University of Madrid, Av. Universidad, 30, 28911 Leganes, Madrid, Spain

Keywords:
Traffic element detection
Outlier based classification
Machine learning techniques
GPS pattern analysis

A b s t r a c t

The automatic generation of street networks is attracting the attention of research and industry communities in areas such as routable map generation. This paper presents a novel mechanism that focuses on the automatic detection of street elements such as traffic lights, street crossings and roundabouts which could be used to generate street maps and populate them with trafficinfluencing infrastructural elements such as traffic lights. In order to minimize the system requirements and simplify the data collection from many users with minimal impact for them, only traces of GPS data from a mobile device while driving are used. Speed and acceleration time series are derived from the GPS data. An outlier detection algorithm is used first in order to detect abnormal driving locations (which can be due to infrastructural elements or particular traffic conditions). Using deep learning, speed and acceleration patterns are automatically analyzed at each outlier in order to extract relevant features which are then classified into a traffic light, street crossing, urban roundabout or other element. The classification results are enhanced by adding the degree of atypicity for each point calculated as the percentage of times that a particular location is detected as an outlier in several drives. The proposed algorithm achieves a combined recall of 0.89 and a combined precision of 0.88 for classification.

1. Introduction

Mobile phone integrated sensors offer advanced services that have the potential of enhancing the overall user knowledge, perception and experience applied to areas such as transportation or agriculture (Kamilaris & Pitsillides, 2016). Using data obtained from embedded sensors on mobile devices when worn by a human user, different human activities can be detected (Munoz-Organero & Lotfi, 2016). Using similar data when the mobile device is transported while driving, relevant infrastructural and traffic related information can be extracted. Infrastructural information such as the location of traffic lights and their red/green periods, the location of roundabouts or street crossings and the common speed patterns while traversing those elements can be used to provide better estimations of the travel time to reach a parti-cular destination using a routable map. The travel time estimates could be personalized to a particular user, the time of the day and the day of the week. The main objective of this manuscript is to present and va-lidate a new algorithm to extract road infrastructural elements based on GPS traces while driving. The algorithm combines outlier detection and deep learning classification techniques based on speed and acceleration patterns derived from the GPS sensor on a mobile device.

Using in-vehicle sensors to automatically detect road and traffic conditions while driving has motivated several previous research studies. Hemminki, Nurmi, and Tarkoma (2013) used the acceleration data from a mobile device for accurate and fine-grained detection of different transport modes by using a set of accelerometer features that are able to capture key characteristics of vehicular movement patterns, and a hierarchical decomposition of the detection task. Lan, Xu, Khalifa, Hassan, and Hu (2016) used a different sensor for the same objective. They used the output voltage from the kinetic energy harvesting device as the signal source to achieve transportation mode detection. The accelerometer data from a mobile device has also been used to detect infrastructural elements on the road, either when carried by pedestrians or vehicles. Bujari, Licar, and Palazzi (2012) used acceleration based movement pattern recognition in day-by-day urban street behavior to detect when a pedestrian stops and then crosses a street ruled by a traffic light. Perttula, Parviainen, and Collin (2016) used on-vehicle inertial measurement units to detect pedestrians, driving behavior, as well as road conditions like bumps. Smartphones have also been used to automatically detect traffic accidents using sensors on mobile devices.

White, Thompson, Turner, Dougherty, and Schmidt (2011) used accelerometers and acoustic data to immediately notify a central emergency dispatch server after an accident, and provide situational awareness through photographs, GPS coordinates, VOIP communication channels, and accident data recording.

Using the camera sensor on a mobile device, several road-infrastructural information has been obtained in previous research studies. Mathibela, Newman, and Posner (2015) addressed the problem of automatically reading the rules encoded in road markings and inferring the semantics of road scenes. Mascetti et al. (2016) and Wu, Watanabe, and Ishikawa (2016) detected traffic lights based on image processing on mobile devices.

The information about the vehicle's speed (normally derived from another sensor such as GPS or accelerometer) has also been used to detect driving related events or infrastructural elements. Panichpapiboon and Leakkaw (2016) used the driving speed calculated from accelerometer data in order to detect traffic levels.

The information gathered from different sensors together can be fused to improve detection rates and accuracy levels. Aly, Basalamah, and Youssef (2017) automatically detected road related information such as tunnels, bumps, bridges, footbridges and crosswalks based on the combined use of various sensors on mobile devices including inertial sensors (such as accelerometer, gyroscope and magnetometer) as well as cellular network information. They also combined information from pedestrians and vehicles and used crowdsensing in order to improve the accuracy of results. Using the information of several users provides an additional source of data to validate that a detected element is not an outlier (false positive). Dunlop, Roper, Elliot, McCartan, and McGregor (2016) also used crowdsensing techniques to detect dangerous road sections.

The GPS sensor on mobile devices has also been used for automatically extracting road related information. Wang, Wang, Song, and Raghavan (2017) used GPS information to automatically extract road network properties such as intersections and traffic rules to facilitate the production of high-quality routable maps. D'Andrea and Marcelloni (2016) used GPS information while driving for real-time detection of road traffic congestions and incidents.

The information obtained from embedded sensors on mobile devices has to be automatically analyzed in order to detect common patterns associated with the particular elements to be detected. Several machine learning techniques and approaches have already been used in order to detect road related information from data obtained from mobile sensors. Ren and Liu (2016) used K-means to detect potholes based on acceleration data while driving. Ferri (2016) used decision trees, logistic regression, Naïve Bayes, KNN (K-nearest neighbours), SVM (support vector machines) and MDA (Mixture Discriminant Analysis) to extract information from GPS traces. Corcoba Magaña and Muñoz-Organero (2016) used several classification techniques applied to on-vehicle telemetry data to detect traffic incidents. Magaña, Organero, Fisteus, and Fernández (2016) made used of deep learning methods to detect the driver state while driving based on GPS data from a mobile device and a heart rate wearable sensor.

Outlier detection techniques have also been used in previous studies applied to sensed data while driving in order to automatically detect traffic incidents. Ma, Ngan, and Liu (2016) used a density-based outlier detection method by measuring the LOF (Local Outlier Factor) on a projected PCA (Principal Component Analysis) domain from real world spatiotemporal traffic signals to detect traffic data outliers which are errors in data and traffic anomalies in real situations such as accidents and congestions. Tang and Ngan (2016) also used an outlier detection algorithm in a traffic system to alert the transport department and drivers with abnormal traffic situations such as congestion and traffic accidents.

The research presented in this paper combines outlier detection with deep learning based pattern recognition and classification from GPS derived data while driving. Only GPS data is used in order to minimize the deployment requirements and therefore facilitate the use of the proposed algorithm in crowdsensing scenarios. Speed and acceleration data series are computed form the GPS data. Window-segments of a constant duration around locations classified as outliers while driving are used to train a DBN (Deep Believe Network) followed by a classifier in order to detect 3 particular road elements: traffic lights, street crossings (where two roads cross each other with signs only) and roundabouts. The algorithm is applied to 16.97 h (61,076 s.) of unlabeled driving data in order to assess the detection performance.

2. Methods

The methods used to pre-process the data, perform a pre-classification process to simplify the searching space and finally perform a road element detection while driving are presented in this section.

2.1. Data sensing and pre-processing

The GPS sensor in a mobile device is used to obtain both the vehicle's location and the estimated driving speed. The speed can be derived from the distance traveled per time unit following Eq. (1). The distance traveled between two points 1 and 2 can be calculated form the GPS coordinates as captured in Eq. (2)

$$v = \frac{\Delta s}{\Delta t} \tag{1}$$

$$\Delta s = \cos^{-1}(\sin\varphi_1 \bullet \sin\varphi_2 + \cos\varphi_1 \bullet \cos\varphi_2 \bullet \cos\Delta\delta) \bullet R \tag{2}$$

where φ_i represents the latitude in radians of location i, $\Delta\delta$ is the difference of longitudes in radians and R is the Earth radius.

The errors in the coordinates that the GPS sensor provides will propagate when using Eqs. (1) and (2) in order to estimate the instant speed. The random errors can be reduced by increasing the size of Δt. However, the resolution in time decreases when Δt increases. A value of 5 s has been used as a tradeoff in order to estimate the vehicle's speed.

The instant acceleration can be estimated from the estimated speed as shown in Eq. (3).

$$a = \frac{\Delta v}{\Delta t} \tag{3}$$

The travelling direction in bi-directional road and street segments has to be taken into account. The traffic lights affect the traffic crossing them in one particular direction. Considering the driving direction in roundabouts, on the other side, allows us to detect when the driver is entering or leaving it. Data obtained from Eqs. (1), (2) and (3) has to be associated to a particular location and a particular travelling direction.

2.2. Outlier detection

A novel outlier detection technique is proposed in this paper in order to pre-select candidate locations in which a road element could be most likely present. The idea is that road elements such as traffic lights, street crossings or roundabouts will generate outliers in speed and acceleration patterns when comparing with adjacent locations in the same drive (fast changes in speed and acceleration when approaching the particular element) but will not be considered as outliers when generating the stochastic information from the same location in different drives (deceleration patterns will coincide in similar conditions in different drives for example). This particularity will allow us to pre-filter outlier driving points due to random traffic conditions such as traffic jams from infrastructural road elements which we want to detect.

The proposed algorithm consists on the following steps:

1. A feature vector f = (v, a) is computed every second while driving.
2. A window of 20 s centered at each location while driving of feature vectors F = (f_1, f_2, ..., f_{10}, f_{12}, ..., f_{21}) is obtained per each location L_N.

3. The Mahalanobis distance is then computed for each location as $MD_N = MD(f_{11}, F)$
4. If $MD_N > Th$ then the location is pre-marked as a candidate location to hold a road element to detect (Th is a threshold value).
5. Then, a variable length array of feature vectors $F' = (f_1, \ldots, f_M)$ is created for each location L_N previously pre-filtered considering the feature vectors generated in the same location in different drives (for all the drives).
6. The Mahalanobis distance is then computed for each outlier at each location as $MD_{NO} = MD(f_o, F')$; f_o being the feature vector for each outlier in L_N in the previous step (4).
7. If $MD_{NO} > Th'$ the candidate point is discarded since it represents a sporadic outlier which is likely to be caused by a non-infrastructural road element (Th′ could be chosen to coincide with Th).

2.3. Deep Believe Network based classification

Once the candidate points have been pre-filtered as likely to contain road infrastructural elements using the outlier detection algorithm described in the previous section, a classifier could be used in order to assess which is the most likely element behind that location.

A deep-belief network (DBN) (Hinton, Osindero, & Teh, 2006) is defined as a stack of Restricted Boltzmann Machines (RBM), in which each RBM layer communicates with both the previous and subsequent layers. The nodes of any single layer don't communicate with each other laterally. The end of DBN is a classifier.

Restricted Boltzmann Machines are bipartite graphs with a layer of "hidden" neurons and a layer of "visible" neurons, without connections between neurons in the same layer. Each node represents a random variable and each edge a dependency between variables that connects.

An energy function (E) and probability distribution are used to describe a RBM. The energy of a configuration (pair of Boolean vectors) (v, h) is defined as:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j \quad (4)$$

where a_i is the bias weight (offset) for the visible unit v_i, b_j is the bias weight for the hidden unit h_j, and $w_{i\,,\,j}$ the weight associated with the connection between hidden unit h_j and visible unit v_i.

A probability Distribution is associated with the energy function in Eq. (4) as shown in Eq. (5):

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)} \quad (5)$$

where Z is a partition function defined as the sum of $e^{-E(v,\,h)}$ over all possible configurations. The aim is to ensure the probability distribution sums to 1.

Through summation we can get the marginal distribution of visible layer??:

$$p(v) = \frac{1}{Z} \sum_h e^{-E(v,h)} \quad (6)$$

A RBM is trained to maximize the product of probabilities (as described by Eq. (6) assigned to a training set V

$$\underset{W}{argmax} \prod_{v \in V} p(v) \quad (7)$$

Training a DBN to maximize Eq. (7) with samples from the same class will automatically learn the common patterns in the class. A DBN will therefore extract the best features from the samples of a particular class in order to describe those samples. After training a DBN for each class, a final classifier could be used in order to assess the accuracy of the trained DBN to differentiate among the different classes.

2.4. Pattern recognition based on similarities with auto-encoded segments

Using a classifier at the end of a DBN will allow us to separate samples in a set of classes. However, for automatically classifying non-tagged pre-filtered candidate samples while driving based on the outlier pre-detection algorithm, the null class has also to be considered. The null class will contain the pre-selected samples which do not belong to one of the categories that we want to detect (traffic light, street crossing and roundabout). The major difficulty in this approach is how to accurately describe the null class since it should include samples from any possible situation different from the target classes. In order not to have to characterize the null class, a similarity based approach will be used when detecting the target road elements while driving.

A method based on training a different auto-encoder per class will be performed in order to describe each class. The windows of 20 s of feature vectors $F = (f_1, \ldots, f_{21})$ from the input data centered at the outlier locations labeled as members of each class are used as the input for each auto-encoder. Auto-encoders are designed to minimize the error between the input and the reconstructed output according to Eq.(8).

$$\varepsilon(x,x') = \|x - f_2(W'(f_1(Wx + b)) + b')\|^2 \quad (8)$$

where x′ is the reconstructed signal which is the concatenation of the encoder and decoder functions (f_1 and f_2 are activation functions such as the sigmoid function). A final detection function is required at the end of the auto-encoder in order to assess the similarity of the input and the reconstructed output. We have used the Pearson's correlation coefficient as a similarity function.

The similarity function of each pre-selected outlier location will be computed for each class. If the maximum similarity is greater than a threshold we will assume that the outlier belongs to that class.

3. Scenarios and datasets

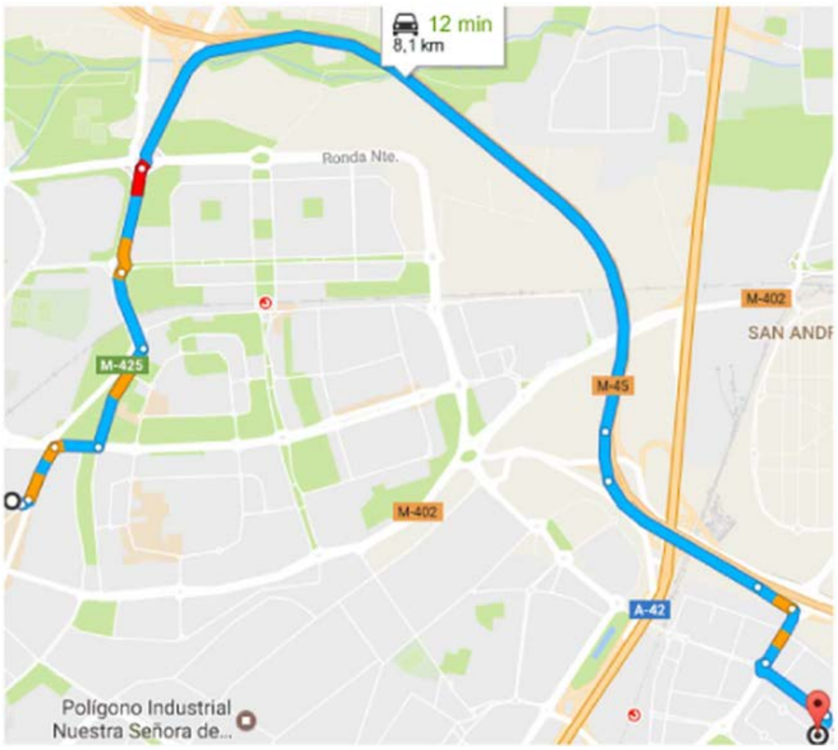
Two datasets have been used in order to validate the proposed algorithm. The first one is a specifically designed dataset that we have generated using the driving path shown in Fig. 1 which has been traveled 55 times using 3 different car models (captured in Table 1). The journey includes two urban areas and a connecting motorway. The overall length is 8.1 km and includes several road elements of interest (traffic lights, street crossings and roundabouts). The dataset has been generated from one driver driving the 3 different vehicles without being aware of the purpose of the experiment. He was asked to drive normally while an Android mobile device recorded the GPS traces. A Nexus 6 Android mobile device was used to record the GPS traces along the way. In order to validate the generalization of results, the dataset in Schneegass, Pfleging, Broy, Schmidt, and Heinrich (2013) has been used as the second dataset. A group of 10 different participants re-corded data from different sensors while driving in a circuit in Stuttgart, Germany. We have only used the data form the GPS sensor in the Schneegass et al. (2013) database, sampled at I Hz for consistency with the sampling rate in our database. The driving path also comprised urban areas and connecting motorways.

The GPS sensor in both datasets was sampled at 1 Hz (1 sample per second). The speed and acceleration were obtained as captured by Eqs. (1)–(3). Each location was automatically tagged as belonging to a particular class (traffic lights, street crossings, roundabouts and null class) by calculating the distance to the next road element (using the coordinates of the elements in the driving path and Eq. (2)) and comparing the distance with a proximity threshold (in our case we have used 50 m as the area of influence of each infrastructural element in order to include the traveled distance needed to stop the vehicle and the number of vehicles already waiting at that particular element). The speed and acceleration patterns are then used to feed the outlier de-tection algorithm proposed in Section 2.2 in order to pre-select

Table 1
– Vehicles used for the data gathering.

Model	Times used
Peugeot 206	7
Citroen Xsara Picasso	28
Opel Zafira	20

Fig. 1. – Driving path.



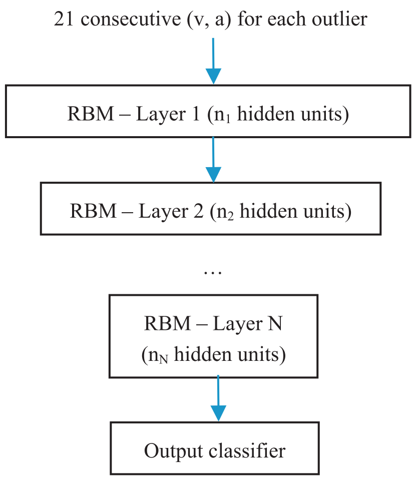

21 consecutive (v, a) for each outlier
RBM – Layer 1 (n1 hidden units)
RBM – Layer 2 (n2 hidden units)
…
RBM – Layer N (nN hidden units)
Output classifier

Fig. 2. – Classification schema.

candidate locations to contain road elements in the target classes. The thresholds Th and Th′ described in Section 2.2 control the number of samples that are pre-filtered before performing training and detection of candidate locations. The bigger the threshold the smaller the number of pre-selected points and therefore the faster the computation of re-sults. However, when value for the threshold Th is increased, the number of training samples decreases and the training of the algorithm could miss important information is that threshold is high. We have tried several values and finally selected a value of 3 for both thresholds Th and Th′. The pre-selected samples tagged as belonging to the traffic lights, street crossings and roundabouts classes are then used to train both a DBN and an autoencoder. The results are presented in the next section.

4. Results and discussion

4.1. Using a DBN to classify target elements

A DBN and a final classifier have been used to assess if the underlying differences in the acceleration and speed patterns in the outlier locations associated with each target class are enough in order to separate each sample into its correct class. The schema used is captured in Fig. 2. The number of layers and the number of hidden units per layer are two parameters that can be adjusted in order to achieve optimal results. The output layer in the initial DBN is used to automatically compute class dependent features which will be used as the input to the final classifier.

Each outlier location associated with a target class (street crossing, traffic light and roundabout) will generate an input vector consisting of a sequence of 21 consecutive pairs (v, a). Part of the input samples will be used to train the DBN + classifier and part of them to validate it. We will use a 10-fold validation schema in which the samples are divided into a 90% training and a 10% validation sets. The process is repeated 10 times so that all the samples are considered once in the validation set.

A first configuration of a DBN consisting of 2 layers with 10 and 2 hidden units has been tested with 3 different classifiers (SVM with a quadratic kernel, SVN with a Gaussian kernel and KNN). Table 2 captures the results for the case of SVN with quadratic kernel for the first dataset. The recall (fraction of relevant samples that are retrieved) is best for the street crossing class (0.95). The precision (fraction of retrieved samples that are relevant) is best both for the street crossing and roundabout classes (0.87). The traffic light shows the worst results both for precision and recall. The 32% of the traffic light associated samples are considered as street crossing and 15% as roundabouts. Table 3 captures similar results for the second dataset. In this case, instead of

Table 2
– Confusion matrix (10,2) SVM quadratic kernel using the first dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	1186	49	15	0.95
Traffic light (TL)	135	222	63	0.53
Roundabout (RA)	49	59	529	0.83
Precision	0.87	0.67	0.87	

Table 3
– Confusion matrix (10,2) SVM quadratic kernel using the second dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	110	4	19	0.83
Traffic light (TL)	13	30	20	0.48
Roundabout (RA)	31	14	73	0.62
Precision	0.71	0.63	0.65	

Table 4–
Confusion matrix (10,2) SVM Gaussian kernel using the first dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	1219	25	6	0.98
Traffic light (TL)	186	184	50	0.44
Roundabout (RA)	146	16	475	0.75
Precision	0.79	0.82	0.89	

Table 5–
Confusion matrix (10,2) SVM Gaussian kernel using the second dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	95	1	37	0.71
Traffic light (TL)	15	10	38	0.16
Roundabout (RA)	15	0	103	0.87
Precision	0.76	0.91	0.58	

Table 6–
Confusion matrix (10,2) KNN using the first dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	1185	49	16	0.95
Traffic light (TL)	126	241	53	0.57
Roundabout (RA)	68	28	541	0.85
Precision	0.86	0.76	0.89	

Table 7–
Confusion matrix (10,2) KNN using the second dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	112	5	16	0.84
Traffic light (TL)	21	26	16	0.41
Roundabout (RA)	32	9	77	0.65
Precision	0.68	0.65	0.71	

having a single participant driving several times on the same driving path, a group of 10 different drivers have been used, each participant driving only once. The scenario in this second dataset is therefore more demanding and results are expected to worsen if compared to the first dataset. However, the results are still acceptable for the street crossing class with a 0.83 recall and 0.87 precision. Traffic lights continue to be the class showing the worst results. Using a classifier based on SVN with a Gaussian kernel achieves similar results (improving the recall for the street crossing class and the precision for the traffic light class but decreasing the recall in the case of the traffic light class). The results for the first dataset are captured in Table 4. Table 5 captures similar results for the second dataset. Using KNN as the output classifier also achieves similar results as shown in Tables 6 and 7.

Using a 3 layer DBN will require more time to compute. We have selected an architecture with 10, 5 and 1 hidden units in the DBN so that the number of output features is half of the number in the 2 layer DBN previously presented and therefore the classifier can run faster (compensating the time consumed in the DBN). Results for the first dataset using the same classifiers are captured in Tables 8, 10 and 12. In this case the results are slightly worse. Results for the second dataset using the same classifiers are captured in Tables 9, 11 and 13.

We can try to improve the classification results by considering how many times a particular point is detected as an outlier. The idea is that traffic lights will be green some of the times and therefore the number of outliers associated to them will be smaller than in the case of roundabouts and street crossings. Moreover, in some clear traffic conditions it is likely that some roundabouts do not generate outliers so that the number of outliers associated with street crossings is expected to be even higher than in the case of roundabouts. We define the degree of atypicity (da) of a particular location as the relative number of times that a location is detected as an outlier. Adding the degree of atypicity (da) as an input variable to the output classifier and trying with both configurations previously used for the DBN in the case of the KNN classifier provides the results captured in Tables 14 and 16 for the first dataset. The results improve significantly for the case of traffic light classification, more prominently in the case of a 3 layer DBN architecture. This architecture is able to detect 96% of the street crossings, 69% of the traffic lights and 89% of the roundabouts. The results for the second dataset are presented in Tables 15 and 17. Using the output of the 3 layer DBN and the degree of atypicity (da) for each location as the input to a KNN classifier provides a precision close to 90% for the street crossings and close to 80% for the traffic lights and roundabouts for the second dataset, which includes information of 10 different drivers following the same path only once. The values achieved for the recall with this second dataset are close but a bit under those achieved for the case of the first dataset as expected.

Table 8
– Confusion matrix (10,5,1) SVM quadratic kernel using the first dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	1178	35	37	0.94
Traffic light (TL)	150	121	149	0.29
Roundabout (RA)	91	20	526	0.83
Precision	0.83	0.69	0.74	

Table 9
– Confusion matrix (10,5,1) SVM quadratic kernel using the second dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	95	9	29	0.71
Traffic light (TL)	23	26	14	0.41
Roundabout (RA)	35	12	71	0.60
Precision	0.62	0.55	0.62	

Table 10
– Confusion matrix (10,5,1) SVM Gaussian kernel using the first dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	1192	28	30	0.95
Traffic light (TL)	155	150	115	0.36
Roundabout (RA)	99	17	521	0.82
Precision	0.82	0.77	0.78	

Table 11
– Confusion matrix (10,5,1) SVM Gaussian kernel using the second dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	107	2	24	0.80
Traffic light (TL)	50	8	5	0.13
Roundabout (RA)	45	0	73	0.62
Precision	0.53	0.80	0.72	

Table 12
– Confusion matrix (10,5,1) KNN using the first dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	1163	53	34	0.93
Traffic light (TL)	141	190	89	0.45
Roundabout (RA)	113	30	494	0.78
Precision	0.82	0.70	0.80	

Table 13
– Confusion matrix (10,5,1) KNN using the second dataset.

True/predicted	ST	TL	RA	Recall
Street crossing (ST)	99	4	30	0.74
Traffic light (TL)	36	20	7	0.32
Roundabout (RA)	37	5	76	0.64
Precision	0.58	0.69	0.67	

Table 14
– Confusion matrix (10,2 + da) KNN using the first dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	1193	46	11	0.95
Traffic light (TL)	59	291	70	0.69
Roundabout (RA)	35	46	556	0.87
Precision	0.93	0.76	0.87	

Table 15
– Confusion matrix (10,2 + da) KNN using the second dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	119	4	12	0.88
Traffic light (TL)	9	41	13	0.65
Roundabout (RA)	13	14	91	0.77
Precision	0.84	0.69	0.78	

Table 16
– Confusion matrix (10,5,1 + da) KNN using the first dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	1198	41	11	0.96
Traffic light (TL)	53	288	79	0.69
Roundabout (RA)	31	42	564	0.89
Precision	0.93	0.78	0.86	

Table 17
– Confusion matrix (10,5,1 + da) KNN using the second dataset.

True/Predicted	ST	TL	RA	Recall
Street crossing (ST)	119	3	11	0.89
Traffic light (TL)	5	42	16	0.67
Roundabout (RA)	12	9	97	0.82
Precision	0.88	0.78	0.78	

Table 18–
Confusion matrix. Threshold rth = 0.97 using the first dataset.

Detected as	TP	FP	Precision
Street crossing (ST)	5	2	0.71
Traffic light (TL)	1	0	1
Roundabout (RA)	5	1	0.83

Table 19
– Confusion matrix. Threshold rth = 0.96 using the first dataset.

Detected as	TP	FP	Precision
Street crossing (ST)	15	8	0.65
Traffic light (TL)	4	4	0.50
Roundabout (RA)	7	3	0.70

Table 20–
Confusion matrix. Threshold rth = 0.97 using the second dataset.

Detected as	TP	FP	Precision
Street crossing (ST)	0	0	
Traffic light (TL)	2	0	1
Roundabout (RA)	2	0	1

Table 21
– Confusion matrix. Threshold rth = 0.96 using the second dataset.

Detected as	TP	FP	Precision
Street crossing (ST)	3	0	1
Traffic light (TL)	2	0	1
Roundabout (RA)	2	2	0.5

Table 22
– Confusion matrix. Threshold rth = 0.95 using the second dataset.

Detected as	TP	FP	Precision
Street crossing (ST)	5	0	1
Traffic light (TL)	2	0	1
Roundabout (RA)	2	3	0.4

4.2. Detecting road elements while driving

In order to detect road elements in the target classes while driving, and avoid characterizing the null class, a different schema has been used as described in Section 2.4. The driving samples associated with 2 particular instances of each class (2 traffic lights, 2 street crossings and 2 roundabouts) have been used to train 3 autoencoders with 10 hidden units each. The detection process uses the output of the outlier detection schema in order to pre-select candidate points which could be associated with one of the target classes. The preselected samples are then encoded and decoded with the previously trained autoencoders and the Pearson's correlation coefficient is used to assess how similar the reconstructed output is to the pre-selected input. A threshold is used in order to consider the pre-selected sample a potential member of the class. If the similarity with more than 1 class is above the threshold, the sample is assigned to the class with a higher similarity. If the similarity with all the 3 classes is below the threshold, the sample is considered to belong to the null class. The results for the first dataset are captured in Tables 18 and 19. Table 18 captures the detection results for the case of a threshold value of $r = 0.97$. In this case 7 locations are considered to be street crossings (2 of them being false positives), 1 location is correctly detected as being a traffic light and 6 locations (1 being a false positive) are detected as roundabouts. The precision is also captured in the table. However, only a limited number of elements are detected. In order to increase the number of elements detected, the threshold value can be lowered. Table 19 captures the results for a threshold value of $r = 0.96$. In this case 15 street crossings, 4 traffic lights and 7 roundabouts are correctly detected but at the cost of decreasing the values for the precision to 0.65, 0.50 and 0.70 respectively.

The results for the second dataset are captured in Tables 20, 21 and 22. In this case, a threshold value of 0.97 does not provide any false positive but is not able to provide true positive samples for the case of street crossings. We have therefore used the values of 0.96 (Table 21) and 0.95 (Table 22) in order to assess the performance of the algorithm. Using a threshold of 0.96 for the Pearson's correlation index is able to detect 3 street crossings and 2 traffic lights with no false positive. Lowering the threshold to 0.95, a total of 5 street crossings are detected with no false positives. However, the number of false positives in the case of roundabouts worsens as the threshold is decreased.

As captured in Section 1, previous studies have used crowdsensing techniques in order to improve results. As a future work, we plan to increase the number of drivers and test drives to be able to filter the results from the detection process based on the number of times that a particular location is detected as such.

4.3. *Comparing results with previous studies*

In the comparison of results of the algorithms presented in this section, the recall, precision, the type of sensors and machine learning techniques used, together with the scope of each research study are included in order to provide an overall picture.

The results presented in Table 16 show a combined recall and precision of 0.89 for the first dataset when classifying elements belonging to any of the 3 target classes in this study: traffic lights, road crossings and roundabouts. The results presented in Table 17 show a combined recall and precision of 0.82 for the second dataset when classifying elements belonging to any of the 3 target classes in this study: traffic lights, road crossings and roundabouts. We have used an automatic feature extraction mechanism based on a DBN (enhanced by the degree of atypicity (da) computed for each candidate location) together with a final classifier based on KNN and SVN with two different kernels. The proposed algorithm uses the GPS sensor sampled at 1 Hz as the input information in order to generate patterns of speed and acceleration at points classified as outliers using both an intra-drive and inter-drive sample selection process. The algorithm has been extended in order to provide real-time detection while driving without the need to characterize the null class.

Ren & Liu, 2016, proposed a related system that tries to detect road potholes while driving. They used the GPS sensor to perform a pre-clustering of samples based on the travelling speed and the accelerometer to perform actual pothole detection. 2 axis of the accelerometer sensors are taken into account. The k-means clustering algorithm is used to automatically detect the samples belonging to the pothole cluster. The authors state that their algorithm improves previously existing ones but recognize that they were not able to isolate potholes form certain abnormal traffic conditions.

D'Andrea & Marcelloni, 2016, detected traffic congestion and incidents from GPS trace analysis. They only used GPS information (both location and speed) in order to feed a heuristic algorithm for traffic congestion detection. The algorithm achieved a recall of 0.916, but the scope was limited to detecting traffic congestion cases.

Ghosh & Smith, 2014, proposed a mechanism for automatic incident detection adapted to the scenario of signalized urban arterials. A traffic volume database generated based on the use of inductive loop detectors was used. Several machine learning algorithms were used. Best results were achieved when using SVM. A recall of 0.87 was achieved.

Mascetti et al. (2016), proposed a vision recognition based system to detect traffic lights based on mobile devices for pedestrians with visual impairment. The authors achieved a 0.85 recall and the scope was limited to traffic light detection.

Aly et al., 2017, proposed the use of several sensors on mobile devices and the combination of the information extracted from multiple users together (crowdsensing) in order to detect several road elements such as bumps and rail crossings. The proposed algorithm used GPS, accelerometer, gyroscope, magnetometer, and cellular network information to feed a probabilistic model. Several road elements such as bumps (recall = 0.90, precision = 0.97) and rail crossings (re-call = 0.92, precision = 0.76) were automatically classified. Despite the fact of using several sensors and several users together, the per-formance achieved for detecting elements such as bumps and rails-crossings is worse compared to the results obtained using the method proposed in this paper for classifying street crossing and roundabouts.

5. Conclusions

In this paper, a novel mechanism to automatically detect road-infrastructural elements based on the use of GPS sensor data while driving is presented. The results focus on three particular classes of elements: traffic lights, street crossings and roundabouts. The mechanism is validated using two different datasets. A first dataset with several drives following the same circuit by a single driver and a second dataset in which 10 different drivers follow a particular circuit once each. Infrastructural information such as the location of traffic lights and their red/green periods, the location of roundabouts or street crossings and the common speed patterns while traversing those elements can be used to provide better estimations of the travel time to reach a parti-cular destination using a routable map. The travel time estimates could be personalized to a particular user, the time of the day and the day of the week.

The proposed mechanism combines a pre-selection process based on a novel outlier detection algorithm and a DBN + output classifier. The outlier detection algorithm uses the speed and acceleration patterns both in intra-drive and inter-drive data in order to detect candidate locations that could contain a relevant infrastructural element. Intra-drive data is used to detect abnormalities in speed and acceleration patterns that can be due either to infrastructural elements as well as traffic conditions. Inter-drive outlier detection is aimed at separating sporadic traffic related conditions to fix infrastructural elements. Using a DBN, automatic features are extracted from speed and acceleration patterns that capture the particularities of each road element. A final classifier, based on KNN and SVM (with both a quadratic and a Gaussian kernel) algorithms, is used achieving a combined recall and precision of 0.89 for the first dataset and a combined recall and pre-cision of 0.82 for the second dataset. The results are enhanced by cal-culating the percentage of times that a particular candidate location is pre-selected as an outlier in similar drives and feeding this information into the output classifier together with the DBN automatic features.

In order to apply the detection algorithm to real-time scenarios and avoiding to have to characterize the null class, a variation has also been introduced into the proposed architecture based on a similarity measure. Instead of a DBN, a schema based on the use of auto-encoders together with a similarity measure based on the Pearson's correlation coefficient is used. The results show acceptable values in the precision if the recall is accepted to be low but degrade when the number of detected elements increases. These results could be improved using a crowdsensing approach as other related studies have previously presented. As a future work, an improved version of the proposed algorithm enhanced by combining the elements detected by multiple users will be generated. Another improvement of the algorithm will include the information from other sensors such as the accelerometer available in most mobile devices together with the GPS information.

Acknowledgements

The research leading to these results has received funding from the "HERMES-SMART DRIVER" project TIN2013-46801-C4-2-R (MINECO), funded by the Spanish Agencia Estatal de Investigación (AEI), and the “ANALYTICS USING SENSOR DATA FOR FLATCITY” project TIN2016-77158-C4-1-R (MINECO/ ERDF, EU) funded by the Spanish Agencia Estatal de Investigación (AEI) and the European Regional Development Fund (ERDF).

References

Aly, H., Basalamah, A., & Youssef, M. (2017, Oct. 1). Automatic rich map semantics identification through smartphone-based crowd-sensing. IEEE Transactions on Mobile *Computing, 16*(10), 2712–2725.

Bujari, A., Licar, B., & Palazzi, C. E. (2012, January). *Movement pattern recognition through smartphone's accelerometer. In consumer communications and networking conference (CCNC), 2012 IEEE.* IEEE502–506.

Corcoba Magaña, V., & Muñoz-Organero, M. (2016). WATI: Warning of traffic incidents for fuel saving. *Mobile Information Systems, 2016*.

D'Andrea, E., & Marcelloni, F. (2016). Detection of traffic congestion and incidents from GPS trace analysis. *Expert Systems with Applications*.

Dunlop, M. D., Roper, M., Elliot, M., McCartan, R., & McGregor, B. (2016, May). Using smartphones in cities to crowdsource dangerous road sections and give effective in-car warnings. *Proceedings of the SEACHI 2016 on smart cities for better living with HCI and UX* (pp. 14–18). ACM.

Ferri, C. (2016). Identifying the sport activity of GPS tracks. *Procedia Computer Science, 80*, 301–312.

Ghosh, B., & Smith, D. P. (2014). Customization of automatic incident detection algorithms for signalized urban arterials. *Journal of Intelligent Transportation Systems, 18*(4), 426–441.

Hemminki, S., Nurmi, P., & Tarkoma, S. (2013, November). Accelerometer-based transportation mode detection on smartphones. *Proceedings of the 11th ACM conference on embedded networked sensor systems* (pp. 13). ACM.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527–1554.

Kamilaris, A., & Pitsillides, A. (2016, Dec.). Mobile phone computing and the internet of things: A survey. *IEEE Internet of Things Journal, 3*(6), 885–898.

Lan, G., Xu, W., Khalifa, S., Hassan, M., & Hu, W. (2016, March). Transportation mode detection using kinetic energy harvesting wearables. In Pervasive computing and communication workshops (PerCom workshops), 2016 IEEE international conference on (pp. 1–4). IEEE.

Ma, M. X., Ngan, H. Y., & Liu, W. (2016). Density-based outlier detection by local outlier factor on largescale traffic data. *Electronic Imaging, 2016*(14), 1–4.

Magaña, V. C., Organero, M. M., Fisteus, J. A., & Fernández, L. S. (2016, June). *Estimating the stress for drivers and passengers using Deep Learning. 2016*. Almeria. Spain: JARCA.

Mascetti, S., Ahmetovic, D., Gerino, A., Bernareggi, C., Busso, M., & Rizzi, A. (2016). Robust traffic lights detection on mobile devices for pedestrians with visual impairment. *Computer Vision and Image Understanding, 148*, 123–135.

Mathibela, B., Newman, P., & Posner, I. (2015). Reading the road: Road marking classification and interpretation. *IEEE Transactions on Intelligent Transportation Systems, 16*(4), 2072–2081.

Munoz-Organero, M., & Lotfi, A. (2016). Human movement recognition based on the stochastic characterisation of acceleration data. *Sensors, 16*(9), 1464.

Panichpapiboon, S., & Leakkaw, P. (2016). Traffic sensing through accelerometers. *IEEE Transactions on Vehicular Technology, 65*(5), 3559–3567.

Perttula, A., Parviainen, J., & Collin, J. (2016, October). Pedestrian detection with high resolution inertial measurement unit. *SENSORS, 2016 IEEE* (pp. 1–3). IEEE.

Ren, J., & Liu, D. (2016, December). PADS: A reliable pothole detection system using machine learning. *International conference on smart computing and communication* (pp. 327–338). Cham: Springer.

Schneegass, S., Pfleging, B., Broy, N., Schmidt, A., & Heinrich, F. (2013). A data set of real world driving to assess driver workload. *Proceedings of the 5th international conference on automotive user interfaces and interactive vehicular applications (AutomotiveUI '13)* (pp. 150–157). New York, NY, USA: ACM. http://dx.doi.org/10.1145/2516540.2516561 (http://doi.acm.org/10.1145/2516540.2516561).

Tang, J., & Ngan, H. Y. (2016). Traffic outlier detection by density-based bounded local outlier factors. *Information Technology in Industry, 4*(1), 6–18.

Wang, J., Wang, C., Song, X., & Raghavan, V. (2017). Automatic intersection and traffic rule detection by mining motor-vehicle GPS trajectories. *Computers, Environment and Urban Systems, 64*, 19–29.

White, J., Thompson, C., Turner, H., Dougherty, B., & Schmidt, D. C. (2011). Wreckwatch: Automatic traffic accident detection and notification with smartphones. *Mobile Networks and Applications, 16*(3), 285.

Wu, Z., Watanabe, Y., & Ishikawa, M. (2016, November). Hybrid LED traffic light detection using high-speed camera. In intelligent transportation systems (ITSC), 2016 IEEE 19th international conference on (pp. 1235–1241). IEEE.