IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 37, No. 6, December, 2020, pp. 1003-1008
Journal homepage: http://iieta.org/journals/ts

Detection of Unusual Targets in Traffic Images Based on One-Class Extreme Machine Learning

Lei Yu*, Binglin Zhang, Rui Li

School of Information Engineering and Media, Hefei Polytechnic, Hefei 230011, China

Corresponding Author Email: yulei@htc.edu.cn

https://doi.org/10.18280/ts.370612

Received: 8 July 2020
Accepted: 10 October 2020

Keywords:
Traffic images, multiple levels, extreme learning machine (ELM), semi-supervised learning

ABSTRACT

In traffic image target detection, unusual targets like a running dog has not been paid sufficient attention. The mature detection methods for general targets cannot be directly applied to detect unusual targets, owing to their high complexity, poor feature expression ability, and requirement for numerous manual labels. To effectively detect unusual targets in traffic images, this paper proposes a multi-level semi-supervised one-class extreme learning machine (ML-S2OCELM). Specifically, the extreme learning machine (ELM) was chosen as the basis to develop a classifier, whose variables could be calculated directly at the cost of limited computing resources. The hypergraph Laplacian array was employed to improve the depiction of data smoothness, making semi-supervised classification more accurate. Furthermore, a stack auto-encoder (AE) was introduced to implement a multi-level neural network (NN), which can extract discriminative eigenvectors with suitable dimensions. Experiments show that the proposed method can efficiently screen out traffic images with unusual targets with only a few positive labels. The research results provide a time-efficient, and resource-saving instrument for feature expression and target detection.

1. INTRODUCTION

The rapid expansion of automobile industry has led to severe traffic congestion and a surge in traffic accidents, making traffic safety a social concern. To improve traffic safety, the concept of assisted driving came into being. Assisted driving systems have been gradually introduced to driving. The cornerstone of these assisted driving systems is the real-time detection and ranging of traffic targets. As a major branch of image processing and computer vision, target detection and ranging integrate cutting-edge technology in various fields.

Like other image detection methods, traffic target detection algorithm first captures the information of the original image, and then realizes advanced expression of image features through complex spatial transform, e.g., point-to-point or window-to-window mapping. In recent years, deep learning, manifested as neural networks (NNs), have been widely applied in such fields as image recognition, voice and audio analysis, and natural language processing. The accuracy of image recognition could be increased by more than 10% through deep learning. In particular, deep learning can elevate the accuracy of face recognition to 99.47% [1].

Apart from medical image classification [2], face recognition [3], and remote sensing [4], deep NNs have made remarkable achievements in pedestrian and vehicle target detection in the traffic field. The target detection is generally realized in the following steps: identifying the regions of interest (ROIs) in the original image by sliding window traversal or complex region selection algorithm, and checking for the presence of targets in each ROI. Based on convolution NN (CNN), Sermanet et al. [5] proposed an unsupervised learning method for pedestrian detection; the CNN improves the discrimination ability of the original target detection algorithm, without undermining its robustness in detecting occluded or deformed targets. Hosang et al. [6] designed an adaptive image classification network for line detection, which effectively improves the detection accuracy of local features in independent channels. Girshick et al. [7] developed a region-based CNN (R-CNN) that classifies the ROIs preliminarily selected by the CNN.

Image target detection is often treated as a multi-class classification problem. The labeling of positive classes (unusual targets) in the original image is usually easier and more accurate than that of negative classes (non-targets). Hence, this problem has been recently treated by many as a one-class (OC) classification problem, a.k.a. novelty or outlier detection [8]. Different from those in multi-class classification, the data samples of OC classification mainly come from only one class, i.e., the target class, while those of other classes are either too few or uncredible. In other words, the classes of non-target samples are unknown. OC classification is being widely applied in machine fault detection [9], disease detection [10], and credit score [11]. During OC classification, the classifiers observes the target samples carefully, learns how to select unknown samples similar to the target class, and determines the class of each test sample. In general, each classification task is equivalent to describing the data distribution of the target class. The most representative OC classification method is support vector data description (SVDD) [12].

At present, various OC classification methods have been developed and implemented. Early on, OC classification was achieved by estimating the probability density function of the training data [13]. For example, Robert [14] superimposed the estimation of Parzen window density on only one training sample to derive the probability density function, and rejected

any test sample whose estimated probability is below the threshold. However, it often requires a huge number of training samples to deduce the true density distribution. A much simpler method is to find the data distribution domain. For instance, Scholkopf et al. [15] constructed a hyperplane with the largest distance from the origin to separate areas with no or sparse data. Another method is to find a hypersphere that covers most of the target data with the smallest radius [12]. Both methods could be realized through quadratic programming, or sometimes through linear programming [16]. For example, Juszczak et al. [17] designed an OC classifier based on the minimum spanning tree, which determines the class of each sample based on the shortest distance from the sample to the nearest edge of the tree.

Over the years, many OC algorithms have been developed based on artificial NNs (ANNs). For instance, Gutoski et al. [18] proposed a self-encoding (AE) OC classifier from a feedforward NN with only one implicit level; the training of the classifier depends heavily on the backpropagation (BP) algorithm. In addition, much research attention has been paid to the training of ANNs through iterative update, random variable assignment, and least squares algorithm. Extreme learning machine (ELM) is the most representative algorithm for OC. It obtains the implicit level export from the random variables of neurons in the implicit level, and solves export weights by least squares method [19]. Leng et al. [20] put forward an OC ELM that outshines many traditional OC classifiers on various benchmark datasets.

Nevertheless, the existing ELMs cannot be directly applied to our problem of detecting unusual targets in traffic images, mainly due to the following problems: (1) High computing complexity: deep NNs consume lots of computing resources, and a large amount of power, reducing the continuous running time of many battery-powered portable devices; (2) High need for manual labeling: the images collected by the surveillance cameras must be labeled manually to provide sufficient labeled samples for the training of deep NNs, pushing up the workload of personnel; (3) Weak capability of feature expression: Most ELMs adopt an NN with only one implicit level, whose capability of feature extraction is poor. In our problem, if the array of numerous traffic images are vectorized and directly imported to the ELM, the curse of dimensionality will occur and result in the failure of classification.

Therefore, this paper proposes a multi-level semi-supervised OC ELM (ML-S2OCEML) for detecting unusual targets in traffic images. As shown in Figure 1, each original image is segmented into a series of sub-images by one or several fixed-size windows, with the expectation that each sub-image contains as many targets in the same class as possible. Then, the sub-images with unusual targets are processed manually.

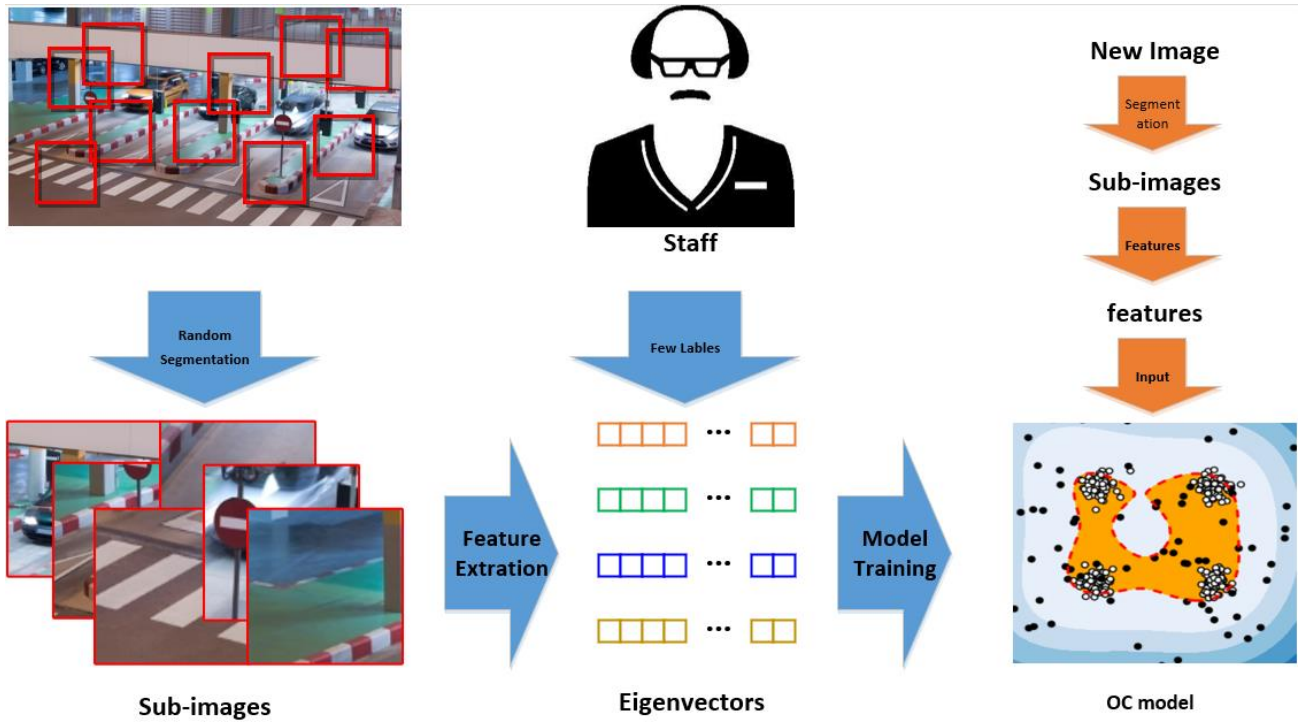

Figure 1. The workflow of the ML-S2OCEML
Note: The blue arrows indicate the training process; the red arrows indicate the online prediction process

The main innovations of the ML-S2OCELM are as follows:

(1) The classifier was developed based on the ELM. The closed solutions of ELM variables could be directly calculated, saving lots of computing resources.

(2) The hypergraph Laplacian array was employed to improve the depiction of data smoothness, making semi-supervised classification more accurate.

(3) A stack auto-encoder (AE) was introduced to implement a multi-level NN, which can extract discriminative eigenvectors with suitable dimensions.

2. METHODOLOGY

2.1 ELM

Proposed by Prof. G.B. Huang, the ELM [21] could be mathematically described as follows: The ELM aims to learn decision rules or approximate functions from a labelled training dataset $\{(\mathrm{x}_i, \mathrm{t}_i)|\mathrm{x}_i \in \mathbb{R}^d, \mathrm{t}_i \in \mathbb{R}^m, i = 1, \ldots, N\}$, where x_i and t_i are the import and export vectors, respectively; N and d are the total number and dimension of the samples,

respectively; m is the number of classes.

The ELM is generally trained in two phases. In the first phase, an implicit level is built with a fixed number of randomly generated mapping neurons. The mapping could be any nonlinear piecewise continuous function, namely, sigmoid function and Gaussian function:

$$g(\mathrm{x};\mathrm{w},b)=\frac{1}{1+\ \exp\left(-(\mathrm{w}^{T}x+b)\right)} \tag{1a}$$

$$g(\mathrm{x};\mathrm{w},b)=\ \exp\left(-b\|\mathrm{x}-\mathrm{w}\|\right) \tag{1b}$$

where, x is a sample; w is the vector of the import weight; b is the bias. Let L be the number of neurons in the implicit level. The weight vectors and biases of L neurons could be generated as $\{\mathrm{w}_1,\dots,\mathrm{w}_L\}$ and $\{b_1,\dots,b_L\}$, respectively. For each x, the corresponding implicit level export vector (i.e., ELM features) could be formulated as:

$$h(\mathrm{x})=[g(\mathrm{x};\mathrm{w}_1,b_1),\dots,g(\mathrm{x};\mathrm{w}_L,b_L)]\in\mathbb{R}^{1\times L} \tag{2}$$

Then, the implicit level export array is:

$$\mathrm{H}=[h(\mathrm{x}_1);\dots;h(\mathrm{x}_N)]\in\mathbb{R}^{N\times L} \tag{3}$$

A significant feature of the ELM is that the mapping function variables from the import level to the implicit level could be generated by random based on arbitrary distribution of continuous probabilities. During training, the only variable to be optimized is the weight of the export from the implicit level to the export level. The variable optimization could be analogized to a regularized least squares problem. Therefore, it is much more efficient to construct an ELM model than NN back-propagation learning.

Let $\beta\in\mathbb{R}^{L\times m}$ be the export weight of the implicit level to the export level. Then, the ELM model could be established as:

$$f(\mathrm{x})=h(\mathrm{x})\beta \tag{4}$$

Consequently, the export weight array could be optimized by minimizing the deviation of network export from target β^*:

$$\min_{\beta}\|\mathrm{H}\beta-\mathrm{T}\|_{\mathrm{F}}^{2} \tag{5}$$

where, $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm; β and T are the export weight array and target array, respectively.

The export weight array β^* could be optimized through least squares method:

$$\beta^{*}=\mathrm{H}^{\dagger}\mathrm{T} \tag{6}$$

where, $H^{\dagger}$ is the pseudo inverse of H. The regularized ELM (RELM) could reduce the training error as much as possible, as well as the export weight norm. Thus, we have:

$$\min_{\beta}\frac{1}{2}C\|\mathrm{H}\beta-\mathrm{T}\|_{\mathrm{F}}^{2}+\frac{1}{2}\|\beta\|_{\mathrm{F}}^{2} \tag{7}$$

where, the second objective function is designed to avoid overfitting; C is the trade-off coefficient. The solution of formula (7) could be expressed as:

$$\beta^{*}=\mathrm{H}^{\mathrm{T}}\left(\frac{\mathrm{I}}{C}+\mathrm{H}\mathrm{H}^{\mathrm{T}}\right)^{-1}\mathrm{T} \tag{8}$$

If the training data far outnumber implicit level neurons, i.e., $N\gg L$, the solution of formula (7) could be expressed as:

$$\beta^{*}=\left(\frac{\mathrm{I}}{C}+\mathrm{H}^{\mathrm{T}}\mathrm{H}\right)^{-1}\mathrm{H}^{\mathrm{T}}\mathrm{T} \tag{9}$$

where, I is a unit array with appropriate dimension.

2.2 OC-ELM

In OC classification, the model can only be trained with the data of the target class. Therefore, the solution of OC-ELM becomes:

$$\min_{\beta}\frac{C}{2}\|\mathrm{H}\beta-\mathrm{t}\|_{\mathrm{F}}^{2}+\frac{1}{2}\|\beta\|_{2}^{2} \tag{10}$$

where, $\|\cdot\|_2$ is the 2-norm. The training dataset could be described as $\{(\mathrm{x}_i,\mathrm{t})|\mathrm{x}_i\in\mathbb{R}^d, i=1,2,\dots,N\}$, with $\mathrm{t}=[t,\dots,t]^{\mathrm{T}}\in\mathbb{R}^N$ being the desired vector. Without loss of generality, t could take any value. The array of export weights could be obtained by substituting array T with t and then solving formula (8) or (9). For the robustness of performance, the OC-ELM could remove a few samples based on the deviation $\varepsilon(x)$ of the network from the target:

$$\varepsilon(\mathrm{x})\triangleq|h(\mathrm{x})-t| \tag{11}$$

In the ideal scenario, the lower the $\varepsilon(x)$, the better the result. If the test sample x has a large error $\varepsilon(x)$, then it could be regarded as an anomaly or outlier that deviates from the target class. In other words, if $\varepsilon(x)\leq\theta$, then x is the target; otherwise, it is an outlier. The threshold θ could be determined in two steps: First, $\varepsilon(x_1),\dots,\varepsilon(x_N)$ are arranged in descending order as $\varepsilon_1\geq\varepsilon_2\geq\dots\geq\varepsilon_N$; then, $\theta=\varepsilon_{\lfloor N\phi\rfloor}$ is set to get the threshold, where $\phi\in(0,1)$ is a manually-set value.

2.3 ML- S2OCELM

There are three core parts of the ML-S2OCELM: hypergraph Laplacian array, multi-level feature extraction, and S2OCELM.

2.3.1 Hypergraph Laplacian array

In semi-supervised learning, it is necessary to introduce a graph Laplacian array to describe the smoothness of the data. Here, a more general form than the graph Laplacian array is introduced, namely, the hypergraph Laplacian array.

In machine learning problems, the data are usually assumed to have a pairwise relationship. A dataset given by a pairwise relationship could be considered a graph, which could be undirected or directional. In some problems, however, it is incomplete to represent the relationship between the samples with simple graphs. This problem could be solved through hypergraph learning. Compared with traditional graphs, the hypergraph uses hyperedges that connect three or more points to represent the complex relationships in the data.

Figure 2 provides an example of hypergraph $G(V, E)$, where V is a set of points (samples), and E is a collection of V subsets satisfying $\bigcup_{e\in E}e=V$. The weight of edge e could be expressed as $w(e)$, the degree of each point v as $d(v)=$

$\sum_{e\in E|v\in E} w(e)$, and the degree of hyperedge e as $\delta(e)=|e|$. In addition, the event array $\mathbb{I}$ corresponding to the hypergraph could be described as $|V|\times|E|$ array. If $v\in e$, then $\mathbb{I}(v,e)=1$; otherwise $\mathbb{I}(v,e)=0$. Furthermore, it could be obtained that $d(v)=\sum_{e\in E} w(e)\mathbb{I}(v,e)$, and $\delta(e)=\sum_{v\in V}\mathbb{I}(v,e)$.

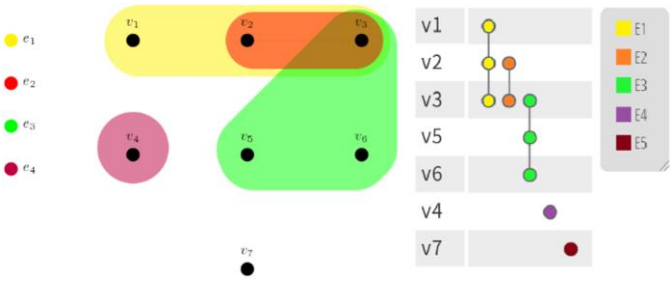

Figure 2. An example hypergraph (left), and the corresponding event array (right)

Let D_v be a diagonal array containing point degrees, D_e be a diagonal array of hyperedge degrees, and W be a diagonal array of hyperedge weights. Then, the hypergraph Laplacian array could be defined based on these variables.

Currently, the Laplacian array of hypergraphs is mainly established in two types of methods. The first type of methods is to build a simple graph from the original hypergraph, and then divide the points through spectral clustering; the second type is to define the hypergraph Laplacian array by analogy to the Laplacian array of a simple graph:

$$\mathrm{L}=\mathrm{I}-\mathrm{D}_v^{-1/2}\mathrm{HWD}_e^{-1}\mathrm{H}^T\mathrm{D}_v^{-1/2} \tag{12}$$

Suppose the adjacency array of the hypergraph is expressed as:

$$\mathrm{W}=\mathrm{HWH}^{\mathrm{T}}-\mathrm{D}_v \tag{13}$$

Then, the Laplacian array of a simple graph could be obtained by representing D_e as 2I:

$$\mathrm{L}=\mathrm{I}-\frac{1}{2}\mathrm{D}_v^{-\frac{1}{2}}\mathrm{HWH}^T\mathrm{D}_v^{-\frac{1}{2}}=\frac{1}{2}\left(\mathrm{I}-\mathrm{D}_v^{-\frac{1}{2}}\mathrm{WD}_v^{-\frac{1}{2}}\right) \tag{14}$$

2.3.2 Multi-level feature extraction

In general, the existing NNs for OC classification are built on only one implicit level for data learning. This structure is inefficient in feature extraction. To make up for the deficiency of traditional OC-ELM in processing image data, this paper extends the OC-ELM to multi-level network. Here, the ELM-based AE (ELM-AE) is adopted for feature learning.

After setting the export of the training set of ELM-AE to $X=[x_1;\ldots;x_N]$, the export weight array of the AE could be obtained to realize feature extraction. Similar to ELM, the import weight array $\mathrm{W}_a=[\mathrm{w}_1^a;\ldots;\mathrm{w}_{L_a}^a]$ and the deviation vector $\mathrm{b}_a=\left[b_1^a;\ldots;b_{L_a}^a\right]$ were first generated; then, the export weight array α was obtained by:

$$\begin{aligned}&\min_{\beta}\frac{1}{2}C_a\|\mathrm{H}_a\alpha-\mathrm{X}^{\mathrm{T}}\|_{\mathrm{F}}^2+\frac{1}{2}\|\alpha\|_{\mathrm{F}}^2\\&\text{s.t.}\quad \mathrm{W}_a\mathrm{W}_a^{\mathrm{T}}=\mathrm{I},\quad \mathrm{b}_a^{\mathrm{T}}\mathrm{b}_a=\mathrm{I}\end{aligned} \tag{15}$$

where, H_a is the implicit level export array of AE; X is the import data serving as the export of feature learning in AE; C_a is the trade-off coefficient between training error and model complexity. Similarly, the export weight array α* of AE could be optimized as:

$$\alpha^*=\mathrm{H}_a^{\mathrm{T}}\left(\frac{\mathrm{I}}{C_a}+\mathrm{H}_a\mathrm{H}_a^{\mathrm{T}}\right)^{-1}\mathrm{X}^{\mathrm{T}} \tag{16}$$

If the training data far outnumber implicit level neurons, i.e., $N\gg L$, the solution of formula (13) could be expressed as:

$$\alpha^*=\left(\frac{1}{C_a}+\mathrm{H}_a^{\mathrm{T}}\mathrm{H}_a\right)^{-1}\mathrm{H}_a^{\mathrm{T}}\mathrm{X}^{\mathrm{T}} \tag{17}$$

Finally, the export of ELM-AE could be obtained as $Y=g(\alpha^*X)$. In the ML-S2OCELM algorithm, AEs were stacked to conduct feature learning. On this basis, the encoded feature Y was imported to following AEs for feature learning. After that, the S2OCELM algorithm was called to implement classification. It was assumed that K ELM-AEs are for feature extraction, and X_{k-1} is the coding feature of the k-1-th AE. Then, the encoded feature of the k-th level could be expressed as:

$$\mathrm{X}_k=g(\alpha_{k-1}^*\mathrm{X}_{k-1}) \tag{18}$$

where, α_{k-1}^* is the export weight array of the k-1-th AE (k=1,…, K).

2.3.3 S2OCELM

Using the encoded features, S2OCELM was used for classification in the final phase. In the case of semi-supervised learning, there are fewer labeled data $\{\mathrm{X}_l,\mathrm{Y}_l\}=\{x_i,y_i\}_{i=1}^l$ than unlabeled data $\mathrm{X}_u=\{x_j\}_{i=1}^u$, where l and u are the number of labeled data and that of unlabeled data, respectively. For the lack of labeled data, the manifold regularization was introduced to improve the classification accuracy with unlabeled data:

$$\min_{\beta}\frac{C}{2}\|\mathrm{JH}_K\beta-\hat{\mathrm{t}}\|_{\mathrm{F}}^2+\frac{\lambda}{2}\mathrm{f}^{\mathrm{T}}\mathrm{Lf}+\frac{1}{2}\|\beta\|_2^2 \tag{19}$$

where, H_K is an implicit export array generated from X_K; $\mathrm{L}\in\mathbb{R}^{(l+u)\times(l+u)}$ is a hypergraph Laplacian array; $\mathrm{f}\in\mathbb{R}^N$ is the export array of the network, whose row vector is $f(x_i)$; λ is the trade-off coefficient; J is the diagonal array where the first l elements are 1, and the rest are 0; $\hat{\mathrm{t}}\in\mathbb{R}^N$ where the first l rows are equal to t and the next u rows are equal to 0. Furthermore, formula (19) could be rewritten as:

$$\min_{\beta}\mathcal{O}\triangleq\frac{C}{2}\|\mathrm{JH}_K\beta-\hat{\mathrm{t}}\|_{\mathrm{F}}^2+\frac{\lambda}{2}\beta^{\mathrm{T}}\mathrm{H}_K^{\mathrm{T}}\mathrm{LH}_K\beta+\frac{1}{2}\|\beta\|_2^2 \tag{20}$$

Let the gradient of $\mathcal{O}$ with respect to β be 0:

$$\nabla\mathcal{O}=\beta+\mathrm{CH}_K^{\mathrm{T}}\mathrm{J}(\mathrm{H}_K\beta-\hat{\mathrm{t}})+\lambda\mathrm{H}_K^{\mathrm{T}}\mathrm{LH}_K\beta=0 \tag{21}$$

Furthermore, the closed-loop solution of the optimal β^* could be obtained. If the training samples have too many the implicit neurons, the above equation is an overdetermined equation. Then,

$$\beta^*=\left(\frac{\mathrm{I}}{C}+\mathrm{H}_K^{\mathrm{T}}\mathrm{JH}_K+\frac{\lambda}{C}\mathrm{H}_K^{\mathrm{T}}\mathrm{LH}_K\right)^{-1}\mathrm{H}_K^{\mathrm{T}}\hat{\mathrm{t}} \tag{22}$$

where, I is a unit array with appropriate dimension; otherwise,

$$\beta^* = H_K^T \left(\frac{I}{C} + JHH^T + \lambda LHH^T \right)^{-1} \hat{t} \quad (23)$$

3. EXPERIMENTS AND RESULTS ANALYSIS

A total of 50,000 images were extracted from street surveillance videos, in which the unusual targets (running dogs) appeared in 2,436 images. Then, 40% of the target images were organized into the test set, and the other 60% into the training set. The test set was established by random selection for 10 times. After the two sets were fixed, a certain percentage of data was selected as labeled data for 10 times in each semi-supervised experiment. Thus, under the same labeling ratio, the same algorithm was tested 100 times. As a result, all the following metrics were calculated as the mean of 100 results.

The performance of each algorithm was measured by missing detection ratio (MDR) and false alarm ratio (FAR):

$$\text{MDR} = \frac{\text{The number of unrecognized target images}}{\text{The number of target images}}$$

$$\text{FAR} = \frac{\text{The number of images incorrectly recognized as target images}}{\text{The number of images recognized as target images}}$$

The experiments were conducted on a desktop computer operating on an Intel Core 7 processor (3.4GHz, 8GB RAM), using Python as the programming language, and Spyder as the integrated compilation environment. The variables were configured as: $C_a=10^{-3}$, $C=10^{-2}$, $\lambda=10^{-4}$, $K=5$, $L_a=400$, and $L=500$.

Table 1. The MDR and FAR (%)

Method / Labels	ML-ELM		ML-S2ELM		ML-OCELM		ML-S2OCELM	
	MDR	FAR	MDR	FAR	MDR	FAR	MDR	FAR
5%	31.3	29.6	21.1	33.9	49.3	12.5	22.5	34.4
10%	28.2	25.6	18.7	28.0	41.1	10.8	18.5	30.3
15%	24.9	24.1	12.3	23.3	38.2	10.3	14.6	25.1
20%	21.7	21.7	10.5	20.4	35.8	8.9	12.8	22.9
30%	18.6	19.5	7.8	18.2	31.7	7.8	8.1	19.4

To reflect the superiority of the proposed algorithm, two supervised learning algorithms (ML-ELM and ML-OCELM) and two semi-supervised learning algorithms (ML-S2ELM and ML-S2OCELM) were chosen as contrastive algorithms. All algorithms used the same feature extraction level. The target detection results of the algorithms are compared in Table 1, where the percentage of labels indicates the proportion of images taken from the target class for labeling. The recognition results of our algorithm are also displayed in Figure 3.

As shown in Table 1, the comparison between ML-ELM and ML-OCELM indicates that describing the detection of unusual targets in traffic images as an OC classification problem could greatly reduce the MAR, but significantly increase the MDR.

Comparing ML-S2ELM and ML-S2OCELM, it is easy to find that the distribution information of unlabeled samples could be fully utilized under the semi-supervised mechanism. The OC classification had slightly lower MDR and FAR than the multi-class classification, but the difference is so small as to be negligible. More importantly, ML-S2OCELM does not need to mark most non-target images in the sample, which greatly reducing the labeling work.

Comparing ML-ELM and ML-S2ELM, it is obvious that, when the labeling ratio is low, the semi-supervised mechanism increased the FAR to a certain extent; however, as the labeling ratio increased, the two methods almost had the same FAR.

In addition, judging by the relationship between the MDR/FAR and the labeling ratio, with the growth in the labeled ratio, the MDR/FAR decreased to a certain extent, and the performance difference between supervised and semi-supervised learnings continued to narrow.

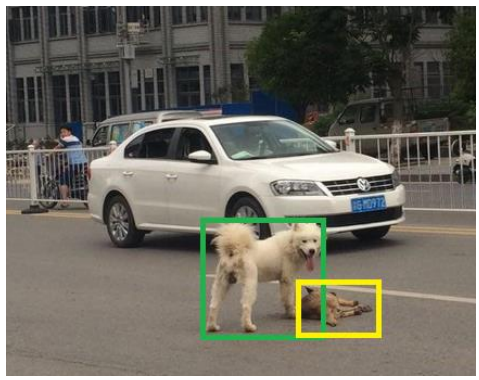

Figure 3. The recognition effect of the proposed algorithm

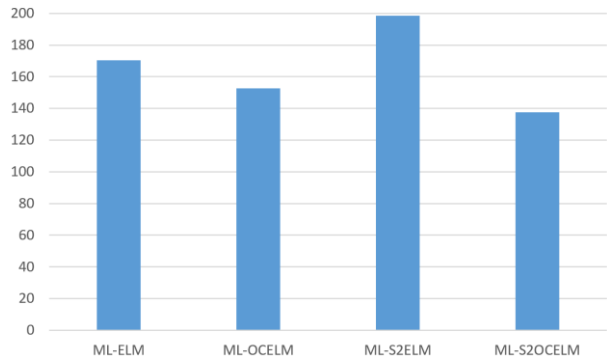

Figure 4. The runtimes of different algorithms

In addition, the runtimes of these algorithms in unusual target detection were compared on 200 images. As shown in Figure 4, the proposed algorithm achieved a high recognition accuracy with the shortest runtime.

4. CONCLUSIONS

This paper proposes a method called ML-S2OCELM for detecting unusual targets in traffic images. Specifically, an ELM was taken as a prototype classifier for secondary development. Due to the closed solution of ELM variables, our classifier does not require too many computing resources, which improves the training speed. Moreover, a stacked AE was introduced to implement a multi-level NN, which can extract distinctive dimensional eigenvectors. In addition, the hypergraph Laplacian array was called to describe the smoothness of the data more clearly, and improve the accuracy of semi-supervised classification. Finally, the detection of unusual targets in traffic images was treated as an OC classification problem, which greatly reduces the labeling work without loss of accuracy, thereby saving a lot of labors and material resources.

ACKNOWLEDGMENT

The key research projects of natural science in universities in Anhui Province are "study on specific target detection of Transportation Image based on OC-ELM", "study on the design of Baxter robot sorting system based on Machine Vision" and "analysis on spatial distribution of aquatic products supply in Anhui Province based on Gis". The project of the Chinese Academy of Sciences (STS), "stereoscopic information acquisition and dynamic monitoring system for agricultural information" (Grant No.: KFJ-STS-ZDTP-057); Project No. 2014BAL01B00 of "Fifteen" national science and Technology Support Plan for Rural Areas Project "Hollow village comprehensive renovation informatization support technology and system platform".

REFERENCES

[1] Taigman, Y., Yang, M., Ranzato, M.A., Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1701-1708.

[2] Sudharshan, P.J., Petitjean, C., Spanhol, F., Oliveira, L. E., Heutte, L., Honeine, P. (2019). Multiple instance learning for histopathological breast cancer image classification. Expert Systems with Applications, 117: 103-111. https://doi.org/10.1016/j.eswa.2018.09.049

[3] Deng, J., Guo, J., Xue, N., Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4690-4699.

[4] Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., Johnson, B.A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. ISPRS Journal of Photogrammetry and Remote Sensing, 152: 166-177. https://doi.org/10.1016/j.isprsjprs.2019.04.015

[5] Sermanet, P., Kavukcuoglu, K., Chintala, S., LeCun, Y. (2013). Pedestrian detection with unsupervised multi-stage feature learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3626-3633.

[6] Hosang, J., Omran, M., Benenson, R., Schiele, B. (2015). Taking a deeper look at pedestrians. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4073-4082.

[7] Girshick, R., Donahue, J., Darrell, T., Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580-587.

[8] Tax, D.M.J. (2002). One-class classification: Concept learning in the absence of counter-examples.

[9] Shin, H.J., Eom, D.H., Kim, S.S. (2005). One-class support vector machines—an application in machine fault detection and classification. Computers & Industrial Engineering, 48(2): 395-408. https://doi.org/10.1016/j.cie.2005.01.009

[10] Cabral, G.G., de Oliveira, A.L.I. (2014). One-class Classification for heart disease diagnosis. In 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2551-2556. https://doi.org/10.1109/SMC.2014.6974311

[11] Kamaruddin, S., Ravi, V. (2016). Credit card fraud detection using big data analytics: Use of PSOAANN based one-class classification. In Proceedings of the International Conference on Informatics and Analytics, pp. 1-8. https://doi.org/10.1145/2980258.2980319

[12] Tax, D.M., Duin, R.P. (2004). Support vector data description. Machine Learning, 54(1): 45-66. https://doi.org/10.1023/B:MACH.0000008084.60811.49

[13] Parzen, E. (1962). On estimation of a probability density function and mode. The Annals of Mathematical Statistics, 33(3): 1065-1076.

[14] Robert, P.W. (1976). On the choice of smoothing variables for Parzen estimators of probability density functions. IEEE Transactions on Computers, 25(11): 1175-1179.

[15] Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C. (2001). Estimating the support of a high-dimensional distribution. Neural Computation, 13(7): 1443-1471. https://doi.org/10.1162/089976601750264965

[16] Pekalska, E., Tax, D.M., Duin, R. (2003). One-class LP classifiers for dissimilarity representations. In Advances in Neural Information Processing Systems, 777-784.

[17] Juszczak, P., Tax, D.M., Pe, E., Duin, R.P. (2009). Minimum spanning tree based one-class classifier. Neurocomputing, 72(7-9): 1859-1869. https://doi.org/10.1016/j.neucom.2008.05.003

[18] Gutoski, M., Ribeiro, M., Aquino, N.M.R., Lazzaretti, A.E., Lopes, H.S. (2017). A clustering-based deep autoencoder for one-class image classification. In 2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI), pp. 1-6. https://doi.org/10.1109/LA-CCI.2017.8285680

[19] Raghuwanshi, B.S., Shukla, S. (2018). Class-specific extreme learning machine for handling binary class imbalance problem. Neural Networks, 105: 206-217. https://doi.org/10.1016/j.neunet.2018.05.011

[20] Leng, Q., Qi, H., Miao, J., Zhu, W., Su, G. (2015). One-class classification with extreme learning machine. Mathematical Problems in Engineering, 2015.

[21] Huang, G.B., Zhu, Q.Y., Siew, C.K. (2006). Extreme learning machine: Theory and applications. Neurocomputing, 70(1-3): 489-501. https://doi.org/10.1016/j.neucom.2005.12.126