Object Classification for Real-Time Video-Surveillance Applications

S Boragno(1)(2), B Boghossian(1), D Makris(2), S Velastin(1)(2)

(1) Ipsotek Ltd - {simone.boragno,boghos.boghossian}@ipsotek.com
(2) Digital Imaging Research Centre, Kingston University - {d.makris,sergio.velastin}@kingston.ac.uk

Keywords: Object classification, video analytics, video surveillance.

Abstract

Object classification is a fundamental step in automatic video-surveillance that allows improved tracking and a more accurate description of events. However, as real-world applications need a real-time, flexible, easy and quick to configure solution, the design of a practical object classification algorithm becomes a challenge. This paper analyses advantages and disadvantages of different frameworks presented in the literature, with particular focus on the ones that are more suitable for real-world operation. A learning-based solution using a reduced training set is proposed, demonstrating that it overcomes many of the limitations associated with other algorithms.

1 Introduction

Automated Video and Audio surveillance is being applied in a wide range of applications such as traffic management, public transport, detection of anti-social behaviour and prevention of crime and terrorism. In most video based surveillance solutions, the scene background is measured over time to detect the objects in the scene that do not belong to the static background. In most implementation of foreground object detection, these objects are initially detected as “blobs”. Object classification attempts to identify the type of object(s) being observed and hence it can add crucial information to video analytics. Object classification therefore allows for more detailed description of events or behaviours. Furthermore it increases detection accuracy and it creates high level metadata that annotate the video sequence.

To illustrate the importance of this functionality it is sufficient to consider some practical scenarios such as that of distinguishing cars from motorcycles, vans and lorries in a vehicle counting application, bags and people for abandoned package detection, groups of people from single persons for intrusion detection in restricted access areas, etc. Moreover, metadata generation is a time-saving functionality for video retrieval and post event analysis.

Unfortunately, the problem of object classification in real-world applications is still open, despite effort by many researchers to propose robust and practical methods to identify the object type. This is mainly due to the particular challenges presented by real-world applications, in which high accuracy, flexibility, real-time operation and cost effectiveness are required.

This paper is organised as follows. In Section 2, the previous work in the literature is discussed, with particular focus on approaches amenable to real-time implementation on embedded systems with limited computing resources. The challenges faced by industrial systems for automatic video-surveillance are outlined in Section 3, referring in particular to the Ipsotek Visual Intelligence Platform. Its system architecture is presented to illustrate the constraints and the resources available to an object classification framework. This review is necessary to highlight the limitations and advantages of the wide range of solutions that have been published on this topic in the last few years. In Section 4 three algorithms (rule-based, k-means and Adaboost) for scene-independent classification are analyzed and tested. An approach to resolve the classification problem is proposed in Section 5, where a flexible scene-dependent framework is demonstrated to achieve more accurate results. Finally, section 6 presents conclusions.

2 Previous research

Most relevant approaches to object classification in the literature have been based on two main processing steps: feature extraction and feature vector classification.

In the first step object specific features (e.g. height, width, area and corner positions) are extracted as means to uniquely identify each class. In the second step these features are analysed to reach a classification based on a single or multiple observations. Papers published in the last few years proposed either a scene-dependent or a scene-independent classification. In the former, a training phase is necessary for every scene; in the latter, algorithms are designed to be unaffected by variations in camera view, camera angle, object position and orientation.

Among the scene-independent solutions proposed, [6,15] distinguish only two object classes (people and vehicles). Although this is generally not sufficient for most video-surveillance applications, the objective of these publications is inline with that of this paper and that is the creation of an object classification module that does not need complex configuration or long training process. Here the authors analyse multiple observations of tracked objects to build confidence in the object class. In [6] the features used are area
and dispersedness, while [15] uses aspect ratio and corner positions.

As scene-independent object classification needs a more
generic approach, in [4] the authors propose a three-class
(person, group of people, vehicle) recognition algorithm that
uses the recurrent motion vectors in an object to identify
human limbs. Therefore, good image quality and accurate
motion detection are necessary for the successful
implementation of this method.

3D modelling is used in [11] to propose a scene-independent
classification method that can support as many object classes
as models. Here the authors distinguish robustly between
similar classes, for example cars from vans by using the
gradient image to match objects to models and hence a high
image resolution is necessary. However, the use of 3D models
is also a limitation because ground plane calibration is
required, and new models could be necessary when working
on a new application or a different scene.

Also scene-independent learning-based solutions present
some drawbacks, the most important one being their high
computational cost. This is the case of Support Vector
Machines (SVM), which are supervised-learning methods that
have been frequently used for the classification of still images.
To our knowledge, the use of SVM for scene-independent
multi-class recognition has not been tested in video-surveillance applications. In many publications the aim is to assign the image a general description of its content. For example in [12], indoor/outdoor scenes or forest/sunset/mountain panoramas are classified. The work reported in [2] is more relevant because the authors describe the use of SVM for scene-independent recognition of different classes (for example, airplanes, birds, boats, buildings, fish, people, cars). However, video-surveillance applications could present more challenging conditions (several similar object classes on the same background).

Another popular algorithm for classification is Adaboost [3].
In [14] this is used for face detection, but the same framework
can be applied to the detection of other classes. However, this
approach is designed for the detection of specific objects and
not for multi-class recognition (which could make it
computationally expensive).

This review demonstrates that the problem of scene-independent classification for video-surveillance applications is still open and presents issues difficult to resolve (these will be analyzed in more details in Section 4).

A scene-dependent framework that is trained for each
different camera view usually provides more accurate results.
Such a solution is presented in [8], where the authors use a
large set of blob-based features that are processed by an
Adaboost network. This allows automatic selection of the
most significant features. Unfortunately, only person/vehicle
classification is considered. Moreover, Adaboost can have a
high computational cost (this problem is analyzed in Section
4.3), and the training process is impractical for a real-world
system.

As the labelling process in scene-dependent frameworks is
too long for practical applications, some methods for the
reduction of the training set have been recently published. For
example, [5] presents a co-training technique to automatically
label unlabelled samples to add them to the training set.

Based on the method presented in [14], the authors use
Adaboost for car detection. This algorithm uses two different
classifiers (actual image and background subtracted image),
so that co-training can be used to label unlabelled objects for
which one of the classifiers is “confident”. Results are
demonstrated to improve when the Adaboost network is
trained by using the automatically labelled samples.

Unfortunately, the number of initially labelled instances could
become excessive when dealing with multi-class recognition.

In [10] a simple but efficient technique to minimize the
training set for tracked objects is presented. The idea is to use
the label assigned in a certain frame to label all the other
observations (in other frames) of the same object.

The main drawback of this method is that, in video-surveillance applications, crowded scenes may present many inconsistent tracks that would produce a poor training set.

3 The challenge

The performance of automatic video-surveillance algorithms
is often stretched when they face real-world problems. This is
mainly due to the continuously changing environmental
conditions and the uncontrolled behaviours of objects in the
scene that can give rise to previously unseen scenarios and
also the need for real-time response in most security related
implementations. Moreover, industrial systems need to be
reliable, flexible, scalable and cost effective.

![Figure 1: Architecture of the Ipsotek VI Platform](image)

Therefore the algorithms outlined in Section 2 need to be
evaluated by considering the practical issues associated with
real-time automatic video-surveillance systems. To provide
an understanding of these issues, we take the Ipsotek Visual
Intelligence Platform as an example.

This system has a server-client multi-thread architecture
(represented in Figure 1) that allows real-time processing.
Each video-processing process runs at 12.5 frames per second
on a NXP Nexperia family Digital Signal Processor (DSP) to
analyse in real-time video signals of standard CCTV cameras
(more details in [1]). The results of the video analytics threads
are analysed by the host application to perform data fusion of video, audio analytics and other real-time data. The host application is also responsible for the configuration of the video processing threads and hence the alerts generated by the data fusion process. The configuration process is flexible and graphical, making it possible for users to quickly define different combinations of video processes to create alerts.

A simple representation of the software running on the DSP is shown by the flow chart in Figure 2. The pixel-level information calculated in the foreground extraction and motion estimation modules is used by a blob-based tracker.

Based on customer surveys we consider the following 9 classes as the ones most commonly present in video-surveillance applications: Package, Person, Bicycle, Motorcycle, Group of people, Crowd, Car, Van and Lorry or Bus. This list is quite generic, however in the case where there are other types of objects in the scene, further classes can be defined.

Table 1 lists the object features that have been used to identify classes in this implementation. These features have been selected from publications [6,8,15] to satisfy the constraints of the challenge that were discussed in the previous section. All spatial information is normalized through camera calibration, in order to obtain comparable values for the same object classes in different scenes and different location in the same scene.

It is important to note that the accuracy of the classifier depends also on the accuracy of the tracker. For this reason, only consistent tracks (where correct detection is maintained for the object \(i\) over \(N\) frames \(k\)) are used both for training and evaluation.

A number of algorithms can be applied to perform a scene-independent classification framework. In the following subsections, three possible methods are presented. All of them perform classification on each video frame and accumulate the results to build confidence on the object class (hence

### Scene-independent approach

The design of a scene-independent classification module is affected by three main elements: the object classes to be recognized, the features on which to base the recognition and the classification algorithm. Scene-independent multi-class recognition should take into account at the design or training stage all possible classes of interest because adding new models or new training sets for different camera views would largely increase the configuration time.
increasing the accuracy of the classification process with time).

The evaluation procedure is based on eleven video sequences that present different views and camera angles (some examples are shown in Figure 3). The ground truth contains 215 objects tracked for a total of 5843 frames. A leave-1-out approach is used to divide the training and test sets.

Figure 3: Different camera views

4.1 Rule-based classification

The simplest method for object classification is a rule-based approach using fixed predefined parameters (as seen in [4,6,15]).

One of the drawbacks of this method is the difficulty of setting the empirical rules that define the expected features for each class, especially when it comes to groups of people or crowds. For this reason, only four parameters (Height, Width, Ratio and AvgSpeed) are used.

Classification is performed by calculating a similarity score between the observed object features and the rules defined for each of the 9 classes previously listed. The Manhattan distance is used to measure the similarity and the Object is assigned to the class with the highest similarity (lowest distance). The results for the recognition of 9 classes clearly show that this algorithm cannot achieve a sufficient accuracy: only a 56% total success rate is achieved.

<table>
<thead>
<tr>
<th>Objects</th>
<th>Bag</th>
<th>Person</th>
<th>Bike</th>
<th>People</th>
<th>Car</th>
<th>Lorry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Person</td>
<td>0</td>
<td>37</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bike</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>People</td>
<td>0</td>
<td>4</td>
<td>14</td>
<td>0</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>Car</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>75</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Lorry</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>17</td>
</tr>
</tbody>
</table>

Total Success Rate: 65%

Table 2: Confusion matrix for the rule-based classifier applied to 6-class recognition

However, by reducing the number of classes and grouping similar ones (bicycles and motorcycles, cars and vans, groups of people and crowds), results slightly improve as shown in Table 3 while the classification of bikes and groups of people remains highly inaccurate. The confusion matrix demonstrates that the rule-based approach can distinguish reliably only pedestrians from vehicles, as already commented in Section 2, and this limits its applicability.

4.2 k-means

k-means [7] is a simple but efficient method to build a classifier. For each parameter in Table 1, a number of clusters equal to the number of classes of interest is created. The centres of these clusters are calculated in a supervised training phase.

Once the training process is completed, classification is performed by measuring the Euclidean distance between the centres of the clusters and the feature vector of the observed object. This will obtain a score for each class. As in the rule-based framework, the object is assigned to the class with the lowest score.

This approach can be improved by assigning specific weights to the different parameters, as in Adaboost networks. These weights can be calculated automatically in a second training step. This is performed by using the training data to classify objects based on one feature at a time. This way more significant weights are assigned to the features that classify the object correctly. Hence, reliable features will influence the classification more than features that do not give much information.

Table 3 shows the results of using k-means on the same data set that was used to evaluate the rule based approach. Initially 9-class recognition was implemented. The algorithm classified correctly 75% of the objects, demonstrating that this method is much more accurate than the rule-based approach. However, results presented in Table 4 for 6-class recognition (for a slight improvement in overall recognition) show that the “Bicycle/Motorcycle” and the “Group of people/Crowd” classes still present high error rates.

<table>
<thead>
<tr>
<th>Objects</th>
<th>Bag</th>
<th>Person</th>
<th>Bike</th>
<th>People</th>
<th>Car</th>
<th>Lorry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Person</td>
<td>1</td>
<td>41</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bike</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>People</td>
<td>0</td>
<td>5</td>
<td>7</td>
<td>18</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Car</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>12</td>
<td>74</td>
<td>5</td>
</tr>
<tr>
<td>Lorry</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>

Total Success Rate: 77%

Table 3: Confusion matrix for k-means classifier applied to 6-class recognition

4.3 Adaboost

Adaboost is a popular method for supervised learning introduced in [3]. An Adaboost network is built of weighted weak classifiers, hence it is able to automatically select the most significant features. Training can be executed offline, but, like k-means, it needs a wide range of samples.

We use the Modest Adaboost algorithm of the GML AdaBoost Matlab Toolbox [13]. The weak classifiers are tree learners made of up to 5 thresholds. For multi-class recognition, a One-Vs-All approach is used [9], as it can be simply implemented without affecting accuracy.

Adaboost is a fast classification algorithm however its computational cost is certainly more significant than k-means. In the testing phase, the tree of solutions has to be explored, multiplying weights and adding values for a maximum number of TN*N*CL cycles; where CL is the number of classes (for multi-class recognition), TN is the total number of weak classifiers (200 in our case) and N is the number of
thresholds representing the weak classifiers. If we assume a maximum number of 64 objects of 9 different classes to be observed in the scene at any time then that yields 200\*5*9*64 = 576000 cycles. This value must be multiplied by the cost of each cycle in processor clocks to estimate the average implementation cost of this approach. As an example, an implementation of the Adaboost classifier on a NXP PNX1302 processor would require an average of 20-30\% of the DSP processing power, hence restricting the real-time implementation of the classifier. This problem could be partially avoided by not performing classification on every frame, however fewer observations yield less confidence in the classification.

Table 4 shows the results of a 6-class Adaboost classifier evaluated on the same dataset as that of the Rule-based and k-means approaches. The Adaboost low performance is mainly due to the poor classification of the two classes “Bike” and “Group of people”. These classes have a wide internal variance and have also affected the performance of the k-means classifier.

In conclusion, the scene independent approach to object classification has poor performance mainly due to the wide variation in the class features caused by scene change. Hence neither k-means nor Adaboost network can find a sufficiently good generalized classifier.

In the next section a scene dependent approach is presented to address the limitations discussed here.

<table>
<thead>
<tr>
<th>Objects</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag</td>
<td>Person</td>
</tr>
<tr>
<td>Bag</td>
<td>Person</td>
</tr>
<tr>
<td>Bag</td>
<td>Person</td>
</tr>
<tr>
<td>Bag</td>
<td>Person</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Classes</th>
<th>Bag</th>
<th>Person</th>
<th>Bike</th>
<th>People</th>
<th>Car</th>
<th>Lorry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Person</td>
<td>1</td>
<td>37</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bike</td>
<td>0</td>
<td>5</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>People</td>
<td>0</td>
<td>5</td>
<td>25</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Car</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>86</td>
<td>1</td>
</tr>
<tr>
<td>Lorry</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>18</td>
</tr>
</tbody>
</table>

**Total Success Rate: 75\%**

Table 4: Confusion matrix for Adaboost classifier applied to 6-class recognition

5 Proposed scene-dependent solution

The results reported in Section 4 demonstrate that a scene-independent solution cannot provide sufficiently accurate results. Despite normalization, features are not sufficiently stable among different scenes due to the variations in view, camera angle, object position and direction. The approach proposed in this section tries to overcome these problems, while keeping the required configuration time to a minimal amount. The idea is to train the system separately for each camera view, so that it can adapt to the different characteristics of the scene.

The training process uses data acquired by the video-surveillance system during a fixed period of time (for example, one week). Samples to be presented to the user, for labelling, are selected through an automatic analysis of the feature vectors. This analysis finds the optimal description of the feature space using a maximum of 32 samples, even when performing multi-class recognition (the value 32 is a trade-off between classification accuracy and configuration brevity). Normalization of the features is used to reduce the size of the training set, as it decreases the intra-class variance. The system then learns the object types by the user identifying such groupings either as classes previously listed, or as new classes (trolley, boat, bus, error, etc.). This provides flexibility and suitability for different applications.

In the classification process, a similarity measure between the observed object instance and the 32 labelled samples is calculated to assign the object to a specific class. To compute the similarity, we use the scene-independent parameters described in Table 1, but also position, horizontal and vertical speed, which depend on the camera view and the specific behaviour of the objects in the scene. Specific weights are applied to each feature. These are calculated only once offline through a procedure similar to the one described in section 0.

The training processes showed that the most significant features are position, height, width, aspect ratio, area, area ratio and absolute, horizontal and vertical speed.

To evaluate the method proposed, we compare its results with the ones achieved by the scene-independent solutions. The learning-based method achieves better results for all classes, in particular for motorbikes and groups of people (Table 5).

<table>
<thead>
<tr>
<th>Class</th>
<th>Number of samples</th>
<th>k-means</th>
<th>Learning-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>17</td>
<td>82%</td>
<td>94%</td>
</tr>
<tr>
<td>Motorbikes</td>
<td>5</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>Small group</td>
<td>4</td>
<td>50%</td>
<td>100%</td>
</tr>
<tr>
<td>Large group</td>
<td>6</td>
<td>50%</td>
<td>83%</td>
</tr>
<tr>
<td>Crowd</td>
<td>5</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>Car</td>
<td>25</td>
<td>88%</td>
<td>100%</td>
</tr>
<tr>
<td>Van</td>
<td>8</td>
<td>62%</td>
<td>75%</td>
</tr>
<tr>
<td>Bus</td>
<td>4</td>
<td>100% (lorry)</td>
<td>100%</td>
</tr>
</tbody>
</table>

**TOTAL 74 70\% 92\%**

Table 5: Objects correctly classified by k-means (algorithm described in Par. 4.2) and our learning-based solution. We consider 74 consistent tracks appearing in a video sequence 35 minutes long; the TOTAL is affected by the number of objects belonging to each class.

The scene-dependent learning-based algorithm can also be trained to recognize erroneous tracks. In Table 6, all tracks are considered, and the class “error” is added. The camera view used for this test presents groups of objects, clumping and strong shadows, so wrong detections are quite frequent. Results demonstrate that the classifier can cope with these challenging conditions, assigning false objects to the class “error”.

Table 6 also shows the results achieved by the Adaboost classification algorithm on the same data set (by using a leave-1-out approach, the training set consists of 364 objects). Adaboost is implemented as described in Section 4.3, but the classifier is trained only on the scene of interest (i.e. Adaboost is used in this case for scene-dependent classification, thus increasing its accuracy). In this way, the two learning-based scene-dependent algorithms can be compared. The results
demonstrate that the proposed method is more accurate than Adaboost though requiring only 32 labelled samples. Furthermore, the results show that a traditional learning algorithm like Adaboost needs a large training set in order to perform. Classes that appear in the scene with very low frequency (for example Bicycles) have not been classified correctly. This means that thousands of labelled samples could be required to obtain sufficient training data for all classes.

<table>
<thead>
<tr>
<th>Class</th>
<th>Number of samples</th>
<th>Scene-dep. Adaboost</th>
<th>Learning-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>47</td>
<td>89%</td>
<td>87%</td>
</tr>
<tr>
<td>Bicycle</td>
<td>3</td>
<td>0%</td>
<td>66%</td>
</tr>
<tr>
<td>Small group</td>
<td>11</td>
<td>55%</td>
<td>82%</td>
</tr>
<tr>
<td>Car</td>
<td>212</td>
<td>90%</td>
<td>92%</td>
</tr>
<tr>
<td>Van</td>
<td>14</td>
<td>43%</td>
<td>64%</td>
</tr>
<tr>
<td>Error</td>
<td>78</td>
<td>86%</td>
<td>83%</td>
</tr>
<tr>
<td>TOTAL</td>
<td>365</td>
<td>85%</td>
<td>88%</td>
</tr>
</tbody>
</table>

Table 6: Objects correctly classified (erroneous detections are considered too). The video sequence lasts 5 minutes and presents cluttering and strong light changes due to sun and clouds. False objects are classified as “Errors”. The scene-dependent Adaboost algorithm is less accurate than our proposed approach, even though it uses a number of samples over 10 times higher.

6 Conclusions

Real-world automatic video-surveillance presents conditions that make the design of an efficient multi-class object classification module a demanding challenge. We discussed previous research and their adaptability to the problem. However, we found out that most algorithms fail to satisfy the requirements of real-time operation, flexibility, cost effectiveness and robustness to varying environmental conditions.

Scene-independent algorithms such as rule-based, k-means and Adaboost show low performance due to variations in camera view and camera angle. On the other hand, most scene-dependent methods require a long training process that is not practical for real-world deployment.

Our learning-based solution overcomes these issues. A supervised training process allows the system to adapt to different camera views and scenarios. Unlike most learning algorithms, the proposed object classification framework can be configured in just 2 minutes. The reported results show that the proposed approach is accurate and flexible, despite its low computational cost.

Acknowledgements

The authors acknowledge financial support from DBERR, Knowledge Transfer Partnership, under grant number KTP006256.

References