Noname manuscript No.
(will be inserted by the editor)

Active Mining of Parallel Video Streams

Samaneh Khoshrou · **Jaime S. Cardoso** · **Luís F. Teixeira**

arXiv:1405.3382v1 [cs.CV] 14 May 2014

Received: date / Accepted: date

Abstract The practicality of a video surveillance system is adversely limited by the amount of queries that can be placed on human resources and their vigilance in response. To transcend this limitation, a major effort under way is to include software that (fully or at least semi) automatically mines video footage, reducing the burden imposed to the system. Herein, we propose a semi-supervised incremental learning framework for evolving visual streams in order to develop a robust and flexible track classification system. Our proposed method learns from consecutive batches by updating an ensemble in each time. It tries to strike a balance between performance of the system and amount of data which needs to be labelled. As no restriction is considered, the system can address many practical problems in an evolving multi-camera scenario, such as concept drift, class evolution and various length of video streams which have not been addressed before. Experiments were performed on synthetic as well as real-world visual data in non-stationary environments, showing high accuracy with fairly little human collaboration.

Keywords Video surveillance · Parallel streams · Active learning

S. Khoshrou, J. S. Cardoso
INESC TEC (formerly INESC Porto)
Rua Doutor Roberto Frias 378, 4200-465 Oporto
Tel.: +351-222094000
E-mail: samaneh.khoshrou@inescporto.pt
jaime.cardoso@inescporto.pt

L. F. Teixeira
Faculdade de Engenharia da Universidade do Porto (FEUP)
Rua Doutor Roberto Frias 378, 4200-465 Oporto
Tel.: +351-225081400
E-mail: lft@fe.up.pt

1 Introduction

Over the last decades, video surveillance began to spread rapidly, specifically targeted at public areas. Recording for hours, days, and possibly years provides massive amount of information coming from an evolving environment in where traditional learning methods fail to reflect evolution taking place [15]. In such environments, the underlying distribution of data changes over time - often referred to as *concept drift* - either due to intrinsic changes (pose change, movement, etc.), or extrinsic changes (lighting condition, dynamic background, complex object background, changes in camera angle, etc.). Thus, models need to be continually updated to represent the latest concepts. The problem is further aggravated when new objects enter the scene - referred to as *class evolution* in machine learning literature - as new models need to be trained for the novel classes.

Figure 1 demonstrates a typical surveillance scenario. Depending on the view angle and the quality of the camera, every surveillance camera covers an area called Field of View (FoV). Often the fields of view are disjoint due to budget constraints, whereas they overlap in some scenarios. When entering the scene, the object will enter the coverage area of at least one of the cameras. The surveillance system will have to track that object from the first moment it was captured by a camera and across all cameras whose fields of view overlap the object's path. In such environments where objects move around and cross the FOV of multiple cameras, it is more than likely to have multiple streams, potentially overlapping in time, recorded at different starting points with various lengths, for the same individual object (Figure 1). However, this type of scenarios is associated with several difficulties. For example, consider the following situation: three differ-

ent persons are detected by a tracking system in a considered span interval. Person A and person B walk side by side while they are captured by camera 1. Person C enters camera 2 field of view and meets person B. In camera 3, person A and person C start walking side by side. Finally, both are again captured by camera 4, after switching their relative positions. In this simple scenario the typical tracking systems are likely to encounter problems. In fact, mutual occlusion may occur if persons B and C cross. Consequently, their identities can be switched. Moreover, accompanying person A with person B or C, group movement (both are identified as a single object) and prolonged occlusion might occur, which might lead to track loss or mistaken identities [42]. Since the cameras are supposed

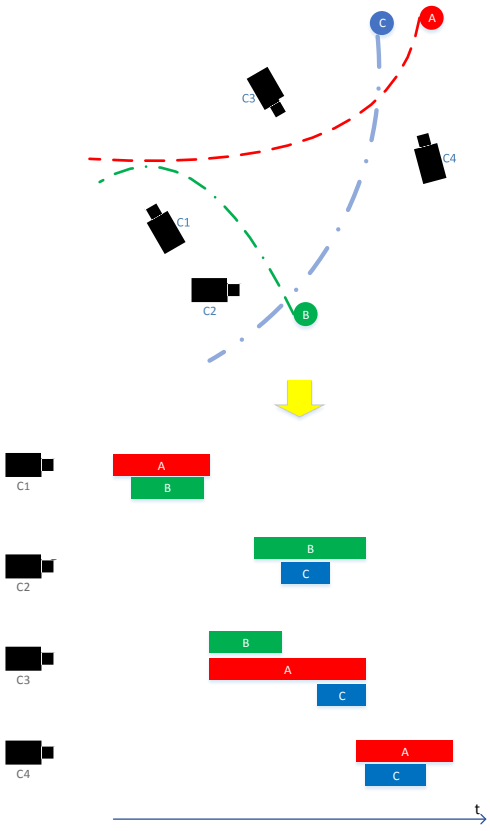


Fig. 1: Typical surveillance scenario

to track all objects in their coverage area, the definition of a global identity for each object is necessary. Multiple appearances of objects captured by the same or by different cameras are identified in the process, allowing also to know the path followed by a given object. The inference of the identity of the objects in the scene is typically addressed with supervised learning methodologies from labelled training data. Obtaining labelled instances, which typically needs human annotation, is expensive, time consuming, and impractical for our scenario. To reduce costs of annotation, semi-supervised learning (SSL) approaches have been extensively explored in limited labelled and usually abundant un-labelled data scenarios [29, 12, 18, 28]; however deploying SSL for evolving visual data in a non-stationary environments (where both concept drift and class evolution are present) is still an unexplored area. Several researchers have shown that the meticulous selection of instances that need to be labelled (mostly addressed in active learning (AL) strategies) could lead to better performance with less effort [4, 34]. In this work we address the need for a more general and systematic view of learning in evolving video streams in a multi-camera surveillance scenario. Considerable body of multi-camera surveillance research assume that adjacent camera view have overlap [9, 26, 19, 45], whereas [20, 38, 21, 33, 31] require non-overlapping views. Herein we put forward a framework to learn continuously from parallel video streams with partially labelled data and that allow us to learn novel knowledge, reinforce existing knowledge that is still relevant, and forget what may no longer be relevant. The method made no assumption of overlapping or non-overlapping view. Hence can be applied in either settings. The framework focuses on the classification of multiple video objects being tracked by a video tracking system. The framework receives directly the tracked sequences outputted by the tracking system and maintains a global object identity common to all the cameras in the system. Thus, a suitable outcome of the framework is a timeline graph (such as the one shown in Figure 1) allocating a stream in each camera for every participant in the system along the time axis for the indicated presence period.

The rest of the paper is organized as follows: next section 1.1 briefly reviews and discusses the limitation of former incremental learning algorithms for visual data. Section 2 provides an overview of our method. Section 3 discusses our experimental methodology. Section 4 presents the results of our method as well as some baseline approaches on a variety of synthetic and real datasets. Conclusions and direction for future work are presented in section 5.

1.1 Literature Review

Much of the recent history on visual data understanding in general and multi-camera surveillance in particular has focused on building robust models applicable in object detection and tracking scenarios [27, 36, 40,

41, 7]. Learning changing video streams over time has received much less attention despite the abundance of applications generating this information. Much of the learning literature is concerned with a stationary environment, where fixed and known number of categories to be recognized and enough resources (labelled data, memory and computational power) are available [32, 46]. To get closer to a practical solution, where obtaining labelled instances is an issue, SSL approaches have been deployed. various SSL methods have been proposed for video annotation [39, 44, 47]. However they have shown promising results for drifting scenarios with pre-determined classes (training data is available for all the classes), but they cannot address class evolution problem. In [3], the person identification task is posed as a graph-based semi-supervised learning problem, where only a few low quality webcam images are labelled. The framework is able to track various objects in limited drifting environments. The classification of objects that have been segmented and tracked without the use of a class-specific tracker, has been addressed with an SSL algorithm in [41]. Given only three hand-labelled training examples of each class, the algorithm can perform comparably to equivalent fully-supervised methods, but it requires full-length tracks (it is therefore an off-line process) generated by a perfect tracker (each stream represents a single object), which would be challenging for real applications, where multiple streams are available simultaneously. Learning from time-changing data has mostly appeared in data mining context and various approaches have been proposed. Ensemble-based approaches constitute a widely popular group of these algorithms to handle concept drift [1, 25] and in some recent works class evolution [16], as well. Learn++.NSE [16] is one of the latest ensemble-based classification methods in literature, that generates a classifier using each batch of training data and applies a dynamic weighting strategy to define the share of each ensemble in the overall decision. As success is heavily dependent on labelled data, this method would not be applicable in wild scenarios. Masud in [30] proposed an online clustering algorithm for single stream that employs an active strategy in order to minimize oracle collaboration.

Although a considerable body of research has emerged from stream mining, learning from multiple streams in wild environments, that views whole or segments of a stream as a unique element to cluster (or classify), is a less explored area. The methods that have been proposed [22, 5, 35, 11, 10], require equal length streams coming from a fixed number of sources. Thus, they would fail to leverage information from time-varying video tracks. An effective and appropriate algorithm to fit in our scenario is required to: a) learn from multiple streams; b) mine streams with various length and starting points (uneven streams); c) handle the concept drift; d) accommodate new classes; e) deal with partially labelled or unlabelled data; f) be of limited complexity; g) handle multi-dimensional data.

Method	Parallel Streams	Uneven Streams	Concept Drift	Class Evolution	Learning	Complexity	Data
[32, 46]	×	×	√	×	SL	Constrained	MD
[1]	×	×	√	×	SL	Unconstrained	MD
[16]	×	×	√	√	SL	Unconstrained	MD
[30]	×	×	√	√	SSL	Constrained	MD
[25]	×	×	√	×	Clustering	Unconstrained	MD
[5, 35, 11, 10]	√	×	√	√	Clustering	Constrained	1D

Table 1: Assessment of learning methods. √ and × denote being fit and inappropriate for the purpose, respectively. "*MD*" and "*1D*" denote multi-dimensional and one-dimensional data. "SL" and "SSL" indicate Supervised Learning and Semi-Supervised Learning.

We wrap up our review in Table 1, presenting a qualitative look at the extent to which the reviewed methods fulfil the requirements for deploying our scenario. To the best of our knowledge, none of the methods have addressed the problem of learning from multiple streams of visual data. In the next section we discuss our proposed algorithm for stream classification.

2 Never Ending Visual Information Learning

In this section we present our Never Ending Visual Information Learning (NEVIL) framework. NEVIL is designed for non-stationary data environments in which no labelled data is available but the learning algorithm is able to interactively query the user to obtain the desired outputs at carefully chosen data points. The NEVIL algorithm is an ensemble of classifiers that are incrementally trained (with no access to previous data) on incoming batches of data, and combined with a form of weighted majority voting.

2.1 Algorithm Description

A high-level sketch of the proposed method is shown in Figure 2. A typical tracking algorithm analyses sequential video frames and outputs the movement of targets between the frames, generating multiple streams of visual data. Environmental challenges such as varying illumination, lack of contrast, bad positioning of acquisition devices, blurring caused by motion as well as occlusion make data often noisy and/or partially missing. We address these challenges by a batch divisive strategy, as learning from a data batch may reduce the noise and fill the gaps caused by miss-tracking.

The algorithm is provided with a series of data batches $\mathcal{D}_t^{m_i}$, where m_i is the index of the i-th stream present at time slot t, TS_t, (not all streams are necessarily present). Note that a stream corresponds to a track generated by the tracking system and a single camera can yield multiple streams. A single batch aggregates B frames. The starting time of each stream is potentially different from stream to stream but batches are aligned between streams. Inside each frame the data corresponds to some pre-selected object representation (e.g. bag of words, histogram) and is out of the scope of this paper.

The ensemble obtained by all models generated up to the current time slot TS_t is named the composite hypothesis H_{t-1}. With the arrival of the current data batches $\mathcal{D}_t^{m_i}, \quad i = 1 \cdots M$, NEVIL tries to predict the class label for each of the batches in current TS_t based on the probability estimate $p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$, where C_k runs over all the class labels observed so far.

Algorithm 1 NEVIL

Input: $\mathcal{D}_t^{m_i}, i = 1, ..., M$
$W_0 \leftarrow \frac{1}{k}$
$H_0 \leftarrow W_0$
while $\mathcal{D}_t$ is *True* **do**
  Batch label prediction (Section 2.1.1)
  $p(C_k|\mathcal{D}_t^{m_i}) \leftarrow (\mathcal{D}_t^{m_i}, H_{t-1})$
  Batch Confidence Level Estimation (Section 2.1.2)
  $BCL \leftarrow p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$
  Multiclass classifier design (Section 2.1.3)
  $h_t \leftarrow \mathcal{D}_t$
  Composite model structure and update (Section 2.1.4)
  $H_t \leftarrow (h_t, H_{t-1}, W_t)$
end while

This kind of on-line learning approach addressed in this work can suffer if labelling errors accumulate, which is inevitable. Unrelated objects will sooner or later be assigned the same label or different labels will be assigned to different views of the same object. To help mitigate this issue, we allow the system to interact with a human, to help it stay on track.

Algorithm 1 outlines our approach. Initially, the composite model is initialized to yield the same probability to every possible class (uniform prior). When the batches $\mathcal{D}_1^{m_t}$ in time slot t become available, NEVIL starts with computing the probabilities $p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$ for each batch $\mathcal{D}_t^{m_i}$ in the time slot. Once $p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$ is obtained, a batch confidence label (BCL) is estimated; if BCL is high enough (above a prespecified threshold), the predicted label

$$\arg\max_{C_k} p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$$

is accepted as correct, otherwise the user is requested to label the data batch. The labelled batches (either automatically or manually) are used to generate a new multiclass classifier that is integrated in the composite model, yielding H_t.

Four tasks need now to be detailed: a) the batch label prediction (by the composite model); b) the BCL estimation; c) the multiclass classifier design in current time slot; d) the composite model structure and update.

2.1.1 Batch Label Prediction

A batch $\mathcal{D}_t^{m_t}$ is a temporal sequence of frames $\mathcal{D}_{t,f}^{m_t}$, where f runs over 1 to the batch size B. The composite model, H_{t-1}, can be used to predict directly $p(C_k|\mathcal{D}_{t,f}^{m_i}, H_{t-1})$ but not $p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$. The batch (multiframe) Bayesian

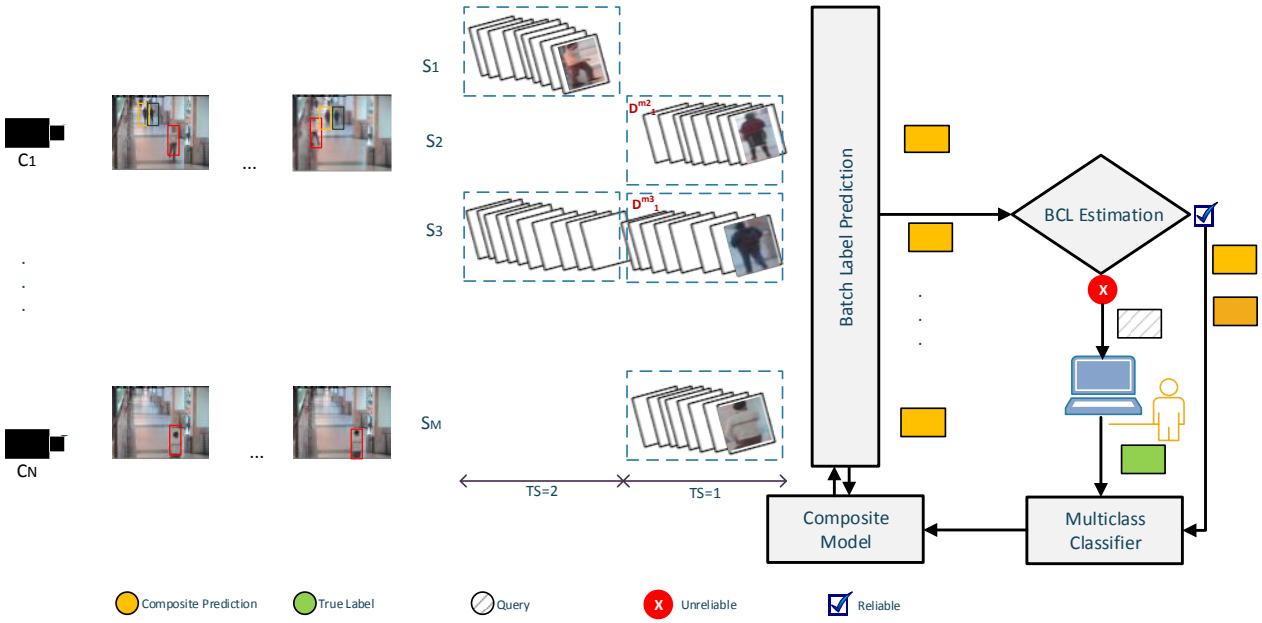


Fig. 2: NEVIL High-level Overview

inference requires conditional independence

$$p(\mathcal{D}_t^{m_i}|C_k,H_{t-1}) = p(\mathcal{D}_{t,1}^{m_i},\cdots,\mathcal{D}_{t,B}^{m_i}|C_k,H_{t-1}) = p(\mathcal{D}_{t,1}^{m_i}|C_k,H_{t-1})\cdots p(\mathcal{D}_{t,B}^{m_i}|C_k,H_{t-1}) = \prod_{j=1}^{B} p(\mathcal{D}_{t,j}^{m_i}|C_k,H_{t-1})$$

From there, and assuming equal prior probabilities, it is trivial to conclude that

$$p(C_k|\mathcal{D}_t^{m_i},H_{t-1}) = Z\prod_{j=1}^{B} p(C_k|\mathcal{D}_{t,j}^{m_i},H_{t-1}), \quad (1)$$

where Z is a normalization constant. In practice, products of many small probabilities can lead to numerical underflow problems, and so it is convenient to work with the logarithm of the distribution. The logarithm is a monotonic function, so that if $p(C_k|\mathcal{D}_t^{m_i},H_{t-1}) > p(C_\ell|\mathcal{D}_t^{m_i},H_{t-1})$ then

$$\log p(C_k|\mathcal{D}_t^{m_i},H_{t-1}) > \log p(C_\ell|\mathcal{D}_t^{m_i},H_{t-1}).$$

Then we can rewrite the decision as choosing the class that maximizes

$$\log p(C_k|\mathcal{D}_t^{m_i},H_{t-1}) = \log Z + \sum_{j=1}^{B} \log p(C_k|\mathcal{D}_{t,j}^{m_i},H_{t-1}) \quad (2)$$

The batch label prediction can also be analysed as a problem of combining information from multiple (B) classification decisions. Considering that, per frame, the composite model produces approximations to the a posteriori probabilities of each class, different combination rules can be considered to build the batch prediction from the individual frame predictions [2, 24]. While Equation (1) turns out to be the product rule (or geometric mean), the sum rule (or arithmetic mean) is also often preferred:

$$p(C_k|\mathcal{D}_t^{m_i},H_{t-1}) = Z\sum_{j=1}^{B} p(C_k|\mathcal{D}_{t,j}^{m_i},H_{t-1}) \quad (3)$$

In fact some authors have shown that the arithmetic mean outperforms the geometric mean in the presence of strong noise [2, 24]. Experimentally, we will compare both options.

2.1.2 The Batch Confidence Level Estimation

Having predicted a class label for a data batch, one needs to decide if the automatic prediction is reliable and accepted or rather a manual labelling be requested.

Various criteria have been introduced as uncertainty measures in literature for a probabilistic framework [37]. Perhaps the simplest and most commonly used criterion relies on the probability of the most confident class, defining the confidence level as

$$\max_{C_k} p(C_k|\mathcal{D}_t^{m_i},H_{t-1}). \quad (4)$$

However, this criterion only considers information about the most probable label. Thus, it effectively "throws away" information about the remaining label distribution [37].

To correct for this, an option is to adopt a margin confidence measure based on the first and second most probable class labels under the model:

$$p(C^*|\mathcal{D}_t^{m_i}, H_{t-1}) - p(C_*|\mathcal{D}_t^{m_i}, H_{t-1}), \tag{5}$$

where C^* and C_* are the first and second most probable class labels, respectively. Intuitively, batches with large margins are easy, since the classifier has little doubt in differentiating between the two most likely class labels. Batches with small margins are more ambiguous, thus knowing the true label would help the model discriminate more effectively between them [37].

Note that while the estimation of the wining class for batch label prediction requires only the comparison of the relative values as given by (1), (2) or (3), both approaches (4) and (5) for the confidence level require the exact computation of the a posteriori probabilities of the classes. This involves computing the normalizing constant associated with (1) or (3), which is specially unstable for (1).

We therefore put forward variants of the two previous measures. As an alternative to the margin confidence measure (5), we base the confidence level on the *ratio* of the first and second most probable class labels:

$$BCL = p(C^*|\mathcal{D}_t^{m_i}, H_{t-1}) / p(C_*|\mathcal{D}_t^{m_i}, H_{t-1}), \tag{6}$$

which can be directly applied for the sum rule or modified to $\log p(C^*|\mathcal{D}_t^{m_i}, H_{t-1}) - \log p(C_*|\mathcal{D}_t^{m_i}, H_{t-1})$ for the product rule. Either way, we eliminate the issue with the normalization constant.

To come up with an alternative to the most confident class measure, we write the decision as

$$\max p_k = \frac{\prod_{j=1}^{B} p_{k,j}}{\sum_k \prod_{j=1}^{B} p_{k,j}} \gtrless T, \tag{7}$$

where we introduced the following simplifications in notation: $p_k = p(C_k|\mathcal{D}_t^{m_i}, H_{t-1})$ and $p_{k,j} = p(C_k|\mathcal{D}_{t,j}^{m_i}, H_{t-1})$. The comparison in Eq. (7) can be rewritten as

$$(1-T)\prod_{j=1}^{B} p_{k^*,j} \gtrless T \sum_{k,k\neq k^*} \prod_{j=1}^{B} p_{k,j}, \tag{8}$$

where $k^* = \arg\max_k p_k$. Since we cannot work directly with the log of (8) due to the sum in the denominator, we introduce the simplification of binarizing the classification in each frame, defining $\bar{p}_{k^*,j} = \sum_{k,k\neq k^*} p_{k,j} = 1 - p_{k^*,j}$.

Accepting the strong assumption of independence for the aggregated class, then

$$\bar{p}_{k^*} = \prod_{j=1}^{B} \bar{p}_{k^*,j}.$$

This ends up in exchanging the order of the sum and product in the right hand side of (8), which can now be rewritten as

$$(1-T)\prod_{j=1}^{B} p_{k^*,j} \gtrless T \prod_{j=1}^{B} \bar{p}_{k^*,j}. \tag{9}$$

Now it is a trivial process to apply the log to obtain a stable decision:

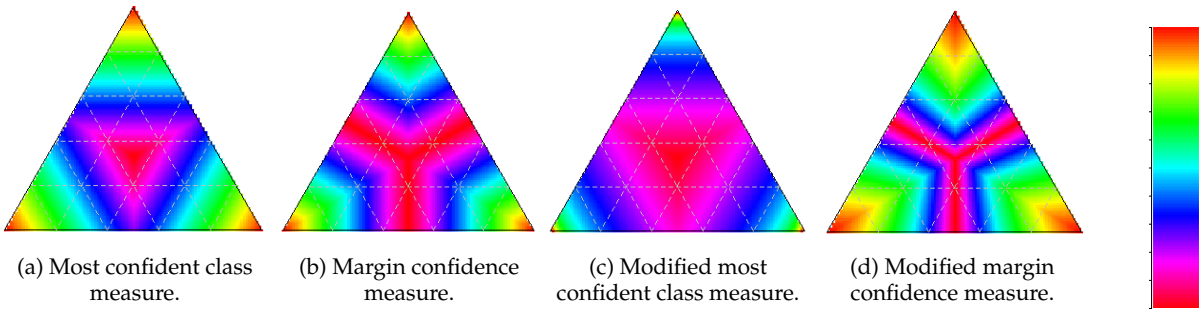
$$\sum_{j=1}^{B} \log p_{k^*,j} \gtrless S + \sum_{j=1}^{B} \log \bar{p}_{k^*,j}, \tag{10}$$

where $S = \log T - \log(1-T)$.

Figure 3 highlights the characteristics of the four confidence measures by a ternary plot (where every corner indicates a class). This plot graphically depicts the ratios of the three variables (herein, occurrence of each class) as positions in an equilateral triangle. The probability of each class is 1 in its corner of the triangle. Moving inside triangle, the percentage of a specific class decreases linearly with increasing distance from the corner till dropping to 0 at the line opposite it. A rainbow-like color pattern shows the informativeness of different composition of three classes. For all methods, the least reliable batch would lie at the center of triangle, where the posterior label distribution is uniform and thus the least certain under the ensemble. Similarly, the most informative batch lies at the corners where one of the classes has the highest possible probability.

2.1.3 Multiclass Classifier

At time slot t, we obtain a new set of batches that are automatically or manually labelled. We assume all the frames belonging to a batch are from the same object (and the underlying tracking system does not mix identities in the time slot period) and therefore the frames inside a batch correspond to observations of the same class. Consider that to the M batches in current time slot correspond $L < M$ labels (some batches can have the same label). We need the design a classifier that can approximate the a posteriori probability function $p(c_k|\mathcal{D}_{t,f}^{m_i})$, which gives the probability of the frame belonging to a given class c_k, given that $\mathcal{D}_{t,f}^{m_i}$ was observed.



(a) Most confident class measure. (b) Margin confidence measure. (c) Modified most confident class measure. (d) Modified margin confidence measure.

Fig. 3: Heatmaps illustrating the behavior of the reliability measures in a three-label classification problem.

A standard way to address this problem is to apply *discriminative approaches* which predict the conditional probability directly. As an alternative, *generative approaches* find the joint distribution $p(\mathcal{D}_{t,f}^{m_i}, c_k)$ and then use Bayes' rule to form the conditional distribution from the generative model. A third option is to find a function $f(\mathcal{D}_{t,f}^{m_i})$, called a discriminant function, which maps each input $\mathcal{D}_{t,f}^{m_i}$ directly onto a class label. In this case, and although probabilities play no role in the design of the discriminant function, it is still possible to get estimated for the conditional probabilities [6]. Each approach has its relative merits and we evaluate experimentally instantiations of each.

One of the challenges we need to handle in a practical scenario is when in a time slot all the batches have the same label (automatically or manually assigned). In these TSs the training of a multiclass classifier is not possible. We resort to one-class classifiers for these time slots, also known as unary classification, to distinguish the single class present in the training set (the batches in the time slot) from all other possible classes [23].

2.1.4 The Composite Model Structure and Update

The composite model H_t in the NEVIL framework is an ensemble of classifiers h_t that are incrementally trained (with no access to previous data) on incoming time slots of data as described previously. The individual models h_t are combined using a weighted majority voting, where the weights are dynamically updated with respect to the classifiers' time of design.

The prediction outputted by the composite model H_t for a given frame $\mathcal{D}_{t,f}^{m_i}$ is

$$p(C_k|\mathcal{D}_{t,f}^{m_i}, H_t) = \sum_{\ell=1}^{t} W_\ell^t h_\ell(C_K|\mathcal{D}_{t,f}^{m_i}),$$

where $h_\ell(.)$ is the multiclass classifier trained at TS ℓ, W_ℓ^t is the weight assigned to classifier ℓ, adjusted for time t.

The weights are updated and normalised at each time slot and chosen to give more credit to more recent knowledge. The weights are chosen from a geometric series $\frac{1}{p^t}, ..., \frac{1}{p^2}, \frac{1}{p}$, normalised by the sum of the series to provide proper probability estimates:

$$W_\ell^t = \frac{\frac{1}{p^{(t-\ell+1)}}}{\sum_{j=1}^{t} \frac{1}{p^j}}$$

3 Experimental Methodology

A series of experiments were conducted to explore the capabilities of the proposed framework. In order to study the behaviour of the system facing various conditions, we generated multiple synthetic streams that were organized in different scenarios. We also tested the NEVIL framework with real video data (see section 3.1).

3.1 Datasets

In order to explore the properties of the proposed framework, we evaluated it on multiple datasets covering various possible scenarios in a multi-camera surveillance system.

We conducted our experiments on synthetic as well as real datasets. The synthetic dataset is generated in the form of (X, y), where X is a 2-dimensional feature vector, drawn from a Gaussian distribution $\mathrm{N}(\mu_X, \delta_X)$, and y is the class label.

Since in real applications visual data may suffer from both gradual and abrupt drift, we tried to simulate both situations in our streams by changing μ_X and δ_X in the parametric equations; Table 2 presents these parametric equations. In this experiment, we generated 7 classes $(C1, C2, ..., C7)$; for some $(C5, C6)$ data changes gradually while others also experience one $(C1, C4, C7)$, or three $(C2, C3)$ dramatic drifts. This process is similar to the one used in [16].

The dataset was organized in 4 different scenarios with different levels of complexity, including streams with gradual drift, sudden drift, re-appearance of objects and non-stationary environments where we have abrupt class and concept drift. Each scenario includes:

- *Scenario* I: gradually drifting streams of 5 classes.
- *Scenario* II: streams with abrupt drifts of 5 classes.
- *Scenario* III : re-appearance of objects.
- *Scenario* IV: a non-stationary environment with class evolution as well as concept drift.

These scenarios are depicted in Fig. 4. Besides synthetic datasets, we run our experiments on a number of CAVIAR video clips [13] including: OneLeave ShopReenter1, Enter ExitCrossingPaths1, OneShopOneWait1, OneStop Enter2 and WalkBy Shop1front. Due to the presence of different perspectives of the same person, streams are drifting in time (see Fig. 5). These sequences present challenging situations with cluttered scenes, high rates of occlusion, different illumination conditions as well as different scales of the person being captured. We employ an automatic tracking approach [43] to track objects in the scene and generate streams of bounding boxes, which define the tracked objects' positions. As the method may fail to perfectly track the targets, a stream often includes frames of distinct objects. An hierarchical bag-of-visterms method is applied to represent the tracked objects, resulting in a descriptor vector of size 11110 for each frame (refer to [42] for additional details). In order to avoid the curse of dimensionality that system may suffer from, Principle Component Analysis (PCA) is applied to the full set of descriptor features as a pre-processing step. Hence, the number of features in each stream is reduced to 85 dimensions. As an explanatory sample, figure 6 depicts the streams in the EnterExitCrossingPaths1 scenario.

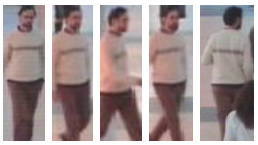


Fig. 5: An example of diversity in appearance

3.2 Instantiation of Classifiers

In Section 2.1.3, we identified three approaches that have been applied in the literature to obtain the posterior probability. A set of experiments were conducted in order to study the behaviour of our framework employing instances of each option. We chose the following methods: *Gaussian Mixture Models (GMM)* and *Naive Bayes* as *generative approaches*, *Support Vector Machines (SVM)* [8] as one of the most popular *discriminant function* and *logistic regression* [17] as a member of *discriminative approaches* family.

Designing a classifier for time slots where batches constitute different labels is quite straightforward. The challenging situation arises when we need to do unary classification. As we employed various approaches with specific characteristics, different strategies are proposed to handle this situation.

Single-class SVM classifies each frame as completely similar or different from given class, whereas generative approaches (GMM and Naive Bayes) provide the probabilistic estimation.

To the extent of our knowledge, using logistic regression in unary problems is an unexplored topic; existing methods need data generated by at least two classes in order to make the prediction. Therefore, we keep the batches from the last multi-class time slot and combine them with the uni-class time slot to build the training set.

3.3 Evaluation Criteria

Active learning aims to achieve high accuracy using as little annotation effort as possible. Thus, a trade-off between accuracy and proportion of labelled data can be considered as one of the most informative measures. Let N denote the total number of batches, MC refer to misclassified batches, then the accuracy of the system

Table 2: Parametric Equations for classes of MS dataset

Drift Rate	C1				C2				C3				C4			
	μ_x	μ_y	δ_x	δ_y	μ_x	μ_y	δ_x	δ_y	μ_x	μ_y	δ_x	δ_y	μ_x	μ_y	δ_x	δ_y
$0 < r < 0.25$	2	5	$0.5r$	$0.5+2r$	$5-5r$	8	$3-10r$	1	$5-5r$	2	$0.5+10r$	0.5	8	$8+15r$	0.5	0.5
$0.25 < r < 0.5$	–	–	–	–	15	$-1+5r$	1	$2+3r$	$1-4r$	2	0.5	$3-4r$	$5-5r$	13	$0.25+4r$	$0.5+4r$
$0.5 < r < 0.75$	10	$-10+5r$	1	$2-r$	17	$2+5r$	0.25	0.15	-1	$-2-4r$	0.25	0.15	–	–	–	–
$0.75 < r < 1$	–	–	–	–	20	$4r$	1	2	-7	$-5-r$	$7+4r$	$1+4r$	–	–	–	–
Drift Rate	C5				C6				C7							
	μ_x	μ_y	δ_x	δ_y	μ_x	μ_y	δ_x	δ_y	μ_x	μ_y	δ_x	δ_y				
$0 < r < 0.25$	12	15	2	$2+2r$	-15	$-5+15r$	1	$2+3r$	10	$5r$	0.5	$2+3r$				
$0.75 < r < 1$		–	–	–	–	–	–	–	$--10$	$-1+5r$	r	$2+3r$				

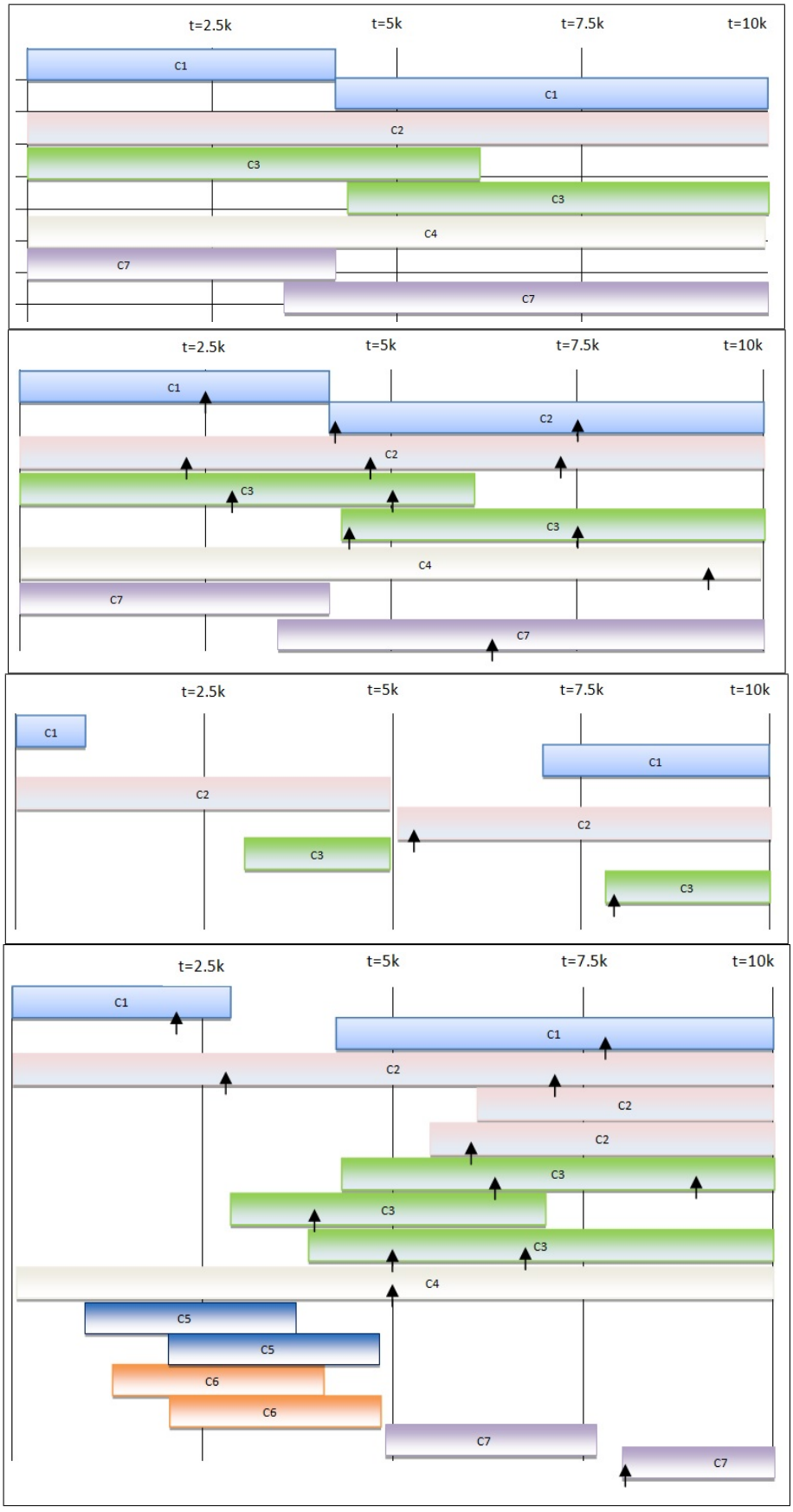


Fig. 4: Scenarios in MS dataset. The sign ↑ denotes the occurrence of an abrupt drift in the nature of data.

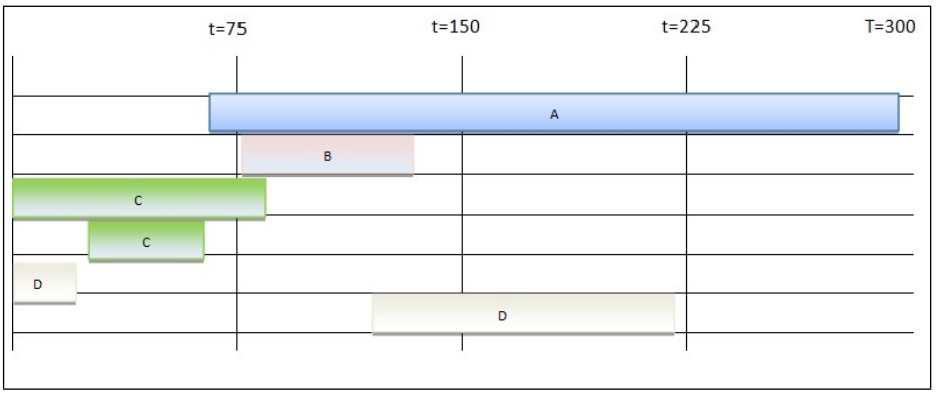


Fig. 6: The EnterExitCrossingPaths1 scenario in the CAVIAR dataset.

in a given time slot is formulated as:

$$Accuracy = \frac{N - \#MC}{N} \tag{11}$$

The total accuracy of a system over a period of time is derived from the mean accuracy of all the time slots.

Assume MLB and TB denote the manually labelled batches and all the batches available during a period (includes one or more time slots), respectively.
The *Annotation Effort* is formulated as:

$$Annotation\ effort = \frac{\#MLB}{\#TB} \tag{12}$$

One expects that the accuracy increases with the increase of annotation effort.

3.4 Baseline Methods

To the best of our knowledge, there is no method that mines multi-dimensional parallel streams in such a non-stationary environment, where the number and length of streams vary greatly. Therefore, we compare our framework with three baseline approaches:

- Passive Learning: The first half of all the batches are submitted to the oracle for labelling. Once the labelled set is obtained, a classifier is trained and applied to classify the other half of stream. This method is far from a real online active learning strategy, as it needs complete data available. For datasets in which there is no dramatic distribution evolution between first and second half, we expect that it provides an upper bound to be compared with our method.
- Even/Odd Learning: As an on-line baseline, for a given stream, batches are marked alternately with odd and even integers, where odd batches are kept in a buffer with their true labels. At each time slot, a model is re-trained using the buffer. We then use this model to classify even batches. Therefore, we may partly follow the distribution changes in this setting leading to better performance than Passive Learning. However, we need to keep all the odd batches, which is far from a practical solution in an on-line scenario.
- Unwise active learning: We use an unwise version of the original framework as a baseline, where all the batches occurred before initiation time (t_{int}) would be annotated. For $t > t_{int}$, NEVIL computes the probabilities of known classes. Once $p(C_k|\mathcal{D}_t^{m_i}, h_{t_{int}})$ are obtained, a batch confidence label (BCL) is estimated; if BCL is high (above a pre-defined threshold), the predicted label

$$\arg\max_{C_k} p(C_k|\mathcal{D}_t^{m_i}, h_{t_{int}})$$

is accepted as correct label of the batch, otherwise the user is requested to label the batch. The method is summarized in Algorithm 2. Despite meticulous selection of queries, as the model is not updated, the algorithm may establish a lower bound the level of performance that can be expected in an evaluation.

Algorithm 2 Unwise active learning

Input: $\mathcal{D}_t^{m_i}, i = 1, ..., M$
$h \leftarrow empty$
while $\mathcal{D}_t$ is *True* **do**
  if $t > t_{int}$ **then**
    Batch label prediction
    $p(C_k|\mathcal{D}_t^{m_i}) \leftarrow (\mathcal{D}_t^{m_i}, h_{int})$
    Batch Confidence Level Estimation
    $BCL \leftarrow p(C_k|\mathcal{D}_t^{m_i}, h_{int})$
  else
    Multiclass classifier design
    $h_{int} \leftarrow \mathcal{D}_t$
  end if
end while

4 Results

Firstly, multiple tests were run to determine the optimal batch size for each dataset to be explored. Batch size was varied between 1% to 50% of the size of the longest stream available in each dataset. Experiments were repeated for 50 equally spaced values in that range. The optimal batch size varies and is influenced by the characteristics of the streams present in each dataset. Optimal batch sizes have been observed to range between 30 and 35 for real video streams and between 200 and 300 for synthetic sequences.

Table 3 provides a summary of the performance of *Passive Learning* and *Even/Odd Learning* using various classifiers on all datasets mentioned in Section 3.1. Since different classifiers provide varying performances on different datasets, the need for a procedure that carefully assesses algorithms seems inevitable. We applied Friedman test [14] that provides a non-parametric rank based statistical significance test. This test is similar to parametric repeated measures ANOVA, which tests if there is a significant difference between the rank of different treatments across multiple attempts. When the test runs over all the datasets shows that null hypothesis is verified which means that type of the classifier has no significant effect on the overall performance of baseline method in real applications. However, the test shows that Logistic Regression has yielded weak results for synthetic data in both learning methods. When we perceive the superior learners based on the mean rank for various scenarios, generative approaches perform fairly better in the synthetic datasets, while discriminative methods win for real video clips. Since the dimension of real data is large, while synthetic data is generated in 2D space, these results also emphasizes the difficulties that generative models face in high-dimensional spaces. As mentioned in 3.4, we expect better or equal results from *Even/Odd Learning* than *Passive Learning* which is the case in all the settings applied discriminative approaches as well as almost all used generative methods. Unexpected behaviour of generative methods when applies on *OneShopOneWait1* dataset can be explained by high bias of these methods when trying to model such complex data.

Figure 7 presents the results of multiple settings on Scenarios I,..., IV. One prominent observation on all these results is that using geometric mean (Prod) to combine information of frames in a given batch and the modified most measure (MMC) to select most informative batches give the best performance.

Figure 8 illustrates the comparative results across multiple classifiers on *Scenarios* I,..., IV from which we can observe that: a) NEVIL achieves more than 90%

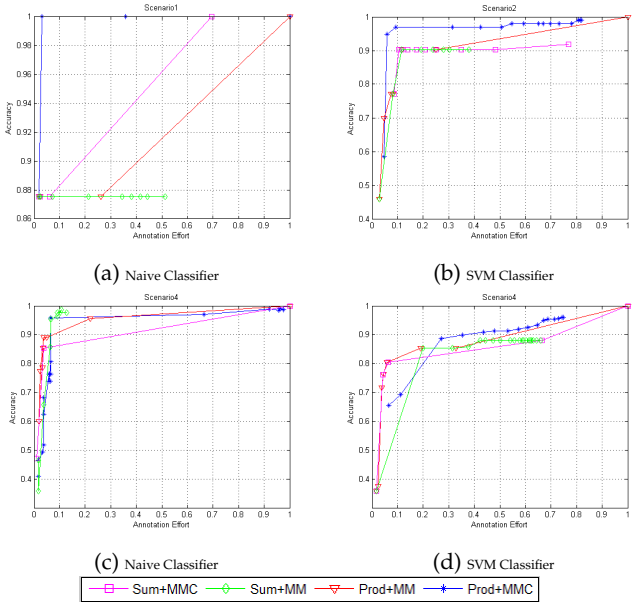


Fig. 7: Multiple configurations tested on the synthetic scenarios. “SUM”, “Prod”, “MMC”, and “MM” indicate sum rule, product rule, modified most confident and modified margin.

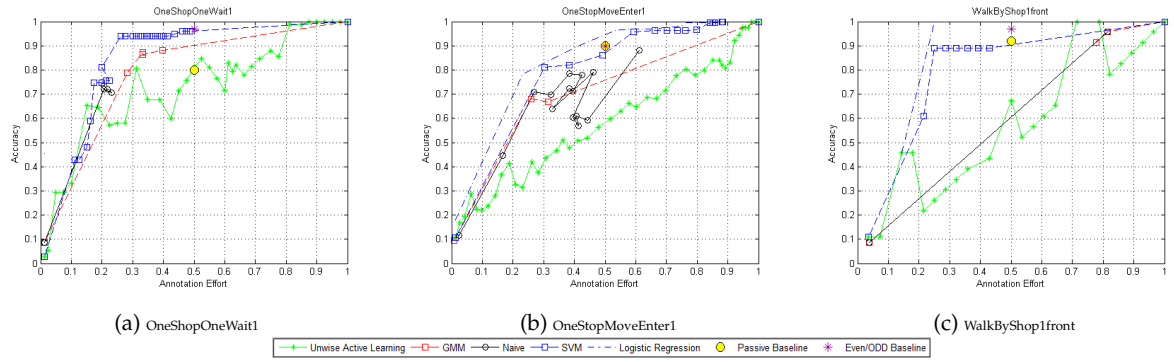
accuracy with less than 15% annotation effort in all the datasets, which obviously outperforms baseline approaches. For all the sets, we reached equal accuracy to *Passive* as well as*Even/Odd* Learning while spending much less human resources. b) Naive classifier gives the best overall performance which emphasise the more flexible nature of generative models than discriminative ones. Needless to say, following the results depicted in Figure 7 we only present the result of winner setting.

Figure 9 shows the comparative results on some CAVIAR sequences with various NEVIL configurations. We observe that unlike synthetic scenarios, employing arithmetic mean (SUM) as combination method and modified margin (MM) as selection criteria present winner results. The presence of challenging noise in real data explains the different behaviour of the framework.

Figure 10 presents the performance of NEVIL employing various classifiers on multiple CAVIAR sequences. The NEVIL framework achieves over 80% accuracy with less than 25% of labelling and in most cases, that is clearly superior to baseline methods. Contrary to results obtained from synthetic data, Discriminative models outperforms than Generative ones. Higher dimension of video streams (herein, equal to 85) may explain this behaviour. Generative models are commonly trained using Maximum-Likelihood Estimation (MLE) that especially for high dimensional data, the likelihood can have many local maxima. Thus, finding the global max-

Table 3: Comparison of baseline approaches on multiple datasets

		Accuracy (%)	
Dataset	Multiple Classifier	Passive Learning	Even/odd Learning
ScenarioI	SVM	97.39	97.19
	GMM	79.61	79.45
	Naive bayes	79.61	79.45
	Logistic Regression	18.44	18.21
ScenarioII	SVM	66.32	72.25
	GMM	79.60	79.45
	Naive bayes	79.60	79.45
	Logistic Regression	40.85	35
ScenarioIII	SVM	74.70	76.65
	GMM	78.37	78.02
	Naive bayes	78.37	78.02
	Logistic Regression	62.18	62.64
ScenarioIV	SVM	80.05	78.61
	GMM	81.81	81.87
	Naive bayes	81.81	81.87
	Logistic Regression	45.51	40.65
EnterExitCrossingPaths1	SVM	89.28	93.7
	GMM	66.45	75.16
	Naive bayes	66.45	75.16
	Logistic Regression	80.12	79.86
OneLeaveShopReenter1	SVM	63.74	100
	GMM	64.06	61.49
	Naive bayes	64.06	61.49
	Logistic Regression	92.18	97.86
OneShopOneWait1	SVM	80.24	95.79
	GMM	92.33	52.88
	Naive bayes	92.33	52.88
	Logistic Regression	81.35	93.42
OneStopEnter2	SVM	83.56	98.63
	GMM	76.73	75.28
	Naive bayes	76.73	75.28
	Logistic Regression	81.36	93.02
WalkByShop1front	SVM	92.32	97.58
	GMM	91.50	90.93
	Naive bayes	91.50	90.93
	Logistic Regression	89.04	96.07
OneStopMoveEnter1	SVM	62.47	79.40
	GMM	90.99	90.93
	Naive bayes	90.99	90.93
	Logistic Regression	56.25	73.76



(a) OneShopOneWait1 (b) OneStopMoveEnter1 (c) WalkByShop1front

Fig. 10: Performance using multiple configurations on the CAVIAR sequences.

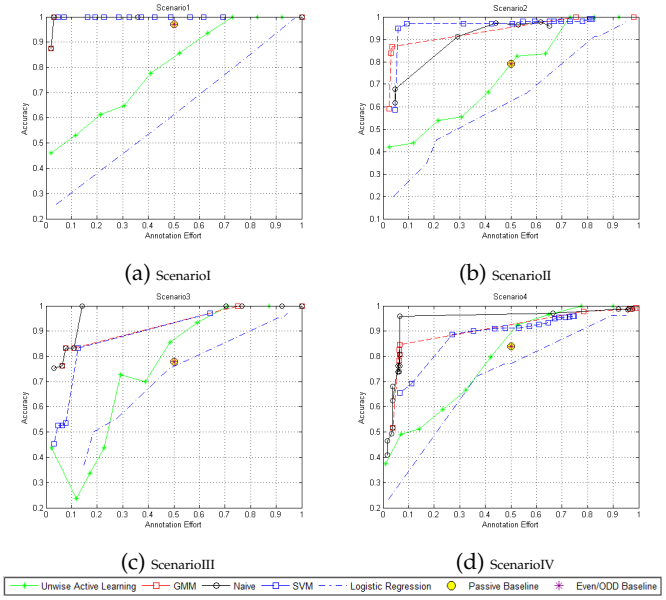


Fig. 8: Multi-class classifier comparison on synthetic scenarios using the best configuration (Prod+MM).

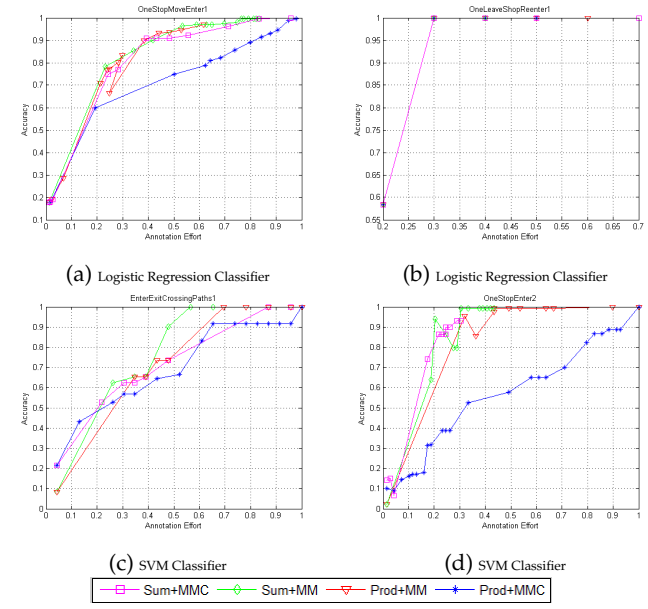


Fig. 9: Multiple configurations tested on the CAVIAR sequences."SUM", "Prod", "MMC", and "MM" indicate sum rule, product rule, modified most confident and modified margin.

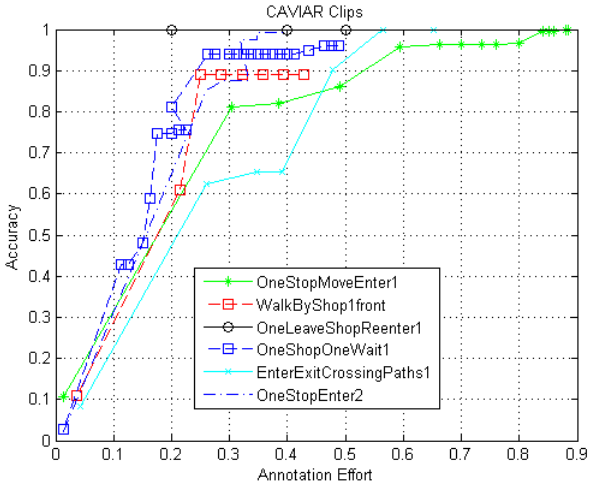


Fig. 11: Performance of NEVIL on multiple CAVIAR sequences. The results were obtained with the SUM+MM configuration and applying SVM as the classifier.

imum affects the performance and renders the approach less practical.

Finally, Figure 11 presents the results obtained across multiple CAVIAR scenarios from the most successful setting, which means SVM, SUM, and MMC as base classifier, combination rule and selection criteria, respectively. Under such setting, NEVIL achieves 80% accuracy with 30% annotation effort for *OneStopMoveEnter1*, the most complex scenario with 42 streams from 14 classes.

5 Conclusions

In this paper, we address the problem of learning from visual streams generated in a multi-camera surveillance scenario. We look at the problem as mining parallel high-dimensional data. Inspired from active learning strategies, in our proposed framework (NEVIL) an oracle provides labelled batches; multiple informativeness measures are used to determine when the oracle is used. As base learners are bottlenecks of any learning pipeline, various groups of classifiers were studied and experimentally evaluated. We ran the experiments on synthetic as well as real datasets.

In synthetic scenarios, where low dimensional clean data is available, applying the geometric mean and the modified most confident measure gives the best and least expensive (in terms of annotation cost) results. However, to get the highest accuracy from noisy visual data we need to apply arithmetic mean for com-

bining information and modified margin to select the most informative batches.

Another question we tried to answer was which classifier to use on a given dataset. In a low dimensional clean data, generative approaches give the best results, however obtaining robust and stable results from high dimensional data is too difficult, as shown by our experiments. The best performance is obtained through discriminative approaches.

While empirical results demonstrated the functionality of the framework, we are currently working the controlling the complexity of the framework which makes it applicable for never-ending scenarios. We would also like to employ special features of video streams generated in a surveillance scenario in order reduce queries as many as possible.

Acknowledgements The authors would like to thank Fundação para a Ciência e a Tecnologia (FCT)-Portugal for financing this work through the grant SFRH/BD/80013/2011.

References

1. Ackermann MR, Lammersen C, Märtens M, Raupach C, Sohler C, Swierkot K (2010) StreamKM++: A clustering algorithms for data streams. In: Journal of Experimental Algorithmics, pp 173–187
2. Alexandre LA, Campilho AC, Kamel M (2001) On combining classifiers using sum and product rules. Pattern Recognition Letters 22(12):1283–1289
3. Balcan M, Blum A, Choi PP, Lafferty J, Pantano B, Rwebangira MR, Zhu X (2005) Person identification in webcam images:an application of semi-supervised learning. In: International Conference on Machine Learning Workshop on Learning from Partially Classified Training Data, pp 1–9
4. Baram Y, El-Yaniv R, Luz K (2004) Online choice of active learning algorithms. Journal of Machine Learning Research 5:255–291
5. Beringer J, Hüllermeier E (2006) Online clustering of parallel data streams. Data Knowledge Engineering 58(2):180–204
6. Bishop CM (2006) Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA
7. Carvalho P, Cardoso JS, Corte-Real L (2012) Filling the gap in quality assessment of video object tracking. Image and Vision Computing 30:630–640
8. Chang CC, Lin CJ (2011) LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2:27:1–27:27
9. hsun Chang T, Gong S (2001) Tracking multiple people with a multi-camera system. In: IEEE Workshop on Multi-Object Tracking
10. Chen L, Zou L, Tu L (2012) A clustering algorithm for multiple data streams based on spectral component similarity. Information Sciences 183(1):35–47
11. Chen Y (2009) Clustering parallel data streams. Data Mining and Knowledge Discovery in Real Life Applications I-Tech Education and Publishing
12. Cong G, Lee WS, Wu H, Liu B (2004) Semi-supervised text classification using partitioned em. In: 11th International Conference on Database Systems for Advanced Applications, pp 482–493
13. project consortium C (2001) Caviar dataset. URL `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`.
14. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7:1–30
15. Dick AR, Brooks MJ (2003) Issues in automated visual surveillance. In: Proc. VIIth Digital Image, pp 195–204
16. Elwell R, Polikar R (2011) Incremental learning of concept drift in nonstationary environments. IEEE Transactions on Neural Networks 22(10):1517–1531
17. Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ (2008) Liblinear: A library for large linear classification. Journal of Machine Learning Research 9:1871–1874
18. Garrette D, Mielens J, Baldridge J (2013) Real-world semi-supervised learning of pos-taggers for low-resource languages. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL-2013), Sofia, Bulgaria
19. Hamid R, Kumar RK, Hodgins JK, Essa IA (2014) A visualization framework for team sports captured using multiple static cameras. Computer Vision and Image Understanding 118:171–183
20. Javed O (2005) Appearance modeling for tracking in multiple non-overlapping cameras. In: In IEEE International Conference on Computer Vision and Pattern Recognition, pp 26–33
21. Javed O, Shah M (2008) Automated Multi-Camera Surveillance: Algorithms and Practice, The International Series in Video Computing, vol 10. Springer
22. Keogh E, Kasetty S (2003) On the need for time series data mining benchmarks: A survey and empirical demonstration. Data Mining Knowledge Discovery 7(4):349–371
23. Khan SS, GMadden M (2010) A survey of recent trends in one class classification. In: Coyle L,

Freyne J (eds) Artificial Intelligence and Cognitive Science, Lecture Notes in Computer Science, vol 6206, Springer Berlin Heidelberg, pp 188–197
24. Kittler J, Hatef M, Duin RPW, Matas J (1998) On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(3):226–239
25. Kolter JZ, Maloof MA (2007) Dynamic weighted majority: An ensemble method for drifting concepts. Journal of Machine Learning Research 8:2755–2790
26. Kuo CH, Huang C, Nevatia R (2010) Inter-camera association of multi-target tracks by online learned appearance affinity models. In: Proceedings of the 11th European Conference on Computer Vision: Part I, Springer-Verlag, Berlin, Heidelberg, ECCV'10, pp 383–396
27. Levin A, Viola PA, Freund Y (2003) Unsupervised improvement of visual detectors using co-training. In: Ninth IEEE International Conference on Computer Vision, pp 626–633
28. Liang P (2005) Semi-supervised learning for natural language. Master's thesis, MIT
29. Masud MM, Gao J, Khan L, Han J, Thuraisingham BM (2010) Classification and novel class detection in data streams with active mining. In: 14th Pacific-Asia Conference Advances in Knowledge Discovery and Data Mining, pp 311–324
30. Masud MM, Gao J, Khan L, Han J, Thuraisingham BM (2011) Classification and novel class detection in concept-drifting data streams under time constraints. IEEE Transactions on Knowledge Data Engineering 23(6):859–874
31. Matei BC, Sawhney HS, Samarasekera S (2011) Vehicle tracking across nonoverlapping cameras using joint kinematic and appearance features. In: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, Washington, DC, USA, CVPR '11, pp 3465–3472
32. Ozawa S, Toh SL, Abe S, Pang S, Kasabov N (2005) Incremental learning of feature space and classifier for face recognition. Neural Networks 18(5-6):575–584
33. Pflugfelder R, Bischof H (2010) Localization and trajectory reconstruction in surveillance cameras with nonoverlapping views. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(4):709–721
34. Raghavan H, Madani O, Jones R, Kaelbling P (2006) Active learning with feedback on both features and instances. Journal of Machine Learning Research 7:1655–1686
35. Rodrigues PP, Gama J, Pedroso JP (2008) Hierarchical clustering of time-series data streams. IEEE Transaction on Knowledge Data Engineering 20(5):615–627
36. Rosenberg C, Hebert M, Schneiderman H (2005) Semi-supervised self-training of object detection models. In: Seventh IEEE Workshop on Applications of Computer Vision, pp 29–36
37. Settles B (2009) Active learning literature survey. Tech. Rep. 1648, University of Wisconsin–Madison
38. Shan Y, Sawhney HS, Kumar R (2005) Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras. In: CVPR (1), pp 894–901
39. Song Y, sheng Hua X, rong Dai L, Wang M (2005) Semi-automatic video annotation based on active learning with multiple complementary predictors. In: In Proceedings of ACM SIGMM International Workshop on Multimedia Information Retrieval, pp 97–104
40. Stalder S, Grabner H, Gool LV (2009) Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In: IEEE 12th International Conference on Computer Vision Workshops, pp 1409–1416
41. Teichman A, Thrun S (2011) Tracking-based semi-supervised learning. In: Proceedings of Robotics: Science and Systems, Los Angeles, CA, USA
42. Teixeira LF, Corte-Real L (2009) Video object matching across multiple independent views using local descriptors and adaptive learning. Pattern Recognition Letters 30(2):157–167
43. Teixeira LF, Carvalho P, Cardoso JS, Corte-Real L (2012) Automatic description of object appearances in a wide-area surveillance scenario. In: 19th IEEE International Conference on Image Processing, pp 1609–1612
44. Wang M, Hua XS, Mei T, Hong R, Qi G, Song Y, Dai LR (2009) Semi-supervised kernel density estimation for video annotation. Comput Vis Image Underst 113(3):384–396
45. Wang X (2013) Intelligent multi-camera video surveillance: A review. Pattern Recognition Letters 34(1):3–19
46. Wu J (2004) An online-optimized incremental learning framework for video semantic classification. In: 12th ACM International Conference on Multimedia, pp 320–323
47. Xu XS, Jiang Y, Xue X, Zhou ZH (2012) Semi-supervised multi-instance multi-label learning for video annotation task. In: Proceedings of the 20th ACM International Conference on Multimedia, ACM, New York, NY, USA, MM '12, pp 737–740