Online Action Recognition via Nonparametric Incremental Learning

Rocco De Rosa
rocco.derosa@unimi.it
Dipartimento di Matematica
Università degli Studi di Milano, Italy

Nicolò Cesa-Bianchi
nicolo.cesa-bianchi@unimi.it
Dipartimento di Informatica
Università degli Studi di Milano, Italy

Ilaria Gori
ilaria.gori@iit.it
iCub Facility
Istituto Italiano di Tecnologia, Italy

Fabio Cuzzolin
fabio.cuzzolin@brookes.ac.uk
Dept. of Comp. and Comm. Tech.
Oxford Brookes University, UK

Abstract

We introduce an online action recognition system that can be combined with any set of frame-by-frame feature descriptors. Our system covers the frame feature space with classifiers whose distribution adapts to the hardness of locally approximating the Bayes optimal classifier. An efficient nearest neighbour search is used to find and combine the local classifiers that are closest to the frames of a new video to be classified. The advantages of our approach are: incremental training, frame by frame real-time prediction, nonparametric predictive modelling, video segmentation for continuous action recognition, no need to trim videos to equal lengths and only one tuning parameter (which, for large datasets, can be safely set to the diameter of the feature space). Experiments on standard benchmarks show that our system is competitive with state-of-the-art non-incremental and incremental baselines.

keywords: action recognition, incremental learning, continuous action recognition, nonparametric model, real time, multivariate time series classification, temporal classification

1 Introduction

Action recognition has been attracting increasing interest in the last decade, due its role in a variety of applications involving the natural interaction between people and electronic devices, patient monitoring systems, and surveillance systems, to cite a few. The main challenge is designing general purpose algorithms that work reasonably well across these many different scenarios. Recently, a number algorithms for gesture, action, and activity recognition were proposed in the literature; we refer the reader to [1] for a rather extensive survey on the topic. Typical approaches to the problem are based on a two-stage procedure: First, discriminative features [48] are extracted from image sequences either locally (frame by frame) or globally, and then collected in a training set. Second, a classifier (e.g., k-NN, SVM, decision tree) is learned from the training set and used to categorize new videos. These methods typically employ a batch learning strategy: the classifier is the outcome of some global optimization process applied to the training set. If new training data become available, the system needs to be re-trained from scratch.

Online and incremental action recognition. Batch learning cannot really be applied to a number of important real-world applications, because of the sheer amount of training data, and the fact that training data is collected in an inherently sequential fashion. Applications in which this constraint holds include: large scale datasets [15], surveillance systems [20, 24, 32], disease recognition [13, 40], human-computer interaction [27, 42], human-robot interaction [12, 16] and robot cooperation [10]. For instance, performing a global optimization training procedure on a large scale dataset might be prohibitive. In other scenarios, such as surveillance systems, incremental learning is needed to quickly process new training examples without having to re-train from scratch. In an incremental setting global video features cannot be used, as the training data are not available as a collection of pre-segmented videos. The use of local features, extracted frame by frame, becomes thus mandatory. Different descriptors of this type exist, such as HOOF [8], 3DHOF [19], HOG [9], PHOG [7], and HON4D [37].

State of the art on incremental learning. Incremental algorithms, which update the learned model sequentially, have been already used in computer vision problems. Examples include face recognition [38], object tracking [23] and detection [45], gait recognition [21], visual tracking [29], and robot cooperation [10]. Nevertheless, such methods generally adapt existing incremental machine learning algorithms designed for "static" data to work with features extracted from videos, ignoring the dynamics of the frame sequence. For instance, [33, 51] apply incremental SVM or incremental discriminant analysis (respectively) to global video features. The authors of [35] manipulate a global descriptor based on snippets [43] to make it more efficient for real-time applications, and then use a recursive Extreme Learning Machine (ELM) for incremental learning.

Proposed methodology. We propose here a general framework for incremental video classification based on the following principles: (i) each video frame is a training example in a local feature space; (ii) incoming training examples are selected to cover the frame feature space with balls whose radius is adjusted according to the distribution of action classes within each ball; (iii) each ball is associated with an estimate of the conditional class probabilities, obtained by collecting statistics around its centre, which is used to make predictions on new unlabeled samples; (iv) the set of balls can be organized in a tree structure [26], allowing logarithmic queries in the number of balls. During training, a new ball is added whenever the input frame example does not belong to the ball whose center is the closest to the frame among the centers in the current set. Otherwise, the ball statistics and its radius are updated. In the prediction phase, the conditional class probability estimates associated with the ball centre nearest to the input frames are used to select the action that maximises the sum of those scores. The method allows us to work *incrementally* at frame level and *in real time*. Our learning method is also nonparametric. That is, the classifier structure is not pre-determined (as for linear classifiers), but it is inferred from the data (as for k-NN). To the best of our knowledge no other approach enjoys all these attractive features.

Application to online action recognition. As it handles videos on a frame-by-frame basis, the method is suitable to tackle the so-called "continuous action recognition" problem [3, 4, 18, 36, 46, 52], in which both training and test videos contain a number of gestures in a sequence. Temporal segmentation is then paramount to cut video streams into single action instances, consistent with the models learnt from the training sequences. We show how our method can be easily combined with the robust temporal segmentation algorithm presented in [12].

Contributions and outline. In summary, (i) we propose a novel algorithm for online nonparametric recognition built upon a recent nonparametric regression method [25]; (ii) we apply our algorithm to incremental action recognition, demonstrating competitive performance on a significant set of benchmarks; (iii) we outline an online recognition method, able to automatically segment and recognise actions in real time. The paper is organized as follows. In Section 3 we introduce our online recognition framework and the training and prediction algorithms. The algorithm is validated on both batch and incremental action recognition in Section 4. In Section 5 we propose an online action segmentation and recognition setup based on our nonparametric approach. Section 6 concludes the paper.

2 Related work

The approach closest to the one proposed here is [41]. There, the authors use an incremental "feature-tree" for indexing local spatio-temporal features extracted from cuboids [11] within labelled videos, and then associate each feature with the video label. In the recognition stage, local features are first detected and extracted. Then, for each feature a query is made on the feature-tree, which returns a set of nearest neighbour features and their corresponding labels. The video is assigned to the label that receives the highest number of votes from its neighbourhood. Although this method appears rather similar to ours, there are considerable differences. First, the authors use an unsupervised partition of the (local) feature space. The cover constructed by our approach, instead, adapts to the local label distribution. This generates partitions which are more discriminative than those produces by classical k-NN in which, additionally, finding the 'true' value of k is generally difficult. Furthermore, their method stores all training data, whereas our algorithm stores a considerably smaller number of ball centres —see Figure 2. Finally, our approach associates with each frame a class score, which is generally more informative than majority voting. For instance, if two actions share some common sub-actions, under [41] the "wrong" action may receive the majority of votes for a number of frames. With our method, on the other hand, those frames will assign a lower score to the true action, but this score may eventually be highly influential in determining the final decision when compared against all the other action scores. The work [41] deals also with simultaneous multiple action recognition, by assigning a label to each cuboid and detecting the parts of the videos where an action of interest is performed. Our method can be also adapted to multiple action recognition by managing individual cuboids as sequences of video frames in their own right.

We did not include the method developed by [41] in our set of baselines, because the computational cost of feature extraction is substantially higher than that of the other methods considered in this work. A more extensive comparison of the time vs. accuracy trade-offs in this context are left to future work.

Note that our approach is only superficially similar to kernel-based classification. This latter technique predicts with a weighted combination of kernels, where the kernel parameter is typically chosen via cross-validation. Our approach, instead, uses only the nearest ball centre to predict, and reduces the ball radius proportionally to the number of mistakes made by the ball centre classifier. As explained in [25], this reduction is key to prove convergence to the Bayes optimal classifier.

3 Learning framework

We consider a model in which the learner is trained on a sequence $(V_1,y_1),(V_2,y_2),\dots$ of labeled videos. Video V_i contains T_i frames $v_1^{(i)},\dots,v_j^{(i)},\dots,v_{T_i}^{(i)}$, where each frame $v_j^{(i)} \in \mathbb{R}^D$ is represented by D features (see Datasets in Section 4 for a description of the features used in this work). The video is annotated with a label y_i denoting an action from a given set $\mathcal{Y} = \{1,\dots,C\}$ of possible actions. The learner's task is to build a classifier, mapping each new video to its correct label. We focus on an *incremental learning* setting, in which the classifier is trained incrementally (via small adjustments to the current model) every time a new labeled video (or labeled frame) is presented to the learner. We adopt a frame based classification approach, but at the same time we take into account the temporal extent of the actions applying a temporal difference encoding to the video frames. For each video V_i and for each frame index $j = 1,\dots,T_i-1$ of V_i, the j-th frame $v_j^{(i)} \in \mathbb{R}^D$ is encoded as $x_j^{(i)} = \left(v_{j+1}^{(i)}, v_{j+1}^{(i)} - v_j^{(i)}\right) \in \mathbb{R}^{2D}$. Hence, each labeled video (V_i,y_i) generates T_i-1 labeled frames $\left(x_j^{(i)},y_i\right)$ for $j = 1,\dots,T_i-1$.

In what follows, we drop the superscripts i and re-index the frames, thus assuming that the learner is fed a sequence $(x_1,y_1),(x_2,y_2),\dots \in \mathbb{R}^{2D} \times \mathcal{Y}$ of labeled frames (i.e., training examples), where each $(x_t,y_t) = \left(x_j^{(i)},y_i\right)$ for some i and $1 \le j < T_i$. Our classification framework is based on a recently developed nonparametric regression method suitable for streaming data [25]; according to this algorithm, the frame feature space is adaptively covered by a set $\mathcal{S}$ of balls, as a function of the distribution of the data points in the feature space and of the empirical class distributions in each ball. Each new data point x_t is classified using the prediction provided by the nearest ball centre $x_{\pi(t)}$ in $\mathcal{S}$ (where $\pi(t)$ denotes the index of the element of $\mathcal{S}$ that is closest to x_t). In the following, we describe the incremental training procedure which defines the set $\mathcal{S}$ and the way ball predictions are computed.

Training. The sequence of observed training examples, obtained via the temporal difference encoding of the incoming videos, is used to build a set $\mathcal{S}$ of balls covering the region of the feature space spanned by the examples. For each ball, an empirical distribution of classes is maintained. The radii of the balls are adjusted to account for mistakes in predicting the class of the training examples. A ball is added only when a new sample does not fall within the nearest existing ball. For each ball center $x_s \in \mathcal{S}$ we keep updated counts $n_s(c)$ of the number of data points x_t of class $c \in \mathcal{Y}$ that at time t belonged to the ball centered on x_s (remember that the ball's radius changes over time). These counts are used to compute Laplace-adjusted class probability estimates for each ball center $x_s \in \mathcal{S}$:

$$p_s(c) = \frac{n_s(c)+1}{n_s+C} \qquad c = 1,\dots,C \tag{1}$$

where $n_s = n_s(1)+\cdots+n_s(C)$.

More specifically, the training algorithm operates as follows —see Algorithm 1. (i) Initially, the set of balls $\mathcal{S}$ is empty. (ii) For each training example x_t, we efficiently compute the nearest neighbour $x_{\pi(t)} \in \mathcal{S}$, according to a given metric ρ. Note that there exist data structures that allow an efficient management of the set $\mathcal{S}$ of ball centers. For example, [26] embeds $\mathcal{S}$ in a tree where nearest neighbour queries and updates can be performed in time $\mathcal{O}(\ln|\mathcal{S}|)$ —see also [25]. (iii) If x_t does not belong to the nearest neighbour $x_{\pi(t)} \in \mathcal{S}$, a new ball of radius $\varepsilon_t = R > 0$ centered on it is created and added to $\mathcal{S}$; its label y_t is used to initialize the empirical class distribution for the new ball via (1). (iv) Otherwise, the label y_t

of the current example is used to update the mistake counts $m_{\pi(t)}$ of the closest ball. The ball center $x_{\pi(t)}$ makes a mistake on (x_t, y_t) if and only if $y_t \neq \operatorname*{argmax}_{c \in \mathcal{Y}} p_{\pi(t)}(c)$. (v) Whenever the ball center $x_{\pi(t)} \in \mathcal{S}$ makes a wrong prediction, the radius $\varepsilon_{\pi(t)}$ is shrunk by an amount that depends inversely on an estimates of the "intrinsic dimension"[1] d of the stream data, as a function of the number of prediction mistakes made so far: $\varepsilon_{\pi(t)} = R m_{\pi(t)}^{-1/(d+2)}$. (vi) Finally, the class probability estimates $p_{\pi(t)}(c)$ for the ball center $x_{\pi(t)}$ are updated via (1). Preliminary knowledge of the full set of action classes $\mathcal{Y}$ is not needed: our incremental learning approach can add new labels to $\mathcal{Y}$ as soon as they first appear in the video sequence.

Algorithm 1 ABACOC (Adaptive Ball Cover for Classification)

```
Input: Initial radius R > 0, metric ρ
Initialize set of ball centers S = ∅ and set of labels Y = ∅
for i = 1, 2, ... do
    Get labeled video (V_i, y_i) and create frame sequence (x_1, y_i), ..., (x_{T_i−1}, y_i) using temporal difference encoding
    if y_i ∉ Y then
        Set Y = Y ∪ {y_i}
    end if
    for t = 1, ..., T_i − 1 do
        if S ≡ ∅ then
            S = {x_t}, set radius ε_t = R, and use y_i to initialize estimates p_t via (1)
        else
            Let x_s ∈ S be the nearest neighbour of x_t in S
            if ρ(x_s, x_t) ≤ ε_s (x_t belongs to current ball centered on x_s) then
                if y_i ≠ argmax_{c∈Y} p_s(c) then
                    Set m_s = m_s + 1 and update radius via ε_s = R m_s^{−1/(2+d)}
                end if
                Use y_i to update estimates p_s via (1)
            else
                S = S ∪ {x_t}, set radius ε_t = R, and use y_i to initialize estimates p_t
            end if
        end if
    end for
end for
```

Prediction. In the prediction phase, we proceed similarly: the video V_i is used to generate the $T_i - 1$ frames $x_1, \ldots, x_{T_i - 1}$ via temporal difference encoding. For each x_t the nearest neighbour $x_{\pi(t)} \in \mathcal{S}$ is computed. Then, the label of the test video V_i is predicted using the following maximum likelihood estimate, which integrates over all the frames of the video:

$$\widehat{y}_i = \operatorname*{argmax}_{c \in \mathcal{Y}} \sum_{t=1}^{T_i - 1} \ln p_{\pi(t)}(c) \; . \tag{2}$$

In the regression setting proposed in [25], all ball radii ε_s shrink uniformly with time t at

[1] This roughly corresponds to the smallest number of dimensions in which the stream can be embedded without significantly increasing the Bayes error.

rate $t^{-1/(d+2)}$, where d is the unknown intrinsic dimension of the space (which can be much smaller than the ambient dimension D, i.e., the number of features in our case). Under Lipschitz assumptions on the true regression function, the method in [25] was shown to be consistent[2], with a nearly optimal nonparametric convergence rate. In this work, besides replacing the regression estimate associated with each ball center with a maximum likelihood estimate (more appropriate for classification), we let the radius of each ball shrink at a rate dependent on the number of mistakes made by the estimator associated with the ball center. This leads to a model where many small balls are used to cover only regions with high mistake rates (where the Bayes optimal classifier is supposedly more complex), whereas low mistake rate regions get covered with a few large balls —see Figure 1. Establishing sufficient conditions on the stream under which the consistency of our classification approach is guaranteed is currently work in progress.

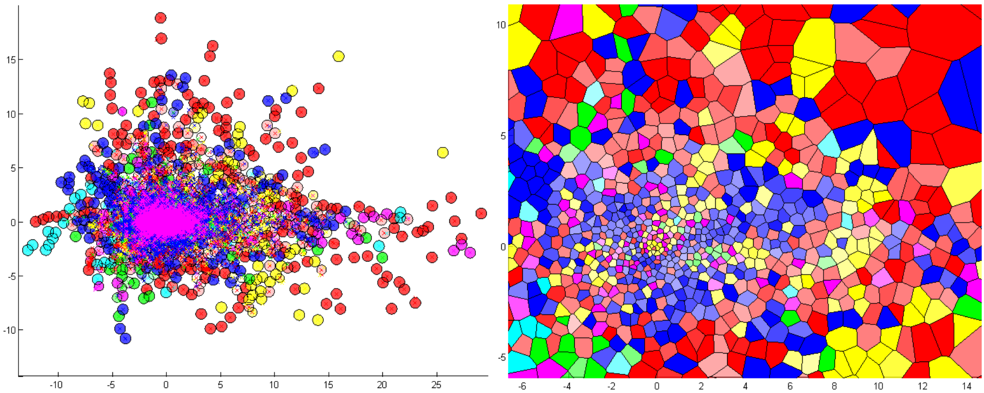


Figure 1: Left: the set of balls resulting from training on the first two principal components of local features extracted from the KTH dataset (colours denote labels, and color intensity expresses the 'purity' of the conditional class distribution within each ball). Right: a close-up of the central area represented as the Voronoi tessellation associated with the balls shows how the regions whose class statistics are more complex are covered by a finer set of balls.

4 Experiments

Batch and online settings. In this section we describe the baselines, the feature descriptors, and the datasets used in the experiments. To emphasize the versatility of our approach, we test the empirical performance of ABACOC in two learning settings: batch and online. In the batch setting, we follow the literature for each specific dataset, using available train-test splits, leave-one-out, or k-fold cross-validation (see below for details). The online setting is geared at an actual real-time scenario; in this case we are interested in the *average performance in time* of the sequence of models generated by ABACOC. Note that, in the online setting we do not measure the performance of a single classifier, but rather the performance

[2] A consistent classifier is one for which the probability of correct classification, given a training set, approaches, as the size of the training set increases, the best probability theoretically possible (Bayes optimal) if the population distributions were fully known.

of the ensemble of classifiers generated by the incremental training process, see below for details.

Datasets. We assessed our method on some of the most common computer vision benchmarks: Weizmann [17] and KTH [44] (all scenarios) for *action recognition*, SKIG [30] and MSRGesture3D [50] for *gesture recognition*, JAPVOW [34] and AUSLAN [22] for *sign language recognition* (UCI Repository [1]). The first four datasets are mostly footage material, which requires a feature extraction step at the frame level. Our approach for KTH and Weizmann datasets is based on the histogram of oriented optical flows (HOOF) [8] using 32 bins. For the dataset SKIG we used the same feature extraction pipeline of [12], which consists of 3DHOF on the RBG frames and GHOG (Global Histogram of Oriented Gradient) on the depth frames. For MSRGesture3D only depth information is available. We therefore extracted two-level pyramidal HOG (PHOG) features using 32 bins.

Competing approaches. Our baselines for batch experiments include Hidden Markov Models (HMM), Dynamic Time Warping (DTW), and Support Vector Machines (SVM). The standard technique using HMMs for classifying video sequences outputs the action associated with the trained HMM that achieves the highest likelihood score on the new frame sequence. We call this approach HMM-Lik —see, e.g., [5, 39][3] Recently, Antonucci et al. [6] proposed a method using instead the expected value w.r.t. the stationary distribution to calculate, from each HMM, a "static" feature that then gets processed via a standard machine learning classifier. As we used a 1-NN technique, this method will be here referred to HMM-1NN. Another baseline, which we refer to as DTW-d, consists in assigning to the new video the action of his nearest training sequence based on the DTW distance [49]. Fanello et al. [12] proposed a real-time prediction system based on one-vs-all SVMs learned on samples generated from a concatenation of a set of feature frames collected from a sliding window. We term this method SVM-b. For HMM-Lik and HMM-1NN we used the same setting as in [6], characterized by Gaussian HMMs with $N = 3$ hidden states. For SVM-b we selected a buffer of 12 frames for all datasets, except for the AUSLAN dataset on which a buffer of size 8 empirically leads to better results.

In the online setting, we compared our approach against SVM-b with a full re-train after the presentation of each new video and the kernel-based binary classifier ALMA [14]. Similarly to ABACOC, ALMA is trained incrementally. However, the learned classifier is linear, and provably approximates the SVM classifier. In order to provide a fair comparison against a nonparametric approach, we ran ALMA with a Gaussian kernel (since SVM with Gaussian kernels is known to be consistent —see [47]). ALMA's parameters were optimally tuned on the full dataset. As for ABACOC we used the Euclidean distance as metric ρ (other metrics could be considered for specific feature descriptors). Since feature vectors are normalized, the parameter R was set to a value smaller than 1. In most cases, this parameter can be safely set to the diameter of the feature space. Nonetheless, when the dataset is small, lower values may provide better results. In order to compute the intrinsic dimension parameter d of the space, which controls the shrinking rate of the radii, we used the method described in [31].

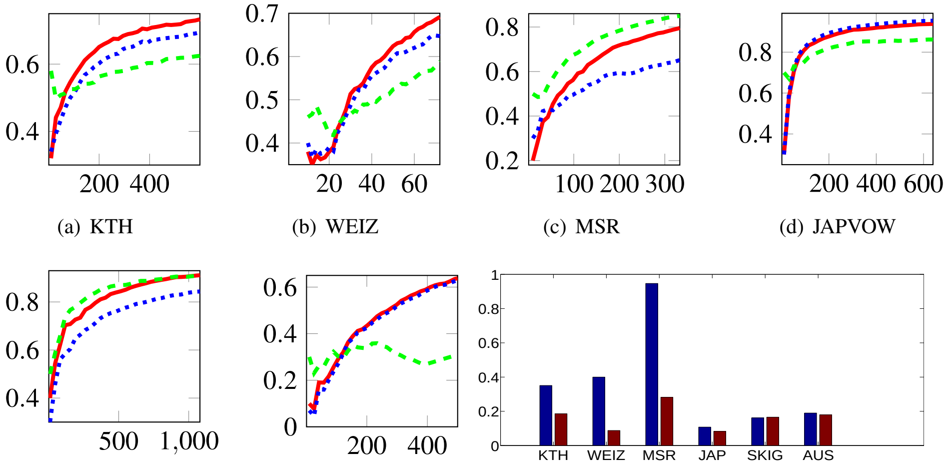
Results on batch classification. We used leave-one-out for KTH and Weizmann, a 3-fold cross-validation averaged on ten runs for SKIG, the same setting as [50] (i.e., a five-person test) for MSRGesture3D, and the already available training and test sets for JAPVOW and AUSLAN. As shown in Table 1, ABACOC is always among the best two methods on all datasets. This suggests that our algorithm, combined with state-of-the-art features, provides

[3] The better performances in [5] are due to aggressive feature selection which we did not use in this work

DATASET	**HMM-Lik**	**HMM-1NN**	**DTW-d**	**SVM-b**	**ABACOC**
KTH	49.92%	68.28%	52.50%	69.83%	**83.20%**
Weizmann	47.80%	87.50%	53.76%	97.22%	**98.61%**
SKIG	16.40%	90.30%	95.74%	94.50%	**97.50%**
MSRGesture3D	17.4%	78.20%	50.65%	**95.55%**	90.33%
JAPVOW	86.65%	95.67%	69.72%	84.59%	**98.01%**
AUSLAN	19.74%	67.07%	**83.81%**	44.78%	72.32%

Table 1: Multiclass accuracies of ABACOC compared against four baseline algorithms on the six benchmark datasets. All the methods share the same extracted features.

an accurate and efficient classification system across the board, even in a batch setting.



(a) KTH (b) WEIZ (c) MSR (d) JAPVOW

(e) SKIG (f) AUSLAN (g) Fraction Selected Frames

Figure 2: The plots *a* to *f* show the online performance of ABACOC (red solid line) against SVM-b (green dashed line) and ALMA (blue dotted line). The x-axis is the number of videos fed to the algorithms and the y-axis is the average accuracy over the ten random permutations. The plot *g* shows the fraction of frames selected by ABACOC as ball centers (red bars) and the fraction of frames chosen as support vectors selected by ALMA (blue bars). ABACOC consistently uses less training examples than ALMA to represent its classifier.

Results on incremental classification. We constructed ten random permutations of the videos in each dataset (only the first 500 videos of AUSLAN were selected because retraining SVM-b is computationally expensive). Then, we fed the videos to the algorithms one by one, in the order specified by the random permutation —see Algorithm 2.

The algorithms had to predict the label of each new incoming video. After each prediction, the video together with its label were given to each algorithm as a new training example. SVM-b was re-trained on all past examples including the new one. ABACOC and ALMA instead, perform an incremental training step (lines 3–19 of Algorithm 1 for ABACOC). The plots in Fig. 2 track the average accuracy of the predictions of each algorithm on increasing prefixes of the random permutation. For ABACOC and ALMA we also plot the fraction

Algorithm 2 Online protocol

Input: Video stream $(V_1, y_1), (V_2, y_2), \ldots$

for $t = 1, 2, \ldots$ **do**

  Receive V_t

  Predict $\widehat{y}_t$

  Receive true action y_t

  Update model fed new example (V_t, y_t)

end for

of frames selected, respectively, as centers and support vectors (plot *g*) as the permutation is fed to each algorithm. Notably, besides being computationally very demanding, SVM-b with re-train has an online performance that is not consistent across the different datasets. ALMA is never significantly better than ABACOC. Moreover, whereas prediction and incremental training of ABACOC are logarithmic in the number of balls, ALMA with kernels predicts and updates its linear model in time linear in the number of support vectors. Finally, whereas ABACOC is naturally multiclass, SVM-b and ALMA require costly one-vs-all reductions to deal with multiclass datasets.

5 Continuous action recognition

The training and prediction phases of our framework can be combined to obtain a truly online system, provided we endow the system with a method able to determine when a new training label is needed, for instance by assessing the confidence level of each classification. Think of a scenario in which a robot recognizes gestures made by humans. Whenever a gesture is classified with low confidence, the robot can query the human and obtain the correct label, which is then used for training. In this scenario the goal could be to minimize the robot's error rate given an admissible query rate.

Our method computes, for each frame, a score over the possible actions, see (2). This allows to analyse the evolution of the action probabilities over time, and segment a continuous sequence of multiple actions by detecting the beginning and the end of an individual action. We do that by using the algorithm presented in [12], which uses SVMs to assign scores. The rationale of the approach is that the standard deviation of the scores is informative. When at a given time a score is clearly higher than the others, the standard deviation is also high, and this is used as an indicator that some specific action is being performed. In contrast, when all scores are similar the standard deviation is low (see Fig. 3, blue dots), and this indicates the onset of a new action —see [12, Section 4.3.2] for details. We apply this method to the scores computed by ABACOC obtaining the pink line in Figure 3. Notably, the peaks of the curve correspond to actions being performed, while the minima represent transitions between actions. We isolate these points by computing the mean of the standard deviation over a small buffer of frames (Figure 3, cyan line). We refer again the reader to [12] for the details.

We tested our "spotting" approach on a home-made dataset of ten manipulative actions. These include different grasping actions, in which the whole hand or only the fingertips are used, release actions, where the fingers are opened, and motion actions, where the arm performs a right or a left movement. Each action was recorded 60 times in two different illumination settings and backgrounds, and 3DHOF and HOG descriptors were extracted

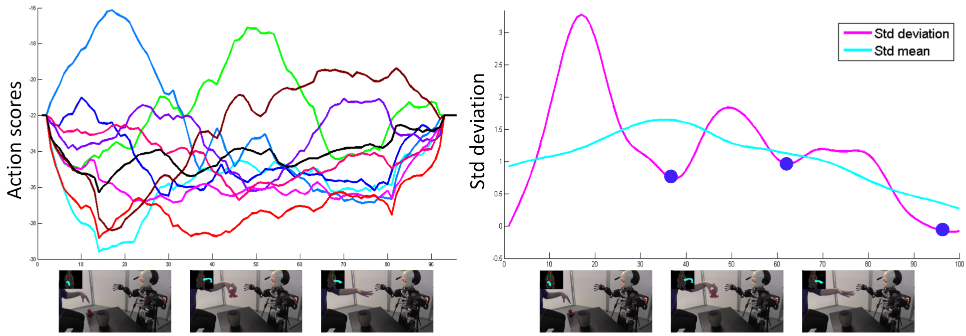


Figure 3: The evolution of the action probabilities over time for one of the test sequences containing 3 actions is depicted on the left. On the right, the pink line represents the standard deviation of the action probabilities for each frame, and the cyan line is the mean of the standard deviation computed over a small buffer of frames.

for each frame. We evaluated our algorithm on sequences representing pick and place activities, formed by a grasping action, a moving action and a releasing action. In order to compute the accuracy between a ground truth sequence and the estimated one, we employed the Levenshtein distance [28], defined as $\frac{S+D+I}{N}$. There, each action is treated as a symbol in a sequence; S represents the number of substitutions (misclassifications), D the number of deletions (false negatives) and I the number of insertions (false positives). Over 20 test sequences, the Levenshtein distance error was 0.12 for the SVM-b classifier and 0.08 for the one proposed here.

6 Conclusions and perspectives

This study has shown an efficient incremental nonparametric prediction system that can be combined with any set of frame-by-frame local feature descriptors for online action recognition. We showed the effectiveness of our approach in different scenarios, when compared to standard baselines methods. In order to test the real-time abilities of our system we have installed it in the RoboThespian[4] with features extracted from Kinect. We have published the code to encourage a widespread adoption of our framework[5]. Further research will focus on finding a confidence measure for the prediction scores, which could be used in a semi-supervised framework, for instance in human-robot interaction scenarios in which a robot can query the human to obtain the correct label whenever a gesture is classified with low confidence.

[4] www.robothespian.co.uk
[5] https://mloss.org/software/view/560/

References

[1] D.J. Newman A. Asuncion. UCI machine learning repository, 2007. URL http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html.

[2] J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Comput. Surv.*, 43(3):16:1–16:43, April 2011. ISSN 0360-0300. doi: 10.1145/1922649.1922653. URL http://doi.acm.org/10.1145/1922649.1922653.

[3] Anjum Ali and JK Aggarwal. Segmentation and recognition of continuous human activity. In *Detection and recognition of events in video, 2001. Proceedings. IEEE Workshop on*, pages 28–35. IEEE, 2001.

[4] Jonathan Alon, Vassilis Athitsos, Quan Yuan, and Stan Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(9):1685–1699, 2009.

[5] A. Antonucci, R. de Rosa, and A. Giusti. Action recognition by imprecise hidden markov models. In *Proceedings of the 2011 International Conference on Image Processing, Computer Vision and Pattern Recognition, IPCV 2011*, pages 474–478. CSREA Press, 2011.

[6] A. Antonucci, R. de Rosa, A. Giusti, and F. Cuzzolin. Robust classification of multivariate time series by imprecise hidden markov models. In *Accepted for publication Int. J. Approx. Reasoning IJAR*, 2014.

[7] Yang Bai, Lihua Guo, Lianwen Jin, and Qinghua Huang. A novel feature extraction method using pyramid histogram of orientation gradients for smile recognition. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 3305–3308. IEEE, 2009.

[8] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. Histograms of oriented optical flow and Binet-Cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[10] G. A. Di Caro, A. Giusti, J. Nagi, and L. Gambardella. A simple and efficient approach for cooperative incremental learning in robot swarms. In *Proceedings of the 16th International Conference on Advanced Robotics (ICAR)*, Montevideo, Uruguay, November 25–29, 2013.

[11] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 65–72. IEEE, 2005.

[12] Sean Ryan Fanello, Ilaria Gori, Giorgio Metta, and Francesca Odone. Keep it simple and sparse: Real-time action recognition. *J. Mach. Learn. Res.*, 14(1):2617–2640, January 2013. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2567709.2567745.

[13] Brook Galna, Gillian Barry, Dan Jackson, Dadirayi Mhiripiri, Patrick Olivier, and Lynn Rochester. Accuracy of the Microsoft Kinect sensor for measuring movement in people with Parkinson's disease. *Gait & Posture*, January 2014. ISSN 09666362. doi: 10.1016/j.gaitpost.2014.01.008. URL http://dx.doi.org/10.1016/j.gaitpost.2014.01.008.

[14] Claudio Gentile. A new approximate maximal margin classification algorithm. *The Journal of Machine Learning Research*, 2:213–242, 2002.

[15] Andrew Gilbert, John Illingworth, and Richard Bowden. Action recognition using mined hierarchical compound features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):883–897, 2011. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2010.144.

[16] Michael A. Goodrich and Alan C. Schultz. Human-robot interaction: A survey. *Found. Trends Hum.-Comput. Interact.*, 1(3):203–275, January 2007. ISSN 1551-3955. doi: 10.1561/1100000005. URL http://dx.doi.org/10.1561/1100000005.

[17] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, December 2007.

[18] Richard D Green and Ling Guan. Continuous human activity recognition. In *Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th*, volume 1, pages 706–711. IEEE, 2004.

[19] Michael B Holte, Thomas B Moeslund, and Preben Fihl. View-invariant gesture recognition using 3d optical flow and harmonic motion context. *Computer Vision and Image Understanding*, 114(12):1353–1361, 2010.

[20] Maodi Hu, Yunhong Wang, Zhaoxiang Zhang, De Zhang, and James J Little. Incremental learning for video-based gait recognition with lbp flow. *IEEE Trans Syst Man Cybern B Cybern*, 2012. ISSN 1941-0492. URL http://www.biomedsearch.com/nih/Incremental-Learning-Video-Based-Gait/22692925.html.

[21] Maodi Hu, Yunhong Wang, Zhaoxiang Zhang, De Zhang, and James J Little. Incremental learning for video-based gait recognition with lbp flow. *Cybernetics, IEEE Transactions on*, 43(1):77–89, 2013.

[22] J.K. Kies. *Empirical Methods for Evaluating Video-Mediated Collaborative Work*. PhD thesis, Virginia Tech, March 1997.

[23] Tae-Kyun Kim, Thomas Woodley, Björn Stenger, and Roberto Cipolla. Online multiple classifier boosting for object tracking. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 1–6. IEEE, 2010.

[24] Teddy Ko. A survey on behavior analysis in video surveillance for homeland security applications. *2013 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, 0: 1–8, 2008. doi: http://doi.ieeecomputersociety.org/10.1109/AIPR.2008.4906450.

[25] Samory Kpotufe and Francesco Orabona. Regression-tree tuning in a streaming setting. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *NIPS*, pages 1788–1796, 2013. URL http://dblp.uni-trier.de/db/conf/nips/nips2013.html#KpotufeO13.

[26] Robert Krauthgamer and James R. Lee. Navigating nets: Simple algorithms for proximity search. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 798–807, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics. ISBN 0-89871-558-X. URL http://dl.acm.org/citation.cfm?id=982792.982913.

[27] Hyeon-Kyu Lee and Jin H. Kim. An hmm-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21: 961–973, 1999.

[28] V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics - Doklady*, 1966.

[29] Jongwoo Lim, David A Ross, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for visual tracking. In *Nips*, volume 17, pages 793–800, 2004.

[30] Li Liu and Ling Shao. Learning discriminative representations from rgb-d video data. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, pages 1493–1500. AAAI Press, 2013. ISBN 978-1-57735-633-2. URL http://dl.acm.org/citation.cfm?id=2540128.2540343.

[31] Gabriele Lombardi, Alessandro Rozza, Claudio Ceruti, Elena Casiraghi, and Paola Campadelli. Minimum neighbor distance estimators of intrinsic dimension. In *Machine Learning and Knowledge Discovery in Databases*, pages 374–389. Springer, 2011.

[32] Yanyun Lu, Anthony Fleury, Jacques Boonaert, and Stéphane Lecoeuche. Online human recognition from video surveillance using incremental svm on texture and color features. In Abdelhamid Bouchachia, editor, *ICAIS*, volume 6943 of *Lecture Notes in Computer Science*, pages 26–39. Springer, 2011. ISBN 978-3-642-23856-7. URL http://dblp.uni-trier.de/db/conf/icais/icais2011.html#LuFBL11.

[33] Yanyun Lu, Khaled Boukharouba, Jacques Boonært, Anthony Fleury, and Stéphane Lecoeuche. Application of an incremental svm algorithm for on-line human recognition from video surveillance using texture and color features. *Neurocomputing*, 126: 132–140, 2014.

[34] J. Toyama M. Kudo and M. Shimbo. Multidimensional curve classification using passing-through regions. *Pattern Recognition Letters*, 20:1103–1111, 1999.

[35] Rashid Minhas, Abdul Adeel Mohammed, and QM Jonathan Wu. Incremental learning in human action recognition based on snippets. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(11):1529–1541, 2012.

[36] Sebastian Nowozin and Jamie Shotton. Action points: A representation for low-latency online human action recognition. 2012.

[37] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 716–723. IEEE, 2013.

[38] Seiichi Ozawa, Soon Lee Toh, Shigeo Abe, Shaoning Pang, and Nikola Kasabov. Incremental learning for online face recognition. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 5, pages 3174–3179. IEEE, 2005.

[39] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

[40] Aarthi Ravi. Automatic gesture recognition and tracking system for physiotherapy. Master's thesis, EECS Department, University of California, Berkeley, May 2013. URL http://www.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-112.html.

[41] Kishore K Reddy, Jingen Liu, and Mubarak Shah. Incremental action recognition using feature-tree. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1010–1017. IEEE, 2009.

[42] Zhou Ren, Jingjing Meng, Junsong Yuan, and Zhengyou Zhang. Robust hand gesture recognition with kinect sensor. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 759–760, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0616-4. doi: 10.1145/2072298.2072443. URL http://doi.acm.org/10.1145/2072298.2072443.

[43] Konrad Schindler and Luc Van Gool. Action snippets: How many frames does human action recognition require? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[44] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *Proc. of International Conference on Pattern Recognition*, 2004.

[45] Pramod Sharma, Chang Huang, and Ram Nevatia. Unsupervised incremental learning for improved object detection in a video. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3298–3305. IEEE, 2012.

[46] Masamichi Shimosaka, Yu Nejigane, Taketoshi Mori, and Tomomasa Sato. Fast online action recognition with efficient structured boosting. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 706–709. IEEE, 2009.

[47] Ingo Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, 18(3):768–791, 2002.

[48] Xinghua Sun, Mingyu Chen, and Alexander Hauptmann. Action recognition via local descriptors and holistic features. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 58–65. IEEE, 2009.

[49] G. A. Ten Holt, M. J. T. Reinders, and E.A. Hendriks. Multi-dimensional dynamic time warping for gesture recognition. *Time*, 5249:23–32, 2007.

[50] Jiang Wang, Zicheng Liu, Jan Chorowski, Zhuoyuan Chen, and Ying Wu. Robust 3d action recognition with random occupancy patterns. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part II*, ECCV'12, pages 872–885, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33708-6. doi: 10.1007/978-3-642-33709-3_62. URL http://dx.doi.org/10.1007/978-3-642-33709-3_62.

[51] Xinxiao Wu, Yunde Jia, and Wei Liang. Incremental discriminant-analysis of canonical correlations for action recognition. *Pattern Recognition*, 43(12):4190–4197, 2010.

[52] Xin Zhao, Xue Li, Chaoyi Pang, Xiaofeng Zhu, and Quan Z Sheng. Online human gesture recognition from motion data streams. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 23–32. ACM, 2013.