Bayesian Filter based Behavior Recognition in Workflows allowing for User Feedback

Dimitrios I. Kosmopoulos*, Nikolaos D. Doulamis**, Athanasios S. Voulodimos**

Abstract

In this paper, we propose a novel online framework for behavior understanding, in visual workflows, capable of achieving high recognition rates in real-time. To effect online recognition, we propose a methodology that employs a Bayesian filter supported by hidden Markov models. We also introduce a novel re-adjustment framework of behavior recognition and classification by incorporating the user's feedback into the learning process through two proposed schemes: a plain non-linear one and a more sophisticated recursive one. The proposed approach aims at dynamically correcting erroneous classification results to enhance the behavior modeling and therefore the overall classification rates. The performance is thoroughly evaluated under real-life complex visual behavior understanding scenarios in an industrial plant. The obtained results are compared and discussed.

Keywords:
Hidden Markov Models, behavior recognition, workflow, user interaction

1. Introduction

The great usefulness of human behavior recognition and understanding in a wide range of applications has attracted the interest of many researchers in

*University of Texas at Arlington, Computer Science and Engineering, Texas, USA
**National Technical University of Athens, School of Electrical & Computer Engineering, Zografou, Greece
Email addresses: dkosmo@ieee.org (Dimitrios I. Kosmopoulos), ndoulam@cs.ntua.gr (Nikolaos D. Doulamis), thanosv@mail.ntua.gr (Athanasios S. Voulodimos)

the areas of computer vision and machine learning. Virtual reality, human-computer interaction, smart environments building and smart monitoring are just a few applications, to which the significance of behavior recognition is indubitable. Especially when it comes to smart monitoring of large-scale enterprises/factories, such as industrial assembly lines, the importance of behavior recognition relates to the safety and security of the staff, to the reduction of costs, to production scheduling, as well as to the quality of the production process. The latter is guaranteed by enforcing adherence to strictly predefined procedures and activities for production or service provision.

In most current approaches the goal is either to detect activities, which may deviate from the norm, or to classify some isolated activities. Attempts to address the problem under discussion are encumbered by a number of important hindering factors; the high diversity of the actions and types of behaviors to be recognized is probably the most important one. The complexity of static object detection and moving object tracking, with the occlusions and illumination changes, naturally affect adversely approaches that follow the bottom-up paradigm.

Despite the above impediments, focusing on monitoring the production line of an industrial plant (such as an automobile manufacturer), which is a fairly structured process, makes modeling of the activities more realistic than in the case of a more unsystematic area of interest, e.g., an airport or a service maintenance system. The former processes are often hierarchically structured as workflows, that comprise sequential tasks. As opposed to isolated action monitoring, the goal here is to monitor activities that occur continuously.

This paper proposes innovative workflow recognition schemes for complex industrial environments such as the one of an automobile construction. The main idea is to classify online the visual observations into the available classes (industrial tasks), so that better production monitoring can be achieved. The proposed methodology recognizes the visual tasks as they are captured by the camera overcoming difficulties arising from the complexity of the environment and of the visual process. In addition, we exploit the expert feedback on part of the footage so as to enhance future classification results.

The online classification is performed in our case by combining a probabilistic theory with supervised time series classifier such as hidden Markov models (HMMs). The sole use of a probabilistic framework cannot solve the problem since we need to know the "a priori" statistics of the visual process. On the other hand, the sole use of HMMs makes the problem only suitable

for categorizing pre-segmented sequences, which severely deteriorate the industrial impact, since the expert users are forced to segment a priori the content (see e.g., [1], [2]).

Taking these observations into consideration, the work presented in this paper contributes mainly in the following ways:

- We present a novel online classification framework for distinct behavior recognition in visual workflows. The proposed framework is based on HMMs and Bayesian filtering and exploits prior knowledge.

- We also propose an approach for improving the supplied results by allowing interaction with the user, who may provide relevance feedback. Two different neural network based schemes are introduced: non-linear and recursive, the latter allowing for significant error decrease using a small number of training samples.

Furthermore, in contrast to most mainstream bottom-up approaches we employ features at the image level, bypassing the error-prone object detection and tracking steps.

The rest of this paper is organized as follows: After surveying the related literature regarding behavior understanding in section 2, we formulate the problem that we are proposing to solve in section 3. Then describe the proposed task representation in section 4 and the proposed online classification framework in section 5. Section 6 focuses on the neural network based rectification schemes. The experimental setup and the outcoming results are described and analyzed in section 7. Finally, section 8 concludes the paper with a summary of the findings.

2. Related Work

Event detection and especially human action recognition has been the focus of interest of computer vision and machine learning communities for years, mostly as isolated activities and not as part of a continuous process. A variety of methods has addressed these problems, including semi-latent topic models [3], spatial-temporal context [4], optical flow and kinematic features [5], and random trees and Hough transform voting [6]. Wada et al. [7] employ Non-deterministic Finite Automaton as a sequence analyzer to present an approach for multi-object behavior recognition based on behavior driven selective attention. Other works focus on more specific domains, e.g.,

event detection in sports [8, 9], retrieving actions in movies [10], human gesture recognition (using Dynamic Time Warping [11] and Time Delay Neural Networks [12]), and automatic discovery of activities [13]. Comprehensive literature reviews regarding isolated human action recognition can be found in [14, 15].

One of the key functionalities of any machine learning model (classifier) suitable for application in visual behavior understanding is the ability to extract the *signature* of a behavior from the captured visual input. The key requirements when designing such a classifier is (a) to support task execution in various time scales, since a task or parts of it may have variable duration; and (b) to support stochastic processes, because of the task intra-class variability and noise.

A very flexible framework for stochastic classification of time series is the HMM (see e.g., [16]). It can be easily extended to handle outliers (see e.g., [17]) and to fuse multiple streams (e.g., [18]). It is very efficient for application in previously segmented sequences (see e.g., [19], [20]), however when the boundaries of the sequence that we aim to classify are not known in advance, the search space of all possible beginning and end points make the search very inefficient [21]. A typical way to treat this problem is given in [22], where a dynamic programming algorithm of cost T^3, is used to perform segmentation and classify then the segments; however the cost is restrictive in real applications.

In the past there have been some efforts to exploit the hierarchical structure of some time series, e.g., by using the hierarchical HMMs [23]. Each state is considered to be a self-contained probabilistic model (an HHMM). Examples of such approaches can be found in [24], where the workflow in a hospital operating room is described. Another approach is the layered hidden Markov model (LHMM) (see [25]), which consists of N levels of HMMs where the HMMs on level $N+1$ corresponds to observation symbols or probability generators at level N. Every level i of the LHMM consists of K_i HMMs running in parallel. In that work a LHMM is used for event identification in meetings. In [26] structure learning in HMMs is addressed in order to obtain temporal dependencies between high-level events for video segmentation. An HMM models the simultaneous output of event-classifiers to filter the wrong detections.

In many workflows, such as in industrial production where a sequence of different tasks has to be completed, the execution of a task means that it will not appear again in the same workflow. Therefore the whole history of

tasks must be kept in memory to exclude false positives and the Markovian property is obviously not applicable. Thus, the above approaches have an inherent problem to describe such workflows.

The Echo State Network (ESN), see, e.g., [27], could be a promising method for online classification of workflow time series, because it does not make any explicit Markovian assumption. However, it was shown in [28] that it effectively behaves as a Markovian classifier, i.e., recent states have a far larger influence on the predicted state. The ESN has already been used in a work using the same dataset that we are using [29]. However, their results are not directly comparable to ours, since the features they are using are different.

In the proposed work we aim to alleviate the problems of online task segmentation and we by-pass the erroneous Markovian assumption by approximating the probability of a label sequence (for each incoming frame) given the whole observation history. To this end we employ Bayesian filtering. The related techniques for online behavior recognition have not been adequately investigated in the literature. In the works of [30] and [31] Rao Blackwell particle filters were used along with a dynamic Bayesian network for tracking of hierarchical events. In our work we make no assumptions about event hierarchy and the representation that we adopt is by definition very simple, so resorting to Rao Blackwell filters for state space reduction is not needed. The work in [32] notices the utility of particle filters in combination with an HMM, however it seeks to perform observation prediction, which is different from our classification problem.

A scheme using an HMM in combination with a particle filter was presented in [33] to model well-log data. More specifically, by using a single (modified) HMM, consisting of hidden and measurable states and with the help of a particle filter the sequence of HMM states was extracted. In contrast to [33], we propose online classification, i.e., to find which of the tasks has generated the current and all previous observations, using several HMMs (each of them modeling a separate task) and prior knowledge about task execution. Each visual task has to be modeled by a separate HMM due to their dynamic characteristics and due to their high complexity. Our work contributes by showing how to execute online multi-class classification by defining appropriately the proposal function, the model of prior knowledge and the integration of HMM models with a particle filter.

Regarding relevance feedback, it is a common approach for automatically adjusting the response of a system regarding information taken from

user's interaction [34]. Originally, it has been developed in traditional information retrieval systems [35], [36], but it has been now extended to other applications, such as surveillance systems [37], [38]. Relevance feedback is actually an online learning strategy which re-weights important parameters of a procedure in order to improve its performance. Re-weighting strategies can be linear or non-linear relying either on heuristic or optimized methodologies [34]. Linear and heuristic approaches usually adjust the degree of importance of several parameters involved in the selection process. Instead, non-linear methods adjust the applied method itself using function approximation strategies [39]. In this direction neural network models have been introduced as non-linear function approximation systems. However, such approaches have been applied for information retrieval systems instead of surveillance applications. It is clear that it is not straightforward to move from one type of application to another due to the quite different requirements of both applications.

A comprehensive review regarding algorithms of relevance feedback in image retrieval has been provided in [40]. In this paper, the authors lay emphasis on comparing different techniques of relevance feedback with respect to the type of training data, the adopted organization strategies, the similarity metrics used, the implemented learning strategies and finally the effect of negative samples in the training performance. However, the compared methods in [40] are designed in a static, non-dynamic framework. Instead, in real life applications, there exists a non-linear and time varying relationship between the input feature vectors and the target output states. Therefore, we need to introduce dynamic approaches for an efficient relevance feedback implementation.

In [41] transfer learning is adopted to address the aforementioned difficulties. The method is applied to detect events in Web videos [41]. With transfer learning, we can use auxiliary data from known classification problems, different from the user's target query, to decrease the amount of data needed to be fed back. The drawback of transfer learning is that it is actually assumed that the previous known environmental conditions, over which the classifier has been trained with, are similar with the current characteristics of the environment. This implies stationary environments. Instead, the performance of the relevance feedback in real life non stationary conditions is highly deteriorated. This is addressed in this paper by introducing an adaptable mechanism able to dynamically adjust the trained non-linear relationship using few samples that represent the current environmental con-

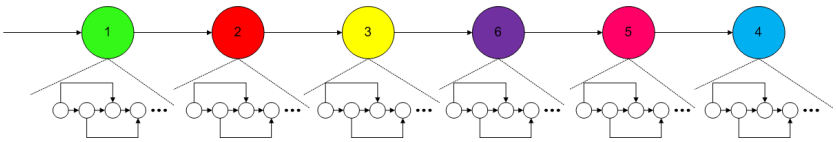


Figure 1: Graph representing the temporal relationships between the tasks of a typical workflow (the order of execution of tasks is an example; there are many possible permutations). Colored nodes represent tasks and white smaller nodes stand for intra-task states, which are the states of the HMM that models the respective task.

ditions along with a minimum degradation of the already obtained previous knowledge of the classifier.

3. Problem formulation

As stated in section 1, we focus on detecting and recognizing visual tasks in complex industrial processes (workflows), being executed in an automobile production line. The visual tasks are recognized from visual cues being captured from a set of cameras.

A workflow is a process that happens repetitively and consists of a sequence of discrete tasks. The order in which tasks appear matters, however permutations are allowed in some cases (which have to be learned). Tasks may have different durations, as a result of the natural differences in workers' productivity. The definition of tasks stems from domain knowledge. An example of such a task is: "A worker picks part1 from rack1 and places it on the welding cell".

A graph that presents the hierarchy of tasks that compose a workflow and their internal states is given in Fig. 1. Each workflow is composed of tasks and each task is modeled by a separate HMM.

Our goal is to determine which tasks are executed and when, given instances of workflows, which are described by sequences of visual observations. Let us denote as $\mathbf{o}_t$ the visual observation vector at the t time instance, or in discrete domain the frame number t. These visual observations are descriptors (features) extracted by processing the pixel values of frame t. The visual observations are described in subsection 4.1.

Our goal can be alternatively expressed as the classification of each frame t to one of the L available classes, i.e., different industrial tasks. Let us denote as $x_t = l_t$ the state vector including the label l_t from the L classes (tasks) that has to be assigned to frame t. Our goal is to calculate the posterior

$p(x_{0:t}|\mathbf{o}_{1:t})$ at every step, given the measurements (visual observations) up to that step.

The notation $x_{0:t}$, which will be used in the following section, accumulates all vectors x_i with $i = 0, ..., t$, that is $x_{0:t} = (x_0..x_t)$. Similarly, the notation $\mathbf{o}_{1:t}$ accumulates all observation descriptors up to time t, that is, $\mathbf{o}_{1:t} = (\mathbf{o}_1..\mathbf{o}_t)$.

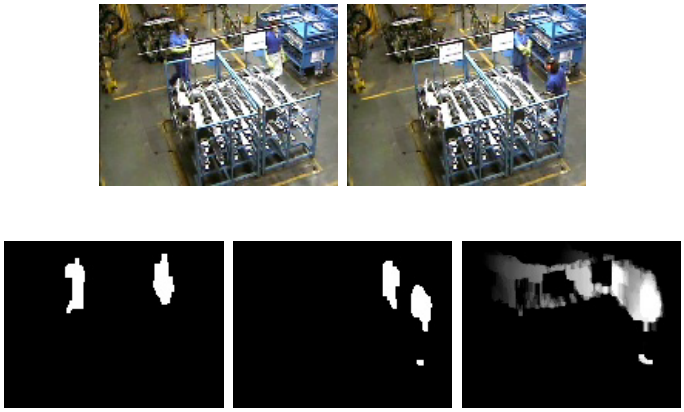


Figure 2: Two keyframes (first row), the respective background subtraction images and the extracted PCH image (second row)

4. Task modeling

In the following we describe how we represent the tasks. In subsection 4.1 we present means of representation of each frame, while in subsection 4.2 we present the HMM for modeling time series.

4.1. Visual Observations

One of the key challenges real-time action recognition systems are confronted with concerns selection of appropriate features for representing the observed raw data. The ideal features should describe different actions accurately, with high discrimination capability, and should be efficiently calculated. Ideally, these features should also provide a hierarchical representation scheme (coarse to fine) so that a desirable, application-wise, trade-off between representation capabilities and computational complexity can be reached.

The employment of features directly extracted from the video frames has the significant advantage of *obviating the need of detecting and tracking the salient scene objects*, a task which is notoriously difficult in cases of occlusions, target deformations, illumination changes etc. Thus, by using such an approach, the intermediate levels of semantic complexity, as met in typical *bottom-up* systems, are completely bypassed. For this purpose, either local or holistic features (or both [42]) may be used. *Holistic features* such as Pixel Change History (PCH) images [43] remedy the drawbacks of local features, while also requiring a much less tedious computational procedure for their extraction. A very positive attribute of such representations is that they can easily capture the history of a task that is being executed.

The PCH value of a pixel is defined as:

$$P_{\varsigma,\tau}(x,y,t) = \begin{cases} \min(P_{\varsigma,\tau}(x,y,t-1) + \frac{255}{\varsigma}, 255) \\ \text{if} D(x,y,t) = 1 \\ \max(P_{\varsigma,\tau}(x,y,t-1) - \frac{255}{\tau}, 0) \\ otherwise \end{cases} \quad (1)$$

where $P_{\varsigma,\tau}(x,y,t)$ is the PCH for a pixel at (x,y), $D(x,y,t)$ is the binary image indicating the foreground region, ς is an accumulation factor and τ is a decay factor. By setting appropriate values to ς and τ we are able to capture pixel-level changes over time (see Fig. 2). These images can then be transformed to a vector-based representation using moments such as the Zernike moments (see, e.g., [19]).

4.2. The HMM framework and its drawbacks

A common approach for stochastically modeling time series is to use hidden Markov models (HMMs). A hidden Markov model consists of states, transitions, observations and probabilistic behavior, and is formally defined as a tuple $\lambda = \langle \mathbf{Q}, \mathbf{A}, \mathbf{B}, \pi \rangle$ satisfying the following conditions:

- $\mathbf{Q} = \{q_1, ..., q_S\}$ is a finite set of S states. In our case, the number of states is an indication of the order (complexity) of the stochastic representation.
- $\mathbf{A}$ is the transition matrix, which represents the transition probabilities between states.
- $\mathbf{B}$ is the observation matrix, which represents the observation probability given the state.

- π represents the probability of each state at the beginning of the sequence.

A supervised training algorithm is used to obtain the parameters λ of the HMM. The training set is formed using representative samples of industrial tasks which have been manually classified to one of the L available classes. This implies that we need first to annotate the tasks, exploiting, for example, the experience of industrial engineers. We also need to identify the start and finish times for each industrial workflow even during the testing phase, which is a burden for a real-life exploitation of the algorithm in industrial environments. In real-world scenarios the starting and ending times of tasks are usually unknown. Therefore, HMM modeling can not be used for online recognition of the tasks. This is because online classification requires searching in the space of possible beginning and end points to perform Viterbi matches in order to find the optimally fitting sequence [16].

Assuming that tasks' appearance follow Markovian behavior (the conditional probability distribution of future tasks depends only upon the present task; that is, given the present, the future does not depend on the past) it is possible to perform online classification by applying techniques such as hierarchical (HHMM) and Layer hidden Markov models (LHMM) [23], [25]. However, such assumptions are not true in a real-world industrial environment, since the processes considered are structured. Usually, in a real-world production environment, the current execution of a task will affect the execution of future tasks, i.e., a task may be executed only once in a workflow.

All the above imply that the use of a conventional HMM for stochastically classifying industrial tasks is very inefficient, especially for real world sequences, which typically contain several thousands of frames. An exhaustive search for all possible combinations would be therefore practically prohibitive from a computational point of view. Hence, for an online classification framework, we need to identify the time boundaries, that is the start and finish times of an industrial task, which are part of a workflow. For this reason, an alternative methodology is required, which constitutes one of the main contributions of this paper.

To this end we propose in the following an approximate, though very efficient, method, which endows the HMM with online classification capabilities.

5. The Bayesian Filter based Classification Framework

A method for online recognition of industrial tasks in visual workflows based on Bayesian filters is proposed here. We assume that we are not aware of the start and finish times of the tasks.

As stated in section 3, our goal is to determine which tasks are executed and when, given instances of workflows. In other words, our goal is to calculate the posterior probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ for every frame t. Estimation of the posterior probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ is a much more complex process than estimating the posterior $p\left(x_t|\mathbf{o}_t\right)$, since in the former case, the probability depends on the classification results of the previous frames.

One possible method to calculate $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ is by employing a Bayesian filter. The solution for the Bayesian filter is commonly expressed as:

$$p\left(x_{0:t}|\mathbf{o}_{1:t}\right) = p\left(x_{0:t-1}|\mathbf{o}_{1:t-1}\right) \frac{p\left(\mathbf{o}_t|x_{0:t}, \mathbf{o}_{1:t-1}\right) p\left(x_t|x_{0:t-1}, \mathbf{o}_{1:t-1}\right)}{p\left(\mathbf{o}_t|\mathbf{o}_{1:t-1}\right)} \qquad (2)$$

Equation (2) is actually a recurrent expression of the probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ with respect to the previous estimates $p\left(x_{0:t-1}|\mathbf{o}_{1:t-1}\right)$ up to time $t-1$. However, the main difficulty in calculating $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ using equation (2) is the fraction term on the right part. To estimate this term, we need additional knowledge regarding the distribution of visual observations of image frame $t-1$ being aware of the class (i.e., task) that this frame belongs to.

One possible way to estimate the additional knowledge, required for the online classification framework, is to exploit a supervised classifier as the HMM, described in subsection 4.2. For this reason, we combine the HMM with the probabilistic framework, indicated by equation (2) to achieve online recognition of industrial tasks, disencumbered from the requirement to know start and finish times in advance.

To estimate the fraction term of the right part of equation (2) we proceed as follows. First, the term $p\left(\mathbf{o}_t|\mathbf{o}_{1:t-1}\right)$ is independent of the class to which the current frame should be assigned to, so it can be omitted from the following calculations.

Second, it is reasonable to assume that the observation $\mathbf{o}_t$ depends only on the current task x_t, so we simplify $p\left(\mathbf{o}_t|x_{0:t}, \mathbf{o}_{1:t-1}\right)$ to $p(\mathbf{o}_t|x_t)$. We propose to calculate this probability by using the observation model of the HMM, which is learned offline for each HMM state. Third, for the term $p\left(x_t|x_{0:t-1}, \mathbf{o}_{1:t-1}\right)$ we propose an alternative expression, which is the $p(x_t, c_t|x_{0:t-1}, \mathbf{o}_{1:t-1})$ or

more simply $p(x_t, c_t|x_{0:t-1})$; the latter holds because if the task history is known then the observation history does not affect the appearance of the next task. The variable c_t is a boolean stochastic variable, which becomes true if the task label changes from $t-1$ to t and false if the task remains the same. $p(x_t, c_t|x_{0:t-1})$ gives the probability that the task in current frame t has the label x_t and there is (or there is not) a switch to a new task, provided that the sequence of all previous task labels (task history) is known. More details about this probability can be found in sub-section 5.1.2.

Under these assumptions equation (2) becomes:

$$p\left(x_{0:t}|\mathbf{o}_{1:t}\right) \propto p\left(x_{0:t-1}|\mathbf{o}_{1:t-1}\right) \cdot p\left(\mathbf{o}_t|x_t\right) p\left(x_t, c_t|x_{0:t-1}\right) \tag{3}$$

As observed in equation (3), the posterior probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ is proportional to a) the recurrent term $p\left(x_{0:t-1}|\mathbf{o}_{1:t-1}\right)$, b) the probability $p(\mathbf{o}_t|x_t)$, which is estimated through the HMM model that captured the supervised knowledge of the task execution with regard to the visual observations and c) the $p(x_t, c_t|x_{0:t-1})$, which expresses our a priori knowledge about task duration and transition from one state to another.

It is clear, therefore, that the estimation of the probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ requires the a priori knowledge about task duration and transition, as well as the distribution of the visual observations (e.g., visual descriptors) with respect to the task that a frame belongs to. However, another difficulty of solving equation (3) is that it involves dependencies from previous frame observations (visual descriptors) and classification (frame assignment to one of the L available classes).

To handle the dependencies of the posterior probability $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ with the previous frame states (e.g., frame classification), we need first to introduce a list of hypotheses and then to validate them under a probabilistic framework. A common approach for performing that is through the usage of Particle Filters theory, which is a method for estimating the importance of a hypothesis according to a set of observed data.

5.1. Particle Filter Driven by the Hidden Markov Model

Let us assume that we have a set of N available hypotheses (particles). A hypothesis describes a particular combination of the classes that the previous frames have been assigned to. For example, a hypothesis is that the first frame belongs to the second task, the second frame to the same task, while the third to the first task, etc. Every hypothesis is evaluated through

the Bayesian filters, which are estimation methods based on simulation and previous observations [44], [45].

Weights are associated to the hypotheses, expressing the significance degree to the modeling process. Therefore given N hypotheses we have N weighted particle trajectories $\left\{x_{0:t-1}^{(n)}, w_{0:t-1}^{(n)}\right\}_{n=1}^{N}$. Each of these trajectories approximates the posterior probability $p\left(x_{0:t-1}|\mathbf{o}_{1:t-1}\right)$ up to time $t-1$.

Let us assume that the particle trajectories up to time $t-1$ are known. Then, we can compute the N particles $\left\{x_t^{(n)}\right\}_{n=1}^{N}$ which are combined with the previous trajectories to form $\left\{x_{0:t}^{(n)}, w_{0:t}^{(n)}\right\}_{n=1}^{N}$ to approximate the posterior $p\left(x_{0:t}|\mathbf{o}_{1:t}\right)$ up to time t. In particular, the current weight $w_t^{(n)}$ for the n^{th} hypothesis at the current frame t is estimated through the following distribution:

$$w_t^{(n)} = p(\mathbf{o}_t|x_t^{(n)}) \tag{4}$$

Equation (4) means that we can estimate the weights $w_t^{(n)}$ if we know a hypothesis, i.e., we know the class x_t to which frame t belongs. The pdf in equation (4) derives from the supervised learning of the HMM. The hypothesis about the value of x_t requires a priori knowledge regarding task duration and order of tasks. The hypothesis is generated by sampling the distribution $p(x_t, c_t|x_{0:t-1})$ which is learned offline (see subsection 5.1.1).

5.1.1. Estimation of the Observation Probability

The observation probability $p\left(\mathbf{o}_t|x_t\right)$ depends not only on the currently executed task but also on the state q of the associated HMM, so it should be fully written as: $p\left(\mathbf{o}_t|x_t, q_t\right)$. Here the HMM state that maximizes the observation probability is selected for each task.

At this point it should be noted that the hidden system state space (currently executed task) is one-dimensional and discrete, with low number of possible states (equals the number of possible tasks). Therefore the method is very efficient and a relatively low number of particles is required.

5.1.2. A Priori Knowledge

Our knowledge about the task order as well as the task duration can be used to estimate the distribution $p(x_t, c_t|x_{0:t-1})$, where we recall that c_t is

a boolean stochastic variable, which becomes true if the task label changes from $t-1$ to t and false if the task remains the same.

Using the Bayes rule we can conclude to:

$$p(x_t, c_t|x_{0:t-1}) = p(x_t|x_{0:t-1}, c_t) \cdot p(c_t|x_{0:t-1}) \tag{5}$$

The term $p(c_t|x_{0:t-1})$ depends only on the duration d of the current task. In other words $p(c_t|x_{0:t-1}) \equiv p(c_t|x_{t-1}, d)$. A common approach for modeling $p(c_t|x_{t-1}, d)$ is to use a Gaussian mixture model of the respective task duration, which can be learned offline. Thus, we have:

$$p(c_t|x_{t-1}, d) = \sum_{i=1}^{K} m_i N(\mu_i, \sigma_i) \tag{6}$$

where K the number of mixture components and m_i, μ_i, σ_i the prior, mean and standard deviation of the i^{th} component.

The other term of equation (5) is given by

$$p(x_t|x_{0:t-1}, c_t) = \begin{cases} 0 & if \quad c_t = true \\ 1 & if \quad c_t = false \end{cases} \tag{7}$$

in the case that $x_t = x_{t-1}$,

and by:

$$p(x_t|x_{0:t-1}, c_t) = \begin{cases} 0 & if \quad c_t = false \\ T_{x_{0:t}} & if \quad c_t = true \end{cases} \tag{8}$$

in the case that $x_t \neq x_{t-1}$. $T_{x_{0:t}}$ is the probability of a path in a decision tree describing the possible transition paths between tasks. The tree is described in the following.

Assuming that the task with value m is possible to appear in the i-th order, we may denote $x(i) = m$. There will be a node in the i-th level of the tree with value equal to m. Given that there are several tasks that may follow that task directly after its completion, i.e., $x(i+1)$ may take n values, in the tree the node with value $x(i) = m$ will have n descendants, with these values.

The root of the tree is defined to be a virtual node (with no associated task value), while its children indicate the tasks that may start the workflow. Additionally, each link that connects a parent node with value $x(i)$ to a child node with value $x(i+1)$ has an associated value $l(x(i+1), (x(i)))$, which

indicates the probability of occurrence of the task $x(i+1)$ directly after $x(i)$ is finished. A complete workflow is represented by a path connecting the root with any of the tree leaves. Given a path P, the probability of $p(P)$ is given by $P(p) = \prod_{pathlinks} l(x(i+1), x(i))$, which is the product of the associated probabilities for all the links in the path.

Such a tree can be learned by using a training set of full workflows, and therefore the "legitimate" paths and their probabilities can be specified. More specifically, for each parent node we find the possible successors (descendants) and based on the normalized frequency that a specific child is selected as next task, we assign a probability value to the connecting link. Since the history of all previous tasks is maintained by using such a tree, we do not rely on the Markovian assumption.

Algorithm 1 presents the steps of the proposed online learning classification framework that combines stochastic modeling and HMMs. The proposed framework is depicted in Fig. 3 as dynamic Bayesian network, where all dependencies are graphically presented.

6. A Neural Network-Based Rectification Scheme

The main drawback of the aforementioned probabilistic approach is that the observation probability $p(\mathbf{o}_t|x_t)$ may sometimes give rise to the wrong task as a result of the EM-based learning, which can be trapped in local maxima. This in turn may result in some particles taking a wrong "trajectory". To address this difficulty, we present in the following a rectification framework able to automatically adjust the $p(\mathbf{o}_t|x_t)$ (the link $\{3\}$ in the graph of Fig. 3) according to user's feedback. The rectification strategy is based on the usage of a dynamic non-linear classifier, which is able to learn the current user's feedback, as expressed by a small set of selected relevant/irrelevant data, while simultaneously provide a minimum degradation of the previous obtained knowledge. Since the rectification function is expected to be non-linear we based our approach on neural networks.

At this point we should mention that any method capable of handling non-linear functional relationships could probably substitute our neural network based approach. Alternatives that could have been addressed include non-linear (kernel) regression [46] or random forests [47].

Algorithm 1 Proposed Method

{OFFLINE TRAINING}
{Decision tree learning}
DecisionTree = Learn(AllTaskPaths)
{Supervised task learning through HMM}
for $s = 1$ to $NumberOfTasks$ **do**
$\prec \mathbf{Q}_s, \mathbf{A}_s, \mathbf{B}_s, \pi_s \succ$ = TrainHMM(AllTaskTimeSeries)
end for
{ONLINE CLASSIFICATION FRAMEWORK}
while (F=AcquireFrame()) $\neq$ NULL **do**
$\mathbf{o}_t$ = ProcessFrame(F); {extraction of visual observations (features) by processing the current captured frame}
{for every Hypothesis do}
for $n = 1$ to N **do**
$x_t^{(n)} = Sample\ p\left(x_t, c_t | x_{0:t-1}^{(n)}\right)$
{get HMM state q_t that maximizes observation probability}
$q_t = \arg\max_q \{p(o_t|x_t, q)\}$
$Weight\ \mathbf{x}_t^{(n)}$ by the following:

$$w_t^{'(n)} = p\left(\mathbf{o}_t | x_t^{(n)}, q_t\right) \tag{9}$$

end for
for $n = 1$ to N **do**
$Normalize$ the weights by:

$$w_t^{(n)} = \frac{w_t^{'(n)}}{\sum_{n_1=1}^{N} w_t^{'(n_1)}} \tag{10}$$

end for
Switching-state particles with low weight are set back to previous state.
for $n = 1$ to N **do**
Update $p(x_{0:t}^{(n)}|o_{1:t})$ {use the recurrent framework described in eq. 3 }
end for
The winner is the particle $n_0 : p(x_{0:t}^{(n_0)}|o_{1:t}) \geq p(x_{0:t}^{(n)}|o_{1:t}), \forall n \in \{1, ..., N\}$
end while

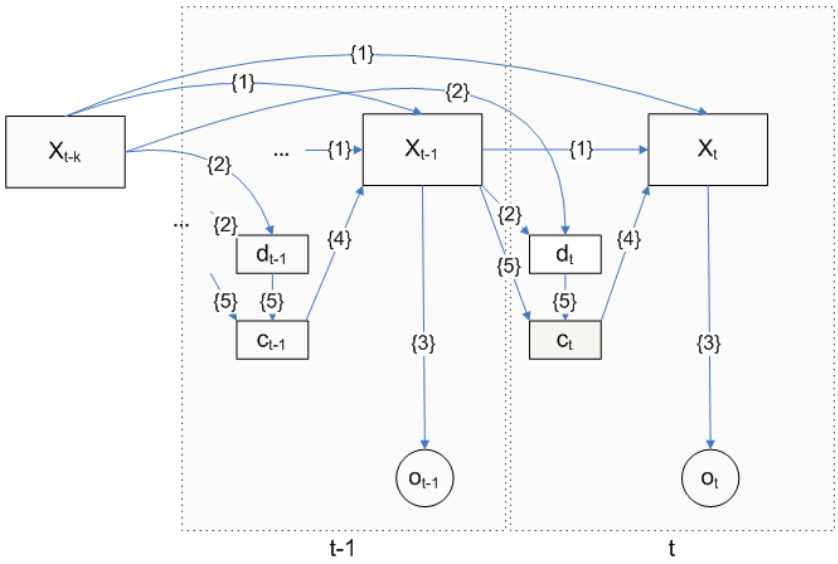


Figure 3: The proposed scheme represented as a dynamic Bayesian network, where rectangles correspond to discrete values and circles to continuous values. The digit-annotated edges represent the dependencies in our framework as follows: {1}: the dependency of the current state x_t on the state history {2}: state duration given the previous states {3}: $p(o_t|x_t)$, derived from HMM. {4} and {1} represent the equations (7,8). {5} represents the equation (6).

6.1. The Non-linear Model

Let us denote as S a set that contains the selected samples by the user. The set $S = \{\cdots(\mathbf{p}_i, \mathbf{d}_i)\cdots\}$ contains pairs of the form $(\mathbf{p}_i, \mathbf{d}_i)$, where as $\mathbf{p}_i$ we indicate the observation probability vector, generated by the HMM, the elements of which express the probability of the corresponding frame to belong to one of the, say, M available classes. Vector $\mathbf{d}_i$ indicates the ideal probabilities for the i^{th} sample. Variable $\mathbf{d}_i$ is an indicator vector meaning that all its elements will be zero apart from one which is equal to one. This element indicates the class that this task belongs to. Assuming the existence of a non-linear function able to correct the erroneous classifications of the HMM, we can derive the following equation

$$\mathbf{d}_i = \underline{f}(\mathbf{p}_i) \tag{11}$$

where $\underline{f}(\cdot)$ is a vector function indicating the non-linear relationship between $\mathbf{p}_i$ and $\mathbf{d}_i$.

The main difficulty with the previous equation is the fact that vector function $\underline{f}(\cdot)$ is actually unknown. Additionally, the non-linear relationship dynamically changes under different conditions and camera system modification. To address the aforementioned difficulties, we introduce a feed forward neural network model able to accurately approximate the unknown vector function $\underline{f}(\cdot)$ with a certain degree of accuracy. In this case, equation (11) is written as follows:

$$\mathbf{d}_i = \underline{f}_{\mathbf{w}}(\mathbf{p}_i) \tag{12}$$

The main difference between equations (11) and (12) is the introduction of the vector weight $\mathbf{w}$. This means that different parameters (weights) of the network yields different performance of the adaptable classifier. Vector $\mathbf{w}$ includes all the parameters (weights) of the non-linear neural network-based classifier.

To estimate the weights $\mathbf{w}$ we need to apply a training algorithm, which actually minimizes the mean square error among all selected from the expert user data (task sequences) and the respective output of the network when a particular set of weights is applied. That is,

$$\mathbf{w} = \arg \min_{forall\mathbf{w}} \epsilon = \arg \min_{forall\mathbf{w}} \sum_i (\underline{f}_{\mathbf{w}}(\mathbf{p}_i) - \mathbf{d}_i)^2 \tag{13}$$

The back propagation algorithm [48] can provide a solution to this non-linear minimization problem. In our experiments, we select a small neural network structure of few hidden neurons and one hidden layer. In this case, we try to minimize the number of neural networks parameters, that is the size of weight vector $\mathbf{w}$. It is clear that the samples of the training set S should be greater than the number of neural network parameters, that is the dimension of the weight vector $\mathbf{w}$. Since the size of the neural network is small few training samples are required.

6.2. Recursive Implementation

The main drawback of the aforementioned approach is that a large number of samples, as provided by the user's interaction through a set of relevant/ irrelevant data is required for training the non-linear classifier. However, in real-life applications this large training set usually are constructed from data taken from quite different environmental conditions. This deteriorates the performance of the neural network model since, on the one hand, it contains

contradictory samples, and on the other, it is an averaging solution over data taken from different environmental conditions. To address the aforementioned difficulties a recursive implementation is proposed in this paper. This implementation requires few training samples that express the current user's feedback through the selection of a set of relevant / irrelevant data. Then, the modification of the neural network weights is accomplished by minimizing the current network error with, however, a minimal modification of the previous neural network knowledge or in other words a minimum modification of the neural network weights.

In particular, let us assume that $\mathbf{w}_b$ is the weights of the neural network classifier. Then, we denote as $\mathbf{w}_n$ the new neural networks weights after the implementation of the recursive algorithm. Assuming that the new neural network weights $\mathbf{w}_n$ are related with the weights $\mathbf{w}_b$ with a very small modification $\mathbf{dw}$, we can provide a very efficient training algorithm able to re-adjust the performance of the neural network classifier to the current user's preferences.

In particular, let us denote that the output of the neural network classifier at a given time instance (e.g., the i^{th} frame) is $\mathbf{d}_i = \underline{f}_{\mathbf{w}_b}(\mathbf{p}_i)$. In this case, we assume that the weights $\mathbf{w}_b$ are used for the classifier. Let us now assume that the output of the classifier is not correct. The user provides the output target through user's interaction. Let us denote this target as $\hat{\mathbf{d}}_i$. Variable $\hat{\mathbf{d}}_i$ refers to the actual output of the neural network classifier as provided by the user's interaction. Then, the small amount of $\mathbf{dw}$ is estimated through the following equation (see [49], [50])

$$\mathbf{e} = \hat{\mathbf{d}}_i - \mathbf{d}_i = \mathbf{a}^T \cdot \mathbf{dw} \tag{14}$$

Equation (14) is derived by using Taylor series expansion on the neural network model (see [50]). In equation (14) vector $\mathbf{a}$ contains elements of the previous network weights $\mathbf{w}_b$. Using only equation (14), we cannot estimate the small perturbation $\mathbf{dw}$. This is due to the fact that only one (or few) equations are not enough to estimate the multi-dimensional vector $\mathbf{dw}$. For this reason, an additional constraint is required. In this approach, the variable $\mathbf{dw}$ is estimated through the minimum modification of the previous network knowledge. Another approach is given by the minimization of the norm of the perturbation $\mathbf{dw}$ resulting in the following minimization approach

$$\hat{\mathbf{dw}} = argmin\|\mathbf{dw}\| \tag{15}$$

subject to

$$\mathbf{e} = \hat{\mathbf{d}}_i - \mathbf{d}_i = \mathbf{a}^T \cdot \mathbf{dw} \tag{16}$$

The previous equations express a convex minimization problem. Therefore, it can be easily solved either analytically using Lagrange multipliers or arithmetically using the gradient projection method [51]. In this paper, we adopt the second option to reduce computational complexity. This is due to the fact that the arithmetic approach is actually an iterative method. Thus, we can restrict the number of iterations with respect to the computational cost needed. In particular, the method starts from a feasible solution, i.e., an arbitrary solution that satisfies the constraints. Then, this solution is iteratively updated according to the gradient of the square function $\|\mathbf{dw}\|$, as being projected on the hyperplane defined by the constraint. It seems that this modification is very efficient for correcting erroneous mistakes of the HMM model combined with the particle filters (see the section of experimental results).

Summarizing, relevance feedback is a methodology of dynamically updating the system response, by either modifying the system parameters or the entire system itself, through information provided by the user, regarding the relevance of a set of few samples selected by the user and feedback to the system. We have used a non-linear relationship, modeled through a dynamic neural network architecture, for the relevance feedback implementation. In particular, at specific or randomly selected time instances, the user interacts with the system, by indicating the perfect target output (real task) for this particular time instance (captured frame). This selected image frame is feedback to the proposed adaptable architecture in order to improve the neural network response at possible future samples. In a nutshell, the proposed relevance feedback framework is described in the following:

- At a random image frame the user interacts with the system by providing the perfect target output for this frame.
- The respective output of the HMMs (observation probabilities per HMM model for the selected image frame) are feedback to the adaptable neural network architecture.

- The gradient project method is used to estimate the small perturbation $d\mathbf{w}$ using equations (15), (16) and information of the sample given by the user.
- The weights of the neural network are updated using the relation $\mathbf{w}_n = \mathbf{w}_n + d\mathbf{w}$.
- The updated network is used to recalculate the task observation probabilities.

7. Experiments and Results

We experimentally verified the applicability of the described methods. To this end, we have acquired some very challenging videos from the production line of a major automobile manufacturer (see [52]). Our previous efforts to apply a detection-tracking scheme have failed due to the low resolution, the heavy occlusions and the illumination changes (e.g., due to welding sparks).

7.1. Experimental Setup

The production cycle on the production line included tasks of picking several parts from racks and placing them on a designated cell some meters away, where welding took place. Each of the above tasks was regarded as a class of behavioral patterns that had to be recognized. The information acquired from this procedure can be used for the extraction of production statistics or anomaly detection. Partial or total occlusions due to the racks made the classification task difficult to effect.

The behaviors (tasks) we were aiming to model in the examined application are briefly the following:

1. One worker picks part #1 from rack #1 and places it on the welding cell.
2. Two workers pick part #2a from rack #2 and place it on the welding cell.
3. Two workers pick part #2b from rack #3 and place it on the welding cell.
4. A worker picks up parts #3a and #3b from rack #4 and places them on the welding cell.
5. A worker picks up part #4 from rack #1 and places it on the welding cell.

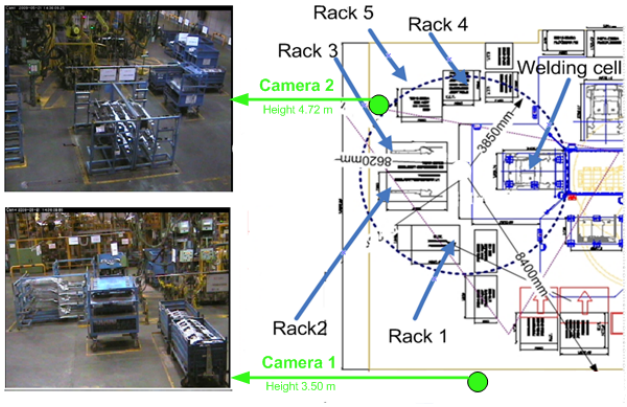


Figure 4: Depiction of a workcell along with the position our camera (camera 1) and the racks #1-5. The recognized behaviors are associated with transferring each part from the respective pallet and putting it on the welding cell.

6. Two workers pick up part #5 from rack #5 and place it on the welding cell.

In addition to these we had a null task (referenced as task 7), during which the workers were idle or absent.

The workspace configuration and the cameras' positioning is given in Fig. 4. A sample task (task 2) is presented in Fig. 5. The work cycle that we sought to model, despite the noise and the several outliers (e.g., persons walking into the working cell, vehicles passing by etc), remains a structured process and is a good candidate to model with holistic features.

For our experiments, we have used 20 sequences representing full assembly cycles, each one containing each of the defined behaviors The length of each sequence ranges from 2000 frames to 4000[1]. The annotation has been done manually. The dataset that we used is unique in the sense that it presents some well defined tasks which are executed in a repetitive and rather structured manner, providing several samples, which is good for learning (of course there is intra-class variability between the same tasks but still the resulting time series are correlated and can be learned and recognized). Fur-

[1]We are going to make the dataset publicly available. It is currently available for review purposes on
http://www.4shared.com/dir/sYeCqK5d/SignalProcessingVideoAnalytics.html
(folder:dataset1 - password:xyz543)

Figure 5: Typical execution of task 2. The relatively low resolution and the several occlusions and self occlusions make very difficult the task of tracking thus necessitating a holistic method

thermore, it includes phenomena such as occlusions, illumination changes and uniform appearance of humans, which make reliable tracking rather unrealistic, though perfectly suitable for approaches based on holistic features. Finally, partially overlapping views are available, which facilitates occlusions handling in our later research steps.

7.2. Holistic representation and online classification

To represent each video frame with a feature vector, we followed the method described in the subsection 4.1. For capturing the spatio-temporal variations we have set the parameters at $\varsigma = 10$ and $\tau = 70$. We have chosen to use the Zernike moments up to sixth order along with the center of gravity and the area, as feature vector. The higher the order of moments that we employ, the more detailed the region reconstruction will be, but also the more processing power will be required. Limiting the order of moments used is also justified by the fact that the details captured by higher order moments have much higher variability and are more sensitive to noise.

Specifically, we employed the complex moments A_{00}, A_{11}, A_{20}, A_{22}, A_{31}, A_{33}, A_{40}, A_{42}, A_{44}, A_{51}, A_{53}, A_{55}, A_{60}, A_{62}, A_{64}, A_{66}, for each of which we used the norm and the angle, except for A_{00},A_{20},A_{40},A_{60}, for which the angle

was always constant. Additionally the center of gravity and the area were used, making a total of 31 parameters, thus providing an acceptable scene reconstruction without a computationally prohibitive dimension. Zernike moments have been calculated in rectangular regions of interest of approximately 15000 pixels in each image, to limit the processing and allow real time feature extraction.

For activity recognition we used three-state HMMs with one mixture component per state to model each of the tasks described above, making a discrete search space of size 7; this was a good trade-off between performance and efficiency. In all cases, we employed full covariance matrices for the adopted observation (mixture) models. We trained all our models using the EM algorithm and we used the first ten scenarios for training and the rest ten for testing.

Although the selection of the features to represent each frame is independent of the proposed classification method, we have compared the features described in subsection 4.1 to the Local Motion Grid (LMG) features that have been used on the same dataset (see our work [20], or [29]) to ensure high accuracy. Using the same HMM configuration and a leave-one-out cross validation scheme for 20 scenarios considering up to seven pre-segmented tasks per scenario, we measured for cameras 1 and 2 a total accuracy of 53.57% and 67.14% for the LMG features [20], versus 84.14% and 86.42% respectively for the PCH-Zernike representation [19]. This comparison justifies the sole use of the PCH-Zernike features in the rest of the experiments.

We have compared the proposed method to some baseline methods the first one being the standard HMM. We have taken sliding time windows of constant size, which was equal to the mean duration of tasks in the training set. For each of those windows we applied the HMM models that represent each task and the winner was the one giving the higher likelihood. Using a voting scheme we were able to classify each frame.

The second baseline method that we used for comparison was the echo state network (ESN). We used a network of 1000 nodes, which was efficient for real time execution and avoided overfitting. Increasing the number of nodes would result in very high memory requirements without actual benefit in accuracy or recall. It also had seven output nodes, each one of them corresponding to a predicted task. The median of the last 101 estimations was taken to ensure lower jitter in the output. We have used the Matlab toolbox provided by the authors [27] using spectral radius 0.60, input scaling 0.3 and noise level 0.0003 after some experimentation for optimal results.

For our method we used only 100 particles, which was a good trade-off, and we were able to perform the whole processing at a rate of about 20Hz, the most costly of which was the feature extraction. The confusion matrix per task for a typical case is given in Fig. 6a. The learning phase included learning the task durations using a Gaussian mixture model, the task trajectories using a decision tree and the task models using HMMs.

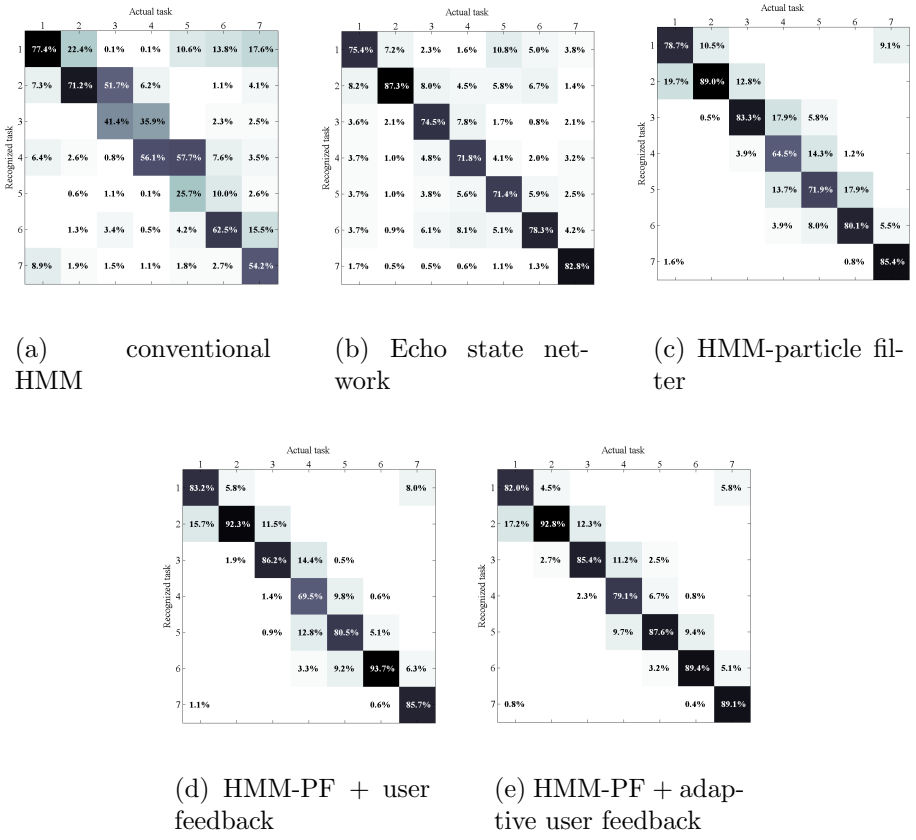
For all methods we extracted recall and precision. Recall indicates the number of true positives divided by the total number of positives in the ground truth ($REC = TP/(TP+FN)$). Precision is the number of true positives divided by the number or true and false positives ($PRC = TP/(TP + FP)$). The average precision and recall per task and the standard deviation are given in Fig. 7, after performing twenty iterations. The overall results are given in Table 1 and the confusion matrices are given in Fig. 6.

As expected the conventional HMM has the worst performance, mainly due to the fact that the task durations may vary, while it uses a sliding window of constant size. Although the ESN performs generally well, each task may be mistaken for almost any other one, as becomes clear from the confusion matrix. This happens mainly because actually only the most recent observations guide the prediction, thus ignoring the whole history (effective Markovian behavior [28]). Providing constraints given the history of tasks helps to discard erroneous task transitions by utilizing a decision tree, as we have explained in subsection 5.1. This becomes obvious when observing the confusion matrix corresponding to our method, where errors seem to be distributed among mutually accessible tasks. Clearly, the more restrictive the structure of the tasks, the more effective such a scheme will be, because the particles will be scattered around the most probable tasks according to the observation history.

In our method the particle that was able to explain best the sequence according to (3) was considered to be the winner. In all cases the work cycle, which consisted of all tasks 1 to 6 and the null was successfully recognized. In few cases the tasks were identified but were misaligned with the real ones; this was mainly due to features' imperfections, which sometimes gave rise to the wrong tasks, due to occlusions and noise.

7.3. User feedback

Regarding the user feedback mechanism, we firstly used a feed-forward neural network model for estimating the distribution probability of a frame to



(a) conventional HMM

(b) Echo state network

(c) HMM-particle filter

(d) HMM-PF + user feedback

(e) HMM-PF + adaptive user feedback

Figure 6: Confusion matrices for the various methods

belong to one of the seven available categories. The feed-forward neural network model (see Fig. 8) has one hidden layer. In our experiments 15 hidden neurons have been selected. As the number of hidden neurons increases the complexity of the neural network training significantly increases as well and generalization performance of the network decreases. In our simulations, we have seven output neurons for the neural model that indicate the probability for one of the seven available categories. As input layer of the neural network model we have the seven observation probabilities of the current frame produced by HMM. The neural network model re-adjusted the probabilities according to the knowledge provided through supervised training. We used the scenarios 1 to 3 to map the incoming maximum observation probabilities given each task (as provided by the HMM) to the ideal ones.

Table 1: Overall precision and recall for 10 test scenarios.

Method	HMM	ESN	HMM-PF	HMM-NN	HMM-NN recurs.
precision	0.603	0.777	0.797	0.851	0.871
recall	0.565	0.772	0.788	0.846	0.863

Table 2: Overall precision and recall for HMM-PF rectified by the recursive neural network.

number of particles	30	60	100	200
precision (%)	87.52	87.88	87.12	87.45
recall (%)	86.63	86.83	86.28	86.44

The neural network output re-adjusts the probabilities of the combined HMM + particle filter model in order to increase the efficiency of the workflow recognition module. There are, however, several cases, where the performance of the neural network model is not satisfactory. For this reason, we used the on-line learning mechanism for dynamically updating the parameters of the neural network model in order to satisfy with more efficiency the target outputs. In particular, at random selected time instances the user provided the perfect target outputs by setting the probability of the desired task to 1 and the other ones close to zero. We did that for 70 random samples. Then, the system updated the parameters of the neural network model so that i) the corrected target output was satisfied as much as possible, while simultaneously ii) the minimum modification of the previous network weights (that is previous neural network performance) was satisfied. Then, the new estimated weights were used for predicting the workflow state of future image frames.

Clearly the performance was improved compared to the previous approach in terms of precision and recall using the same number of particles (see Fig. 6e, Fig. 7 and Table 1). The high performance verifies that the network is able to adapt its weights to minimize the error according to the most recent input samples.

When using the recursive neural network we discovered that the increase of the particles does not significantly affect the overall precision and recall. The results are presented in Table 2 given for 30, 60, 100 and 200 particles. This implies that the proposed feedback scheme provided a good estimate even with low number of samples.

8. Conclusions

In this work we have proposed a novel online framework for behavior recognition in workflows in real-time. In the context of the framework we have handled two important problems for behavior recognition: (a) online behavior classification through a Bayesian filter which is approximated through particles driven by HMM and (b) rectification of erroneous classifications through interaction with the user.

The holistic features gave a good scene representation, thus helping us bypass the difficult tasks of detection and tracking that fail in such complex sequences. The conventional application of Viterbi to obtain optimal result would make the recognition task infeasible given the fact that no initial and end sequence points were known. Furthermore, our method did not rely on the Markovian assumption, which is not appropriate for monitoring workflows.

The proposed methods have been applied with promising results in some very challenging real sequences from an automobile manufacturing process. The good online recognition rates achieved by the particle filter/HMM method are additionally improved significantly when we employed the neural network based rectification scheme that incorporated user feedback. The recursive scheme seemed to perform even better and required fewer particles to achieve similar performance.

In the case of very long tasks, it would be meaningful to have more particles and maintain more hypotheses, e.g., in the case of consecutive workflows. If the workflows can be separated, e.g., when for several particles all expected tasks are considered finished, it would be more practical to reset the particles and to start from the beginning.

In our experiments we used unique tasks, which always had to appear due to the industrial assembly workflow requirements. In different settings, where probably omission or repetition of tasks would be possible we would only need to model these omissions/repetitions as prior knowledge, i.e., possible paths in the tree that we defined in subsection 5.1.2. What is only needed is a good estimation of which task is probable to appear next; that requirement is covered by the proposed representation.

In our future work we are going to investigate some less structured scenarios with more complex interactions and tasks with even higher variability, also considering different viewpoints via fusion schemes.

References

[1] D. Kosmopoulos, A. Voulodimos, T. Varvarigou, Robust human behavior modeling from multiple cameras, in: Pattern Recognition (ICPR), 2010 20th 697 International Conference on, 2010, pp. 3575–3578.

[2] N. Doulamis, A. Voulodimos, D. Kosmopoulos, T. Varvarigou, Enhanced human behavior recognition using hmm and evaluative rectification, in: ACM Multimedia, ARTEMIS Workshop, 2010.

[3] Y. Wang, G. Mori, Human action recognition by semilatent topic models, Pattern Analysis and Machine Intelligence, IEEE Transactions on 31 (10) (2009) 1762 –1774.

[4] Q. Hu, L. Qin, Q. Huang, S. Jiang, Q. Tian, Action recognition using spatial-temporal context, in: Pattern Recognition (ICPR), 2010 20th International Conference on, 2010, pp. 1521 –1524.

[5] S. Ali, M. Shah, Human action recognition in videos using kinematic features and multiple instance learning, Pattern Analysis and Machine Intelligence, IEEE Transactions on 32 (2) (2010) 288 –303.

[6] A. Yao, J. Gall, L. Van Gool, A hough transform-based voting framework for action recognition, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 2061 –2068.

[7] T. Wada, T. Matsuyama, Multiobject behavior recognition by event driven selective attention method, Pattern Analysis and Machine Intelligence, IEEE Transactions on 22 (8) (2000) 873 –887.

[8] D. Sadlier, N. O'Connor, Event detection in field sports video using audio-visual features and a support vector machine, Circuits and Systems for Video Technology, IEEE Transactions on 15 (10) (2005) 1225 – 1233.

[9] M.-H. Hung, C.-H. Hsieh, Event detection of broadcast baseball videos, Circuits and Systems for Video Technology, IEEE Transactions on 18 (12) (2008) 1713 –1726.

[10] I. Laptev, P. Perez, Retrieving actions in movies, in: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007, pp. 1 –8.

[11] A. Bobick, A. Wilson, A state-based approach to the representation and recognition of gesture, Pattern Analysis and Machine Intelligence, IEEE Transactions on 19 (12) (1997) 1325 –1337.

[12] M.-H. Yang, N. Ahuja, Extraction and classification of visual motion patterns for hand gesture recognition, in: Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on, 1998, pp. 892 –897.

[13] R. Hamid, S. Maddi, A. Bobick, M. Essa, Structure from statistics - unsupervised activity analysis using suffix trees, in: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007, pp. 1 –8.

[14] R. Poppe, A survey on vision-based human action recognition, Image and Vision Computing 28 (6) (2010) 976 – 990.

[15] W. Hu, T. Tan, L. Wang, S. Maybank, A survey on visual surveillance of object motion and behaviors, Systems, Man and Cybernetics, Part C, IEEE Transactions on 34 (3) (2004) 334–352.

[16] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of the IEEE 77 (2) (1989) 257–286.

[17] S. P. Chatzis, D. I. Kosmopoulos, T. A. Varvarigou, Robust sequential data modeling using an outlier tolerant hidden Markov model, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (9) (2009) 1657–1669.

[18] Z. Zeng, J. Tu, B. Pianfetti, T. Huang, Audio-visual affective expression recognition through multistream fused HMM, IEEE Trans. Multimedia 10 (4) (2008) 570–577.

[19] D. Kosmopoulos, S. Chatzis, Robust visual behavior recognition, Signal Processing Magazine, IEEE 27 (5) (2010) 34 –45.

[20] A. Voulodimos, H. Grabner, D. I. Kosmopoulos, L. J. V. Gool, T. A. Varvarigou, Robust workflow recognition using holistic features and outlier-tolerant fused hidden markov models, in: ICANN (1), 2010, pp. 551–560.

[21] S. Eickeler, A. Kosmala, G. Rigoll, Hidden markov model based continuous online gesture recognition, in: In Int. Conference on Pattern Recognition (ICPR, 1998, pp. 1206–1208.

[22] F. Lv, R. Nevatia, Recognition and segmentation of 3-d human action using hmm and multi-class adaboost, in: ECCV06, 2006, pp. IV: 359–372.

[23] S. Fine, Y. Singer, N. Tishby, The hierarchical hidden markov model: Analysis and applications, Machine Learning 32 (1) (1998) 41–62.

[24] N. Padoy, D. Mateus, D. Weinland, M.-O. Berger, N. Navab, Workflow Monitoring based on 3D Motion Features, in: Workshop on Video-Oriented Object and Event Classification in Conjunction with ICCV 2009, IEEE, Kyoto Japan, 2009, pp. 585–592.

[25] N. Oliver, A. Garg, E. Horvitz, Layered representations for learning and inferring office activity from multiple sensory channels, Comput. Vis. Image Underst. 96 (2) (2004) 163–180.

[26] T. Xiang, S. Gong, Optimising dynamic graphical models for video content analysis, Comput. Vis. Image Underst. 112 (2008) 310–323.

[27] H. Jaeger, W. Maass, J. Principe, Special issue on echo state networks and liquid state machines, Neural Networks 20 (3) (2007) 287 – 289.

[28] C. Gallicchio, A. Micheli, Architectural and markovian factors of echo state networks, Neural Networks 24 (5) (2011) 440 – 456.

[29] G. V. Veres, H. Grabner, L. Middleton, L. J. V. Gool, Automatic workflow monitoring in industrial environments, in: ACCV (1), 2010, pp. 200–213.

[30] X. Zhang, G.-Y. Xu, X.-L. Xiao, L.-M. Tao, Online analysis of hierarchical events in meetings, in: Usability and Internationalization, HCI and Culture, Vol. 2, 2007, pp. 472 –479.

[31] X. Xiaoling, L. Layuan, Real time analysis of situation events for intelligent surveillance, in: Computational Intelligence and Design, 2008. ISCID '08. International Symposium on, Vol. 2, 2008, pp. 122 –125.

[32] D. Zhang, X. Ning, X. Liu, Smc method for online prediction in hidden markov models, Kybernetes 38 (10) (2009) 1819–1827.

[33] P. Fearnhead, P. Clifford, On-line inference for hidden markov models via particle filters, Journal Of The Royal Statistical Society Series B 65 (4) (2003) 887–899.

[34] N. Doulamis, A. Doulamis, Evaluation of relevance feedback schemes in content-based retrieval systems, Signal Processing: Image Communication 21 (4) (2006) 334 – 357.

[35] J. J. Rocchio, Relevance feedback in information retrieval, in: G. Salton (Ed.), The Smart retrieval system - experiments in automatic document processing, Englewood Cliffs, NJ: Prentice-Hall, 1971, pp. 313–323.

[36] N. D. Doulamis, A. D. Doulamis, Evaluation of relevance feedback schemes in content-based in retrieval systems, in: Signal processing:image communication, Vol. 21, Elsevier, 2006, pp. 334–357.

[37] A. Oerlemans, J. T. Rijsdam, M. S. Lew, Real-time object tracking with relevance feedback, in: Proceedings of the 6th ACM international conference on Image and video retrieval, CIVR '07, ACM, New York, NY, USA, 2007, pp. 101–104.

[38] Z. Chengcui, C. Wei-Bang, C. Xin, Y. Lin, J. John, A multiple instance learning and relevance feedback framework for retrieving abnormal incidents in surveillance videos, Journal of Multimedia 5 (2010) 310–321.

[39] A. Doulamis, N. Doulamis, Generalized nonlinear relevance feedback for interactive content-based retrieval and organization, IEEE Trans. on Circuits and Systems for Video Technology 14 (5) (2004) 656–671.

[40] X. S. Zhou, T. S. Huang, Relevance feedback in image retrieval: A comprehensive review, Multimedia Systems 8 (6) (2003) 536–544.

[41] A. Lam, A. K. Roy-Chowdhury, C. R. Shelton, Interactive event search through transfer learning (2011) 157–170.

[42] X. Sun, M. Chen, A. Hauptmann, Action recognition via local descriptors and holistic features, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 58–65.

[43] T. Xiang, S. Gong, Beyond tracking: modelling activity and understanding behaviour, International Journal of Computer Vision 67 (2006) 21–51.

[44] M. S. Arulampalam, S. Maskell, N. Gordon, A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188.

[45] D. Arnaud, G. Simon, A. Christophe, On sequential monte carlo sampling methods for bayesian filtering, Statistics and Computing 10 (3) (2000) 197–208.

[46] G. Seber, C. Wild, Nonlinear Regression, Wiley, Hoboken, New Jersey, 2003.

[47] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.

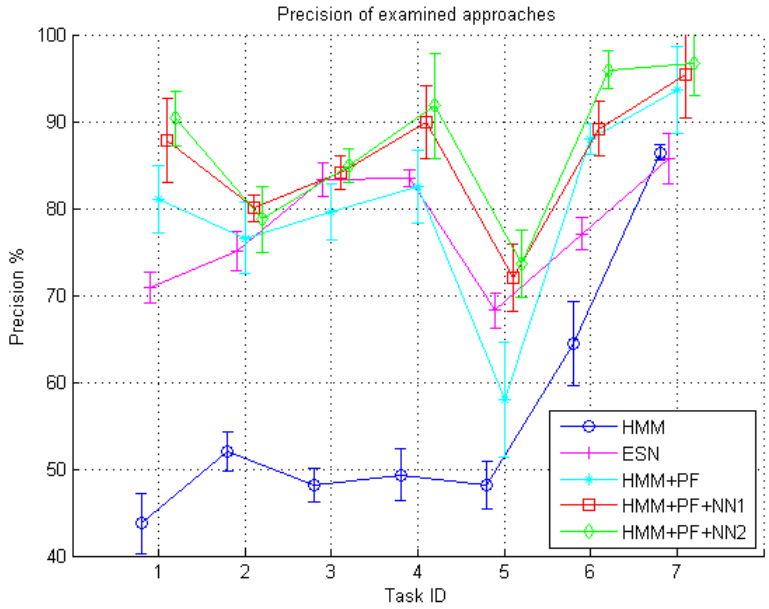
[48] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, MIT Press, Cambridge, MA, USA, 1988, pp. 696–699.

[49] A. Doulamis, Adaptable neural networks for objects' tracking reinitialization, Vol. 5769 LNCS of Lecture Notes in Computer Science, 2009, pp. 715–724.

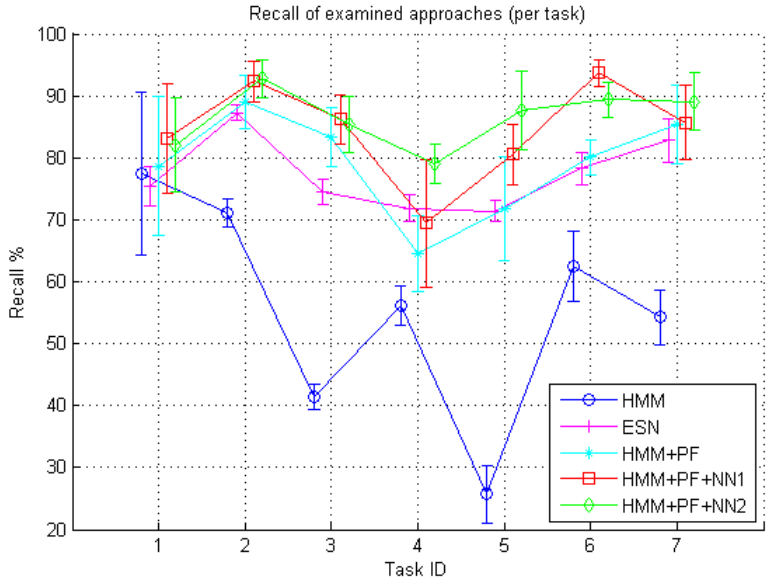
[50] A. Doulamis, Knowledge extraction in stereo video sequences using adaptive neural networks, Intelligent Multimedia Processing with Soft Computing, Springer-Verlag Berlin Heidelberg, 2005, pp. 235–252.

[51] D. G. Luenberger, Linear and nonlinear programming, Addison-Wesley, 1984.

[52] A. Voulodimos, D. Kosmopoulos, G. Vasileiou, E. Sardis, A. Doulamis, V. Anagnostopoulos, C. Lalos, T. Varvarigou, A dataset for workflow recognition in industrial scenes, in: IEEE Int. Conference on Image Processing, 2011.



(a) Precision



(b) Recall

Figure 7: Comparison of mean precision-recall metrics and standard deviation for conventional HMM, ESN, HMM-PF, HMM-PF-NN1 (non-linear), HMM-PF-NN2 (recursive)

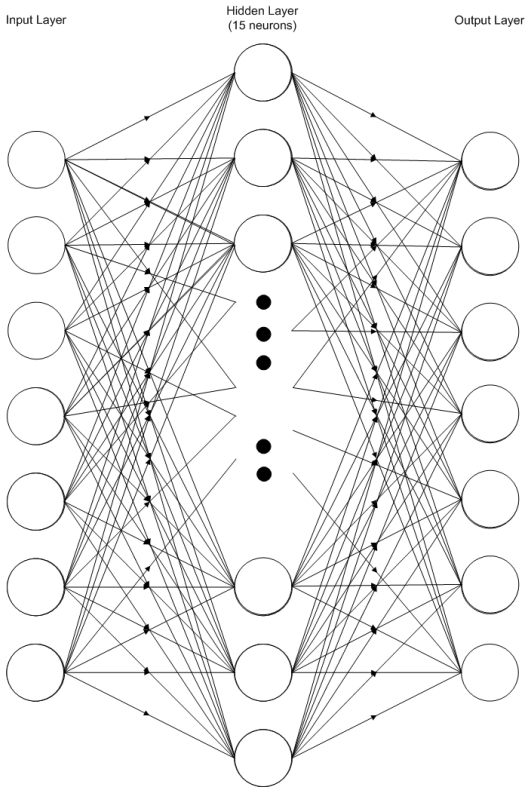


Figure 8: The feed-forward neural network used in the proposed rectification scheme. The input and output layers consist of seven nodes each (one for each task) and the hidden layer comprises 15 nodes