Neural Networks ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

2011 Special Issue

Online classification of visual tasks for industrial workflow monitoring

Athanasios Voulodimos [a,*], Dimitrios Kosmopoulos [b], Galina Veres [c], Helmut Grabner [d], Luc Van Gool [d,e], Theodora Varvarigou [a]

[a] School of Electrical and Computer Engineering, National Technical University of Athens, Greece
[b] Department of Computer Science & Engineering, University of Texas at Arlington, USA
[c] IT Innovation Centre, University of Southampton, UK
[d] Computer Vision Laboratory, ETH Zurich, Switzerland
[e] ESAT-PSI/IBBT, K.U. Leuven, Belgium

ARTICLE INFO

Keywords:
HMM
ESN
Genetic algorithm
Fusion
Workflow
Activity recognition

ABSTRACT

Modelling and classification of time series stemming from visual workflows is a very challenging problem due to the inherent complexity of the activity patterns involved and the difficulty in tracking moving targets. In this paper, we propose a framework for classification of visual tasks in industrial environments. We propose a novel method to automatically segment the input stream and to classify the resulting segments using prior knowledge and hidden Markov models (HMMs), combined through a genetic algorithm. We compare this method to an echo state network (ESN) approach, which is appropriate for general-purpose time-series classification. In addition, we explore the applicability of several fusion schemes for multicamera configuration in order to mitigate the problem of limited visibility and occlusions. The performance of the suggested approaches is evaluated on real-world visual behaviour scenarios.

1. Introduction

Intelligent visual surveillance and classification of visual tasks are research fields that have rapidly gained momentum over recent years. Focusing on industrial plant smart monitoring, the aim is to recognise tasks happening in the scene, to monitor the smooth running of a workflow, and to detect any abnormal behaviour. Deviations from the workflow may cause severe deterioration of the quality of the product or may raise safety or security hazards.

An example of such an industrial scenario is shown in Fig. 1. By monitoring industrial scenes, one faces several challenges such as recording data in work areas (camera positions and viewing area), industrial working conditions (sparks and vibrations), cluttered background (upright racks and heavy occlusion of the workers), high similarity of the individual workers (nearly all of them wearing a similar utility uniform), and other moving objects (welding machines and forklifts). Furthermore, the dynamics of the workflow can be quite complex. Several tasks within a workflow can have very different lengths and can be permutable. The high intraclass and low interclass variances make the classification process significantly challenging. Moreover, the tasks can include both human actions and motions of machinery in the observed process.

Related work. Behaviour and workflow recognition has attracted the interest of many researchers. In the computer vision and machine learning communities, this is mainly addressed in applications such as abnormal behaviour recognition or unusual event detection. Many approaches have been suggested over recent years—reviews can be found in (Poppe, 2010; Turaga, Chellappa, Subrahmanian, & Udrea, 2008). Typically they build a model of normality, and the methods can differ in (i) the model used, (ii) the algorithm employed for learning the model parameters, and (iii) the features used. Models might be previously trained and kept fixed (Antonakaki, Kosmopoulos, & Perantonis, 2009; Wang, Ma, Ng, & Grimson, 2008) or adapt over time (Breitenstein, Grabner, & Gool, 2009) to cope with changing conditions. A broad variety of extracted image features are used, such as global scene three-dimensional (3D) motion (Padoy, Mateus, Weinland, Berger, & Navab, 2009) or object trajectories (Antonakaki et al., 2009; Johnson & Hogg, 1996; Nguyen, Phung, Venkatesh, & Bui, 2005; Shi, Huang, Minnen, Bobick, & Essa, 2004), which require accurate detection and tracking. On the other hand, holistic methods, which define features at the pixel level and try to identify patterns of activity using them directly, can bypass the challenging processes of detection and tracking. Such methods may use pixel or pixel group features such as colour, texture, or gradient; see, for example, (Zelnik-Manor & Irani, 2006)

* Corresponding address: School of Electrical and Computer Engineering, National Technical University of Athens, Heroon Polytechneiou 9, 15780 Athens, Greece. Tel.: +30 210 772 2559.
E-mail address: thanosv@mail.ntua.gr (A. Voulodimos).

0893-6080/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2011.06.001

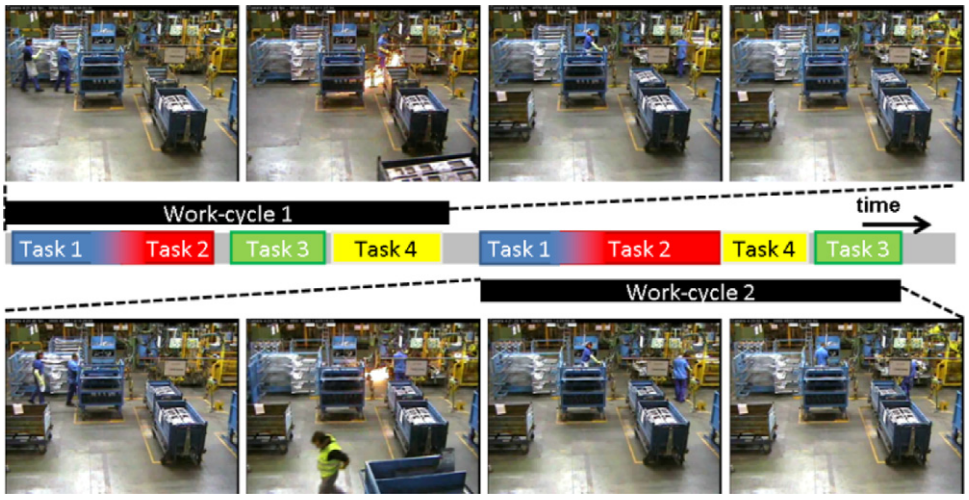


Fig. 1. Example of an industrial scenario: the workflow consists of several tasks of different length, which can be permutable (e.g., tasks 3 and 4 in this example), and the scene may present irrelevant motion and occlusions.

(histograms of spatiotemporal gradients) and (Laptev & Perez, 2007) (spatiotemporal patches). Pixel change history (PCH) is used in (Xiang & Gong, 2006) to represent each target separately after frame differencing. However, the representation of objects in PCH images is very simplistic (through ellipses), and cannot cope with realistic environments. A popular feature to use for action recognition is optical flow (see, e.g., Efros, Berg, Mori, & Malik, 2003), where a relatively small region of interest is extracted around a single human actor. In our case we need a much more efficient method, since our goal is online classification at high frame rates. Furthermore, in real applications, the targets may be partially occluded, so action recognition as defined in works such as (Efros et al., 2003) would not be feasible.

Various machine learning and statistical methods have been used for activity recognition, such as clustering (Boiman & Irani, 2005) and density estimation (Johnson & Hogg, 1996). A very popular approach is hidden Markov models (HMMs) (Ivanov & Bobick, 2000; Lv & Nevatia, 2006; Padoy et al., 2009), due to the fact that they can efficiently model stochastic time series at various time scales. However, the HMMs assume that the input data are already segmented, an assumption which significantly limits their application in realistic applications. For this purpose, more complex HMM-based methods have been proposed such as hierarchical HMMs (HHMMs) (Fine, Singer, & Tishby, 1998; Padoy et al., 2009) and layered HMMs (LHMMs) (Oliver, Garg, & Horvitz, 2004). However, the applicability of these methods assumes that the Markovian assumption holds for the tasks to be recognised, in other words the probability for the appearance of a task depends only on the previous one; this is not true in structured applications, where the execution of a task may influence the appearance of a series of following tasks. In such cases, the Markovian assumption would be an oversimplification, which would violate the application constraints. The use of higher-order models would result in very high complexity (Rabiner, 1989) and would raise issues such as "how many previous states do we have to consider?". With a small number of tasks the problem could be still tractable; however, such approaches are not scalable to large numbers of tasks.

In (Shi et al., 2004), the feasible task paths in a glycose calibration process were defined, using the so-called P-net to encode possible paths. The goals in our work are similar, but here we aim to show how to employ the HMM framework for recognising tasks in workflows, because of its very important extension possibilities (for example, with fusion (Zeng, Tu, Pianfetti, & Huang, 2008) or robustness (Chatzis, Kosmopoulos, & Varvarigou, 2009)); furthermore, we are going to encode possible paths as solutions provided by a genetic algorithm to cover a huge search space efficiently.

An alternative approach to the HMM for the analysis of complex dynamical systems is echo state networks (ESNs) (Jaeger, 2001). ESNs offer several benefits, such as (i) fast and simple learning of many outputs simultaneously, (ii) the possibility of both offline and online learning, (iii) the capability of directly dealing with high-dimensional input data, and (iv) the ability to learn complex dynamic behaviours without any explicit Markovian assumption. On the other hand, there are two main limitations involved: (i) they can only recognise repetitive dynamics and (ii) all significant variations of task order in a given workflow have to be learnt to provide the best classification results. Previously, ESNs have been successfully used for time-series classification in speech recognition (Skowronski & Harris, 2007), human–robot interactions (Hellbach, Strauss, Eggert, Komer, & Gross, 2008), emotion recognition (Scherer, Oubbati, Schwenker, & Palm, 2008), and medicine (Verplancke et al., 2010). Recently, we examined the effectiveness of ESNs for workflow recognition from a single camera (Veres, Grabner, Middleton, & Gool, 2010).

Nevertheless, the target visibility of specific tasks can be limited due to camera configuration and self-occlusions; therefore efficient ways to fuse observations from multiple cameras are necessary. Several fusion schemes for HMMs have been presented in the past, such as synchronous HMMs (Dupont & Luettin, 2000), parallel HMMs (Vogler & Metaxas, 1999), and multistream fused HMMs (Zeng et al., 2008). However, their applicability in multicamera systems has been examined only to a limited extent, for example in (Voulodimos, Grabner, Kosmopoulos, Van Gool, & Varvarigou, 2010), a previous work that is extended in this paper to address online behaviour and workflow recognition in continuous data streams, and in (Kosmopoulos & Chatzis, 2010), where offline classification of segmented sequences was examined. As far as ESNs are concerned, to our knowledge no fusion techniques have been employed for similar applications.

Contribution. To our knowledge, no state-of-the-art tracking-based approach is able to cope with the significant particular challenges (as described above) of workflow analysis in continuous streams within industrial environments. We tried state-of-the-art methods for person detection/tracking (Felzenszwalb, McAllester, & Ramanan, 2008; Grabner & Bischof, 2006)[1]; however, none

[1] The code was downloaded from the authors' webpages.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001

(a) Person detection (Felzenszwalb et al., 2008). (b) Person tracking (Grabner & Bischof, 2006).

Fig. 2. Examples of state-of-the-art methods for detection and tracking. These approaches fail because of the dataset's significant challenges, such as severe occlusions and cluttered background.

of them showed stable and robust results in our industrial environment. Fig. 2(a) shows typical failures of the detector in our dataset, with a recall of 24% and a precision of only 9%. Thus, tracking-by-detection approaches (e.g., Huang, Wu, & Nevatia, 2008) cannot be used to generate trajectories. Also, the person could be hardly tracked, as displayed in Fig. 2(b). As for the tracker, it may start very well; however, it soon loses the person and drifts away.

The reasons for the failures pertain to the nature of the environment, i.e., significant occlusions, clutter similar in structure/shape to a person, the workers coloured similarly to the racks, and unstable background due to welding flare, machinery operation, and lighting changes. Any of these in isolation would cause problems for person detection and tracking, but all of them together make the problem especially difficult for both detection and tracking, and prohibit the use of approaches based on trajectory analysis.

Hence, we choose to use holistic features, which can be efficiently computed, do not rely on target detection and tracking, and can be used to model complex scenes (Veres et al., 2010). We contribute to the solution in the following ways.

- We propose a novel method to automatically segment the input stream and to classify the resulting segments using prior knowledge and HMMs, combined through a genetic algorithm (GA).
- We compare this approach to an online ESN-based method for time-series analysis of continuous streams.
- We suggest using fusion schemes for multiple cameras to provide wider scene coverage, and to better cope with occlusions, thus improving the accuracy.

The rest of this work is organised as follows. Section 2 formally defines the problem. In Section 3, we describe scene descriptors. Sections 4 and 5 describe the HMM-based fusion architectures and the proposed continuous stream segmentation method, while Section 6 presents the proposed GA–HMM that combines HMM classifications of the automatically segmented tasks and prior knowledge. In Section 7, the ESN-based approach addressing fusion is described. Section 8 is the experimental section, while Section 9 discusses the lessons learnt from our research and concludes the paper.

2. Problem formulation

Our goal is to monitor a predefined repetitive workflow. We describe a workflow as a sequence of defined tasks that have to be executed in some order, which is however not strict; i.e., permutations are allowed. A task is a sequence of observations that corresponds to a physical action such as "pick up object and place it somewhere".

Let $I_t \in \Re^{n\times m}$ be the grey-scale image at time t. Given an image sequence $\mathcal{I} = \{I_0, \ldots, I_t\}$ and a set of $L+1$ possible tasks, $\mathcal{L}^{\#} = \{1, \ldots, L, \#\}$, where # corresponds to a task not related to the workflow (void), we want to associate a task $l_t^{\star} \in \mathcal{L}^{\#}$ with each image I_t at time t, using past and present measurements. This can be seen as a temporal $L+1$ class classification problem.

In the case of C different cameras, the fusion problem can be stated in a similar way: the difference lies in the number of given image sequences $\mathcal{I}_c = \{I_{c,0}, \ldots, I_{c,t}\}, 0 < c < C$.

3. Scene representation

Features extracted from the raw pixel values should be discriminative enough to capture relevant changes with respect to the tasks but at the same time be invariant to irrelevant variations. A wide variety of different features have been proposed over the years, representing image appearance, shape, or motion. Motivated by the fact that the workflow consists of object interactions, we use motion as our primary feature cue. To encode it robustly we use *local motion classifiers (LMCs)* (Adam, Rivlin, Shimshoni, & Reinitz, 2008; Veres et al., 2010).

An LMC $M^{(x,y)}$ observes a position (x, y) and the surrounding $(n\times m)$ pixel neighbourhood $\Omega^{(x,y)}$ of the image. The binary output of a motion monitor applied on the image I_t is defined as follows:

$$M^{(x,y)}(I_t) = \begin{cases} 1 & \text{if } \sum_{(i,j)\in\Omega^{(x,y)}} |I_t(i,j) - I_{t-1}(i,j)| > \theta_M \\ -1 & \text{otherwise.} \end{cases} \quad (1)$$

Frame differencing is used to get changes of the image. If the changes are significant (specified by θ_M) within $\Omega^{(x,y)}$, the LMC $M^{(x,y)}$ returns a positive response. The LMCs can be seen as features that extract high-level information from each image.

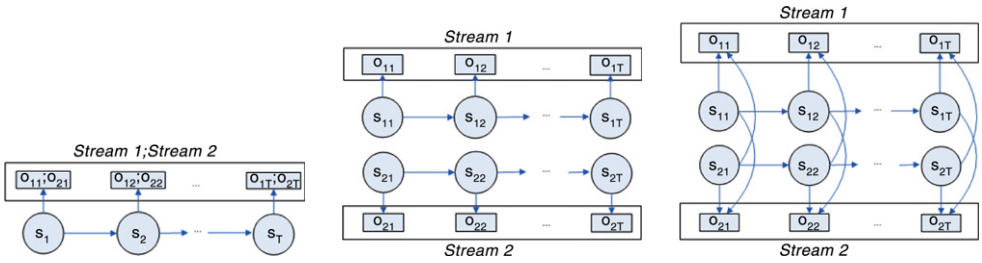
A motion grid is defined as a set of LMCs. We sample an input image by using a fixed overlapping grid. Each grid element corresponds to an LMC. For one time instance t, we concatenate the output of the LMCs within the grid into a vector. The motion grid matrix is used as input for the classifiers:

$$\mathbf{o}_t = [o_t^{(1,1)}, \ldots, o_t^{(x,y)}, \ldots, o_t^{(n,m)}], \quad \text{where}$$

$$o_t^{(x,y)} = \begin{cases} M^{(x,y)} & \text{if } (x,y) \in R_{rel} \\ \text{not used} & \text{otherwise} \end{cases}. \quad (2)$$

The features employed bear similarities to optical flow, since they are based on the sum of pixel differences of the difference image. The difference image itself can be seen as an approximation to the magnitude of the optical flow; however, our method is significantly faster compared, for example, to (Efros et al., 2003). Furthermore, it captures the location information in the image and not in space, as, for example, in (Nguyen et al., 2005), which would probably require object detection. The occlusions do not affect the extracted features, as long as they create consistent patterns.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001



(a) Feature fusion. (b) Parallel fusion. (c) Multistream fusion.

Fig. 3. Various fusion schemes using the HMM framework for two streams: feature fusion (mere concatenation of observation vectors), parallel fusion (independent individual streams), and multistream fusion (cross-coupling between the streams).

4. HMM-based multicamera fusion

The HMM is a very flexible framework that can be tailored to the needs of several applications, one of them being the fusion of multiple streams. An HMM entails a Markov chain comprising N states, with each state being coupled with an observation emission distribution. An HMM defines a set of initial probabilities $\{\pi_k\}_{k=1}^{N}$ for each state, and a matrix $\boldsymbol{A}$ of transition probabilities between the states; each state $\mathbf{s}_t$ is associated with an emitted observation $\mathbf{o}_t$. Gaussian mixture models are typically used for modelling the observation emission densities of the HMM hidden states. However, they are well known to be highly intolerant to the presence of untypical data within the fitting datasets used for their estimation. Finite Student's t-mixture models have recently emerged as a heavier-tailed, robust alternative to Gaussian mixture models, ensuring higher tolerance to outliers. Since our data contain outliers, we use Student's t-distribution as the observation model for the HMM (details of this approach can be found in our previous work (Chatzis et al., 2009)).

In a multicamera set-up, the goal of fusion is to achieve behaviour recognition results that are better than the results that we could attain by using the information obtained by the individual data streams. We will briefly mention in the following some representative approaches.

Among existing approaches, *feature fusion* (Fig. 3(a)) is the simplest; it assumes that the observation streams are synchronous. For streams from C cameras and respective observations at time t given by $\boldsymbol{o}_{1t}, \ldots, \boldsymbol{o}_{Ct}$, the proposed scheme defines the full observation vector as a simple concatenation of the individual observations: $\boldsymbol{o}_t = \{\boldsymbol{o}_{ct}\}_{c=1}^{C}$.

The *parallel HMM* (Fig. 3(b)) assumes that the streams are independent of each other. It can be applied to cameras that may not be synchronised and may operate at different acquisition rates. Each stream c may have its own weight r_c depending on the reliability of the source. Classification is performed by selecting the class $\hat{l}$ that maximises the weighted sum of the classification probabilities from the streamwise HMMs.

Finally, the *multistream fused HMM* approach (Fig. 3(c)) (Zeng et al., 2008) assumes cross-coupling between the streams and is able to capture their interdependencies, by maximising entropy and mutual information criteria. The interstream state–observation dependencies are generally modelled as Gaussian mixture models. Similar to the case of parallel HMMs, the class that maximises the weighted sum of the log likelihoods over the streamwise models is the winner.

5. Segmentation of continuous streams

To solve the time-segmentation problem, which may undermine the utility of our method in real applications, we propose a method to fully automate time segmentation. The key observations that enable a solution in our context are: (i) the tasks are sequential but their order may vary; (ii) each task is executed only once; (iii) the tasks have a variable duration; however, the durations of the same tasks are statistically similar; and (iv) each task ends with placing a part on the welding cell. Based on the above observations, we propose to create a model of the visible actions that signify the termination of each of the tasks. To this end, and assuming a single camera, at this stage we can use an HMM which is trained to recognise the termination of a task in a fixed time window W. The training is effected including sequences of the same duration for all tasks.

In the recognition phase, we employ the HMM that we trained to recognise endings of tasks. The input sequence is defined by a sliding time window of constant duration W, which includes the current frame and all the previous frames that fit in that window. We calculate the probability $p(\mathbf{o}_t, \mathbf{o}_{t-1}, \ldots, \mathbf{o}_{t-W+1}|\lambda)$ that the sequence of the last W observations signifies the end of the task using a standard forward–backward procedure (Rabiner, 1989) ($\mathbf{o}_t$ is the observation vector at time t and λ is the trained HMM model).

Furthermore, we use prior information to model the duration d of all tasks using a Gaussian mixture and we represent it as $p(d|t - t_0)$, where t_0 is the time when the previous task ended (or equivalently when the current task began). Each component of the mixture is a Gaussian probability distribution function (pdf) of the durations of a specific task. Given that each task is executed only once, it is reasonable to remove the components corresponding to tasks that are recognised as finished.

The overall probability that the current task is finished in time t is thus given by

$$p(e_t) = p(\mathbf{o}_t, \mathbf{o}_{t-1}, \ldots, \mathbf{o}_{t-W+1}|\lambda) \cdot p(d|t - t_0). \tag{3}$$

Whenever the above quantity reaches a local maximum, which is above an experimentally defined threshold, we assume that the sequence should be segmented. The observation vectors corresponding to the automatically segmented tasks are buffered and used as input to the task-specific HMMs, which will perform the classification as soon as a new task is segmented. Eq. (3) can be intuitively extended to incorporate observations from several cameras, for example, using the fusion schemes described above.

6. The GA–HMM for task sequence recognition

As may happen in many workflow cases, in our application scenario the execution of one task prohibits the reappearance of the same task until the workflow is over. In this section, we present a method for how to identify the task sequence by using prior information without relying on the Markovian assumption. For this we consider the following probabilities (when having K tasks and $1 \leq i \leq K$): (a) $\log(task(i))$: the log probability distribution for the ith task in the task sequence (i.e., the task that appears at ith order), (b) $\log(task(i)/task(i-1))$: the log probability distribution for the ith task given its previous task. (a) is the output of the task-related HMMs, while (b) can be learnt during training and stored as a priori information.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001

Given the above, and assuming that up to now we have k tasks that have been executed in the workflow, we can create an objective function, which represents the log probability of the task sequence, as follows:

$$E = \log p(task(1), \ldots, task(k)) = w_1 \sum_{i=1}^{k} \log p(task(i)) + w_2 \sum_{i=2}^{k} \log p(task(i)|task(i-1)), \quad (4)$$

where w_1 and w_2 are constants associated to the weight of each term.

It is therefore possible to evaluate the different task permutations according to (4) and select the one with the highest value. However, the number of task permutations is given by $K!/(K-k)!$ if we assume that no task repetitions are allowed. For $K = 7$, for the total sequence we may have 5040 cases to evaluate; for $K = 20$, there are more than 2.43×10^{18} cases; so this approach is not scalable.

We propose a genetic algorithm to find a suboptimal, yet tractable, solution using as objective function the above formula for E. Therefore, we will refer to the proposed method as the GA–HMM approach. Given the number of tasks k executed so far, we define the vector $S = [task(1), \ldots, task(k)]$ which encodes the task sequence so far. We allow the following operations: (a) *mutation*: $task(i)$ randomly changes its value to another value $task(i)'$, provided that $task(i)'$ does not belong to S, and (b) *crossover*: if we have two solutions $S1, S2$, we select a $task1(i)$, $task2(j)$ so that $task1(i) \in S1$ and $task1(i) \notin S2$, $task2(i) \notin S1$ and $task2(i) \in S2$; then we interchange their values in $S1, S2$. This single-value crossover generalises for subsets of values.

By defining the mutation and the crossover probability, we are able to obtain after a sequential evaluation a set of solutions that will locally maximise E. For one task,the solution is simply given by the log probability of a single task as provided by an HMM. As the number of segmented tasks increases by one, some initial solutions have to be created, and the dimensionality of the solution increases as well. For fast convergence, we use the best estimations of the previous step complemented by some random next tasks which did not appear in the solution vector. For example, if a solution for the task assignment of three tasks was $S = [1\ 3\ 2]$, some possible initial solutions for the assignment of four tasks based on that previous step would be $S = [1\ 3\ 2\ 4]$, $[1\ 3\ 2\ 5]$, $[1\ 3\ 2\ 6]$, $[1\ 3\ 2\ 7]$, etc. We need to mention here that (4) can be easily modified to handle fusion. More specifically, the first summation term can be replaced by the associated terms for each separate stream. Another way is to consider that this term is the result of one of the fusion methods mentioned in Section 4.

7. ESN-based multicamera approach

The ESN is a method for online time-series analysis which does not rely on the Markovian assumption. The hidden layer (reservoir) consists of N randomly connected neurons. There are neurons which are connected to cycles, so that past states "echo" in the reservoir. The neurons within the hidden layer are also randomly connected to the k-dimensional input signal, which drives the network. Additionally, only output weights are adapted and learnt; all other weights including feedback are randomly selected and stay static.

Let $\mathbf{o}_t = (o_{1,t}, \ldots, o_{K,t})$ be the input to the network at time t, where K is the dimensionality of a feature vector. Hidden units are $\mathbf{x}_t = (x_{1,t}, \ldots, x_{N,t})$, where N is a number of hidden states, and output units are $\mathbf{y}_t = (y_{1,t}, \ldots, y_{L,t})$, where L corresponds to number of tasks in a workflow here. Further, let $\mathbf{W}^{in}_{N\times K}$ be the weights for the input–hidden connection, $\mathbf{W}_{N\times N}$ be the weights for the hidden–hidden connections, $\mathbf{W}^{back}_{N\times L}$ be the weights for the output–hidden connection, and $\mathbf{W}^{out}_{L\times(K+N+L)}$ be the weights for the read-out neurons, i.e., the connection from all units to the respective read-out neurons. The activations of internal and output units are updated at every time step by $\mathbf{x}_t = f(\mathbf{W}^{in}\mathbf{o}_t + \mathbf{W}\mathbf{x}_{t-1} + \mathbf{W}^{back}\mathbf{y}_{t-1})$, where $f = (f_1, \ldots, f_N)$ are the hidden unit's activation functions. The outputs are calculated as $\mathbf{y}_t = f^{out}(\mathbf{W}^{out}[\mathbf{o}_t, \mathbf{x}_t, \mathbf{y}_{t-1}])$, where $f^{out} = (f_1^{out}, \ldots, f_L^{out})$ are the output unit's activation functions. The term $[\mathbf{o}_t, \mathbf{x}_t, \mathbf{y}_{t-1}]$ is the concatenation of the input, hidden, and previous output activation vectors.

For use on test sequences, the trained network can be driven by new input sequences, and the output is computed as $\hat{\mathbf{y}}_t = \mathbf{W}^{out}\mathbf{x}_t$.

Since the individual read-out neurons are trained independently and usually from highly unbalanced data, we propose normalising the response with respect to their mean responses $\bar{\mathbf{y}}$, calculated on the training data. To identify a task for each time instance the significant maximum is taken:

$$\hat{y}_t = \begin{cases} l^\star & \text{if } \dfrac{\max\limits_{l=1,\ldots,L+1} \mathbf{y}_t(l) - \bar{\mathbf{y}}(l)}{\max\limits_{\substack{l'=1,\ldots,L+1 \\ l^\star \neq l'}} \mathbf{y}_t(l') - \bar{\mathbf{y}}(l')} > \theta_L \\ L+1 & \text{otherwise,} \end{cases}$$

$$\text{where } l^\star = \operatorname*{argmax}_{l=1,\ldots,L+1} \mathbf{y}_t(l) - \bar{\mathbf{y}}(l). \quad (5)$$

In other words, the maximum of the $L+1$ outputs is considered to be significant if the ratio to the second highest value is above some defined threshold θ_L. This threshold influences the precision of our method, and it is set up manually by trial and error during training stage.

Regarding fusion methods, *feature fusion* is achieved by concatenating the scene descriptors for each camera view. As in the HMM case, the observation streams either have to be synchronous or some synchronisation procedure should be applied to these streams. Then an ESN is trained using $\mathbf{y}_t = f^{out}(\mathbf{W}^{out}[[\mathbf{o}_t^1, \ldots, \mathbf{o}_t^{K_n}], \mathbf{x}_t, \mathbf{y}_{t-1}])$, where K_n is the number of cameras. *Parallel fusion* is performed by weighted summing of the respective ESN outputs for all streams, and the result is taken as a label.

8. Experiments

To verify experimentally the applicability of the described methods in real world, we have acquired very challenging videos from the production line of a major automobile manufacturer.[2] Each day the same workflow is performed many times in the production line. Two partially overlapping views were used.

8.1. Experimental set-up

We recorded approximately 8 h of video from a single working cell (including gaps between workflows and breaks). The dataset was captured by two PTZ (pan–tilt–zoom) cameras. We recorded data at 25 fps with relative jitter bounded by 1.6% on the frame rate with resolution of 704×576 pixels. The workspace configuration and the camera positions are shown in Fig. 4. According to the manufacturing requirements, each workflow consists of the following six tasks, which are not necessarily executed sequentially.

Task 1: A part from Rack 1 (upper) is placed on the welding spot by worker(s).
Task 2: A part from Rack 2 is placed on the welding spot by worker(s).
Task 3: A part from Rack 3 is placed on the welding spot by worker(s).
Task 4: Two parts from Rack 4 are placed on the welding spot by worker(s).

[2] The dataset is publicly available through http://www.scovis.eu/.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001

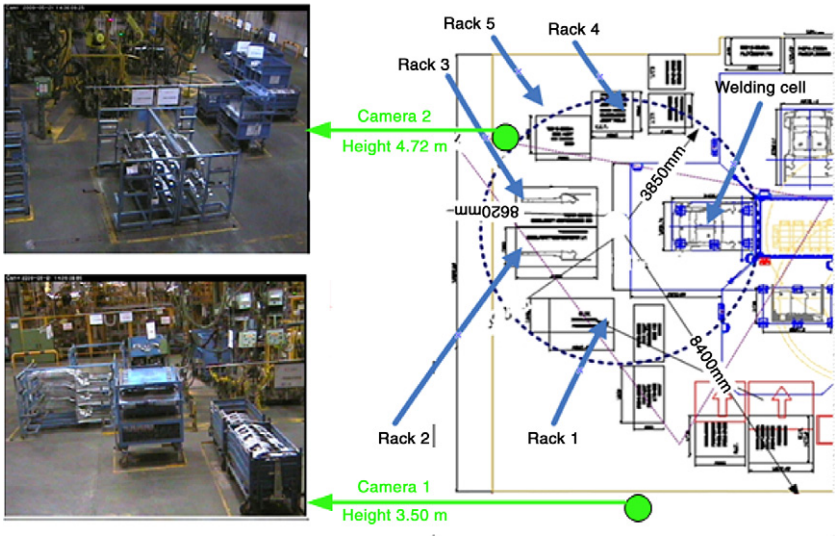


Fig. 4. Schematic representation of the car assembly environment indicating the positions of the racks, the welding cells, and the cameras.

Task 5: A part from Rack 1 (lower) is placed on the welding spot by worker(s).
Task 6: A part from Rack 5 is placed on the welding spot by worker(s).

Moreover, we introduce task 7, which is essential for continuous time-series modelling:

Task 7: Any frame in which no actions from tasks 1–6 take place (void).

The order in which the tasks are executed is not purely random: there are loose patterns and dependencies, which are incorporated in the framework through the approach described in Section 6. Given that most tasks start and end in the welding area, it is difficult to identify, sometimes even by eye, which task the frame belongs to at the start/end of the task. We have annotated being aware that the labelling accuracy is approximately five frames. Moreover, some tasks can have overlapping paths for a number of frames. In addition, there are tasks that bear great visual resemblance, e.g., tasks 1 and 5. In the workflow, the duration of the tasks is different, while the duration of the same task changes from one instance to another. All of these difficulties, together with the severe occlusions in the car assembly environment, present challenges to workflow monitoring and task recognition.

8.2. Implementation details

Here, we give specific implementation details of our methods, which allow reproduction of our results.

Local motion classifiers. The initial motion grid matrix was calculated for 140 patches overlaid onto the whole image. The size of patches (local motion region) was selected as $\Omega = 100 \times 100$ with an overlap of 0.5, i.e. 50% of each patch. The activation motion threshold is set to $\theta_M = 250$.

Prior knowledge. The human operator manually specifies the region where the workflow instances can potentially take place, including the welding machine for each camera. In fact, 65 LMCs in the top half of the image were selected for Camera 1 and 106 LMCs for Camera 2, which form the LMCs for each frame. Our approach would work without this manual definition; however, we would have unrelated motion in the scene, which would require a significantly larger training set. Furthermore, the prior knowledge acquisition concerns learning of the pdfs $p(d|t - t_0)$, as well as of $p(task(i)|task(i-1))$ for all tasks. The learning is effected through the available training samples in each cross-validation cycle.

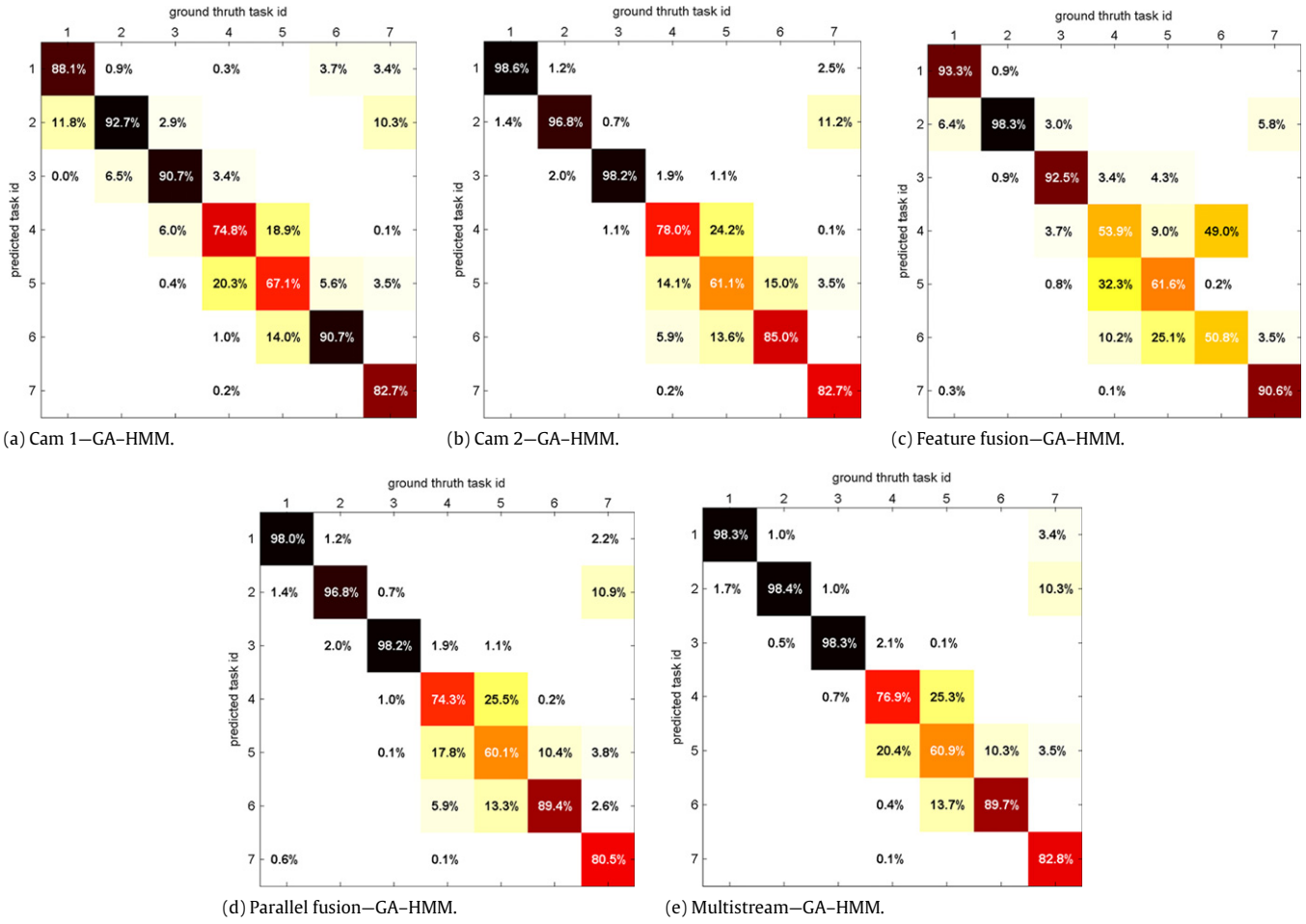
Classification of segmented sequences. For each of the seven tasks, a dedicated three-state HMM with a single component per state was trained (for each camera stream). For the mixture model representing the interstream interactions in the context of the multistream fused HMM, we use mixture models of two component distributions. These experiments were conducted for the case of individual stream HMMs, as well as feature fusion, parallel, and multistream fused HMMs. In all cases, we trained our models using the expectation–maximisation (EM) algorithm, and we used the multivariate Student's t-distribution as the observation model of the HMMs.

Sequence segmentation. A two-state HMM with six mixture components per state was trained to model the ending of tasks (1–6), as described in Section 5. Based on the statistics of the dataset, we selected the time window W, which signifies the end of a task, to equal 50. Whenever a new task was automatically segmented, it was tested in each of the six models to acquire the related log probability, which in combination to the prior knowledge provided the input to the genetic algorithm. Regarding task 7 (void), every time a new task was segmented, we checked the forthcoming sequence through temporal windows to determine whether it corresponded to the void task; this was practically recognised when the log probability surpassed an experimentally set threshold.

Echo state network. We applied a plain ESN with 3000 hidden units, 65 inputs, and 7 outputs for Camera 1 and a plain ESN with 3000 hidden units, 106 inputs, and 7 outputs for Camera 2 to obtain single stream results. The number of states was selected to achieve a trade-off between the training time and generative properties of the trained ESN based on experimental runs by changing number of hidden states from 100 to 12 000. Although increasing the number of hidden states to 10 000 can improve the ESN results by between 1% and 6% on average for a given instance of workflow, the training time could be increased by an order of two. We used the ESN toolbox written in MATLAB by Jaeger et al..[3] The spectral radius was $|\lambda| = 0.98$, and the input scaling and teacher scaling (Jaeger, 2001) were chosen as 0.1 and 0.3, respectively. Furthermore, additional noise was added to the ESN during the training process to improve the stability. Median filtering with a filter length of 51 was performed at the post-processing stage, i.e., on predicted labels.

[3] http://www.reservoir-computing.org/node/129,2009/08/05.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001



(a) Cam 1—GA–HMM. (b) Cam 2—GA–HMM. (c) Feature fusion—GA–HMM.

(d) Parallel fusion—GA–HMM. (e) Multistream—GA–HMM.

Fig. 5. Confusion matrices in the GA–HMM approach: individual camera streams, feature fusion, parallel fusion, and the multistream fused HMM (attaining the best performance).

For feature fusion, we trained the ESN with 3500 hidden units, 171 inputs, and 7 outputs, where 171 inputs (features) consist of concatenation of features for Camera 1 and Camera 2. The parallel fusion was achieved by weighted summing of the corresponding outputs of two ESNs and selecting the output with maximum value as a label for a given frame.

8.3. Evaluation and results

In the dataset, 20 instances of workflow and their tasks were manually labelled. Each workflow instance consists of 3550 to 9100 frames, which correspond to 2–5 min. The LMCs were calculated in real time (20–25 fps). Four-fold cross-validation was performed, with testing sets including scenarios 1–5, 6–10, 11–15, and 16–20, while the remaining instances of workflow each time were used for training. ESN training took approximately 15 min using the MATLAB implementation on a 2.83 GHz computer running Windows Vista. However, testing could be done very efficiently online at 20 fps. HMM training was performed in approximately 2 min, and testing of a sequence of 1000 vectors, for example, in less than 10 s.

For a quantitative evaluation, we used recall–precision measurements. Recall corresponds to the correct classification rate (number of true positives divided by total number of positives in ground truth), whereas precision relates to the trust in a classification (number of true positives divided by number of true and false positives). The F-measure is the harmonic mean of these two measurements.

In the following we present how both GA–HMM-based and ESN-based approaches coped with task recognition using single stream data from Camera 1 and Camera 2, respectively, and by fusing features and labels from both views. Additionally, multistream fusion was examined for the GA–HMM-based approach.

Regarding sequence segmentation, the results showed that the method proposed in Section 5 led to segmentation times rather close to the ones in ground truth. In particular, the mean and standard deviation values of the absolute difference (in frames) between the estimated segmentation time and the ground truth task end time averaged across all tasks of all testing sequences were as follows. Camera 1: 16.8 ± 12.9, Camera 2: 11.5 ± 10.3, feature fusion: 19.1 ± 16.8, parallel fusion: 11.3 ± 10.2, and multistream fusion: 10.1 ± 8.1. Segmentation from Camera 2 was more successful than that from Camera 1, which can be explained by the former's generally better viewpoint to the welding cell, where all task endings take place. Moreover, multistream fusion provided a small but notable improvement, by decreasing the average absolute error in relation to ground truth to merely 10 frames.

The results shown in Table 1 indicate that the proposed GA–HMM approach using LMC features attained very good recognition rates. Camera 2's individual stream yielded better results than Camera 1, since the former offers a generally better viewpoint. Feature fusion and parallel fusion achieved rates lower than and roughly equal to the best single stream, respectively. The multistream fused HMM-based approach provided the maximum recall, precision, and F-measure, as it succeeded in effectively capturing the interdependencies between the two streams. The confusion matrices in Fig. 5 also reflect the superiority of the multistream method, but also indicate that for tasks whose order

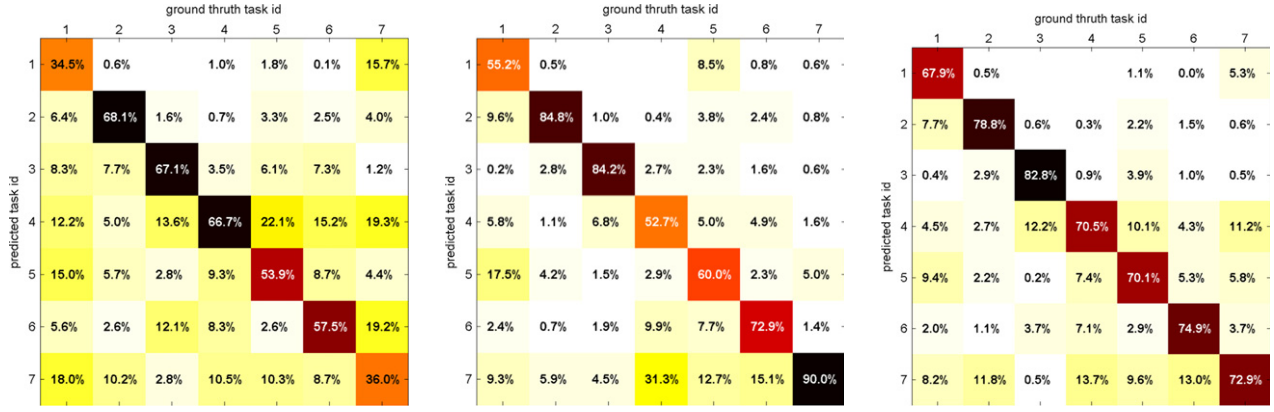
Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001

Table 1
Performance of GA–HMM for all 20 testing workflow instances—Student's t-distribution.

	Camera 1 (%)	Camera 2 (%)	Feature fusion (%)	Parallel fusion (%)	Multistream (%)
Recall	86.0 ± 22.5	89.8 ± 12.5	75.5 ± 24.1	89.9 ± 11.8	90.3 ± 12.0
Precision	85.6 ± 22.4	90.0 ± 12.5	74.8 ± 24.2	90.0 ± 11.9	90.4 ± 12.1
F-measure	85.8 ± 22.4	89.9 ± 12.5	75.1 ± 24.1	89.9 ± 11.8	90.3 ± 12.1

Table 2
Performance of ESN for all 20 testing workflow instances.

	Camera 1 (%)	Camera 2 (%)	Feature fusion (%)	Parallel fusion (%)
Recall	59.3 ± 13	73.5 ± 8.4	74.5 ± 11	76.2 ± 10.3
Precision	55.6 ± 13	72.8 ± 8.7	74.3 ± 12	76.3 ± 11.2
F-measure	57.3 ± 13	73.1 ± 8.5	74.3 ± 11.5	76.2 ± 10.6



(a) Cam 1—ESN. (b) Cam 2—ESN. (c) Parallel—ESN.

Fig. 6. Confusion matrices in the ESN approach—individual camera streams and the parallel fusion scheme (attaining the best results for ESN).

is more statistically variant (i.e., tasks 4, 5, 6) the recognition rates were lower.

As far as the ESN-based approach is concerned, it shows significantly better results for Camera 2 in comparison to Camera 1 (Table 2) due to the significantly lower level of occlusions. Combining these two single streams by feature or parallel fusion allows us to achieve average recall of 74.5% and 76.2%, respectively, with parallel fusion slightly outperforming feature fusion.

Confusion matrices (Fig. 6) indicate that the most difficult task for monitoring was task 1, with correct classification rates (CCRs) of 34.5% and 55.2% for single stream cases. Using parallel fusion allowed us to increase the CCR for this task to 67.9%. Although this task is well separated from the manufacturing requirements point of view, it is not that easy to distinguish it from other tasks using the video recordings, since it shares the same paths as other tasks for some periods of time.

In this application, GA–HMM outperformed ESN according to recall, precision, and F-measures on 20 instances of workflow. This was probably due to the fact that the GA–HMM structure makes it possible to encode the task sequences in a hierarchical fashion using application-specific prior knowledge. On the other hand, the ESN is a general-purpose classifier, which seeks to capture patterns in observation sequences; the prior information is captured only implicitly and requires a large number of nodes. Recently it was demonstrated that despite the fact that ESNs are not based on the Markovian property, in practice they are influenced more by recent states, which of course makes the memorisation of series of past tasks quite hard (Gallicchio & Micheli, 2011). In contrast, the GA–HMM has no such problem because it explicitly encodes the whole task history.

9. Conclusion

In this paper, we have addressed the issue of online recognition of visual tasks and workflows in complex industrial environments. To this end the employment of holistic features based on a grid time matrix so as to bypass the challenging tasks of detection and tracking, which are usually unsuccessful in such environments, leads to a very satisfactory representation. We proposed the GA–HMM, which is an HMM endowed with a method to automatically segment the input stream and to exploit prior knowledge through a genetic algorithm. By doing so, we could take advantage of the versatile HMM architecture, for example, by incorporating elaborate fusion methods (Zeng et al., 2008) or robust models (Chatzis et al., 2009) for online stream classification. We scrutinised the effectiveness of this approach and compared it to an ESN-based approach. The GA–HMM approach outperformed the ESN, although the latter's performance is influenced by the topological complexity and consequently the training time required. The ESN offers a simpler, more straightforward approach, which can yield satisfactory results when the training time can be compromised. A plus of the ESN is the automated segmentation of sequences, while GA–HMM relies on the ability to detect the task segments. Neither approach depends on the Markovian assumption to extract the sequences of tasks. However, as was recently shown, the ESN is practically influenced more by the most recent observations, so it is naturally expected to have more difficulties in classifying long sequences of tasks.

Fusion of multiple camera streams provided added value in many cases. Between the fusion methods employed for both GA–HMM and ESN, the parallel fusion method exploited the

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001

redundancies between the different streams more effectively compared to feature fusion. The latter method assumes strict synchronisation, which is not the case in our setting. The benefits of fusion were more apparent in the ESN, where there was bigger room for improvement. Finally, the GA–HMM based on the multistream fused HMM could better capture interdependencies between streams and led to the highest recognition rates among all approaches.

Finally, the proposed method can be easily employed in other workflows, simply by modifying the constraints of the solution given by the genetic algorithm accordingly, for example by allowing repetitions of tasks, omissions, etc. It is also scalable, because the underlying fusion methods are not limited by the number of streams.

Acknowledgement

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no ICT-216465.

References

Adam, A., Rivlin, E., Shimshoni, I., & Reinitz, D. (2008). Robust real-time unusual event detection using multiple fixed-location monitors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(3), 555–560.

Antonakaki, P., Kosmopoulos, D., & Perantonis, S. (2009). Detecting abnormal human behaviour using multiple cameras. *Signal Processing*, *89*(9), 1723–1738.

Boiman, O., & Irani, M. (2005). Detecting irregularities in images and in video. In *Proceedings of the IEEE international conference on computer vision. ICCV 2005. Vol. 1* (pp. 462–469).

Breitenstein, M., Grabner, H., & Gool, L. V. (2009). Hunting nessie: real time abnormality detection from webcams. In *Proceedings of the IEEE international conference on computer vision workshops. ICCV workshops 2009* (pp. 1243–1250).

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

Dupont, S., & Luettin, J. (2000). Audio-visual speech modeling for continuous speech recognition. *IEEE Transactions on Multimedia*, *2*(3), 141–151.

Efros, A. A., Berg, A. C., Mori, G., & Malik, J. (2003). Recognizing action at a distance. In *Proceedings of the IEEE international conference on computer vision. ICCV 2003* (pp. 726–733).

Felzenszwalb, P., McAllester, D., & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE conference on computer vision and pattern recognition. CVPR 2008* (pp. 1–8).

Fine, S., Singer, Y., & Tishby, N. (1998). The hierarchical hidden Markov model: analysis and applications. *Machine Learning*, *32*(1), 41–62.

Gallicchio, C., & Micheli, A. (2011). Architectural and Markovian factors of echo state networks. *Neural Networks*, *24*(5), 440–456.

Grabner, H., & Bischof, H. (2006). On-line boosting and vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition. CVPR 2006. Vol. 1* (pp. 260–267).

Hellbach, S., Strauss, S., Eggert, J., Komer, E., & Gross, H. -M. (2008). Echo state networks for online prediction of movement data—comparing investigations. In *Proceedings of the international conference on artificial neural networks. ICANN 2008. Vol. 5163* (pp. 710–719).

Huang, C., Wu, B., & Nevatia, R. (2008). Robust object tracking by hierarchical association of detection responses. In *Proceedings of the European conference on computer vision. ECCV 2008* (pp. 788–801).

Ivanov, Y. A., & Bobick, A. F. (2000). Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 852–872.

Jaeger, H. (2001). The echo state approach to analysing and training recurrent neural networks. *Technical Report GMD Report 148*. German National Research Institute for Computer Science.

Johnson, N., & Hogg, D. (1996). Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, *14*(8), 609–615.

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

Laptev, I., & Perez, P. (2007). Retrieving actions in movies. In *Proceedings of the IEEE international conference on computer vision. ICCV 2007*.

Lv, F., & Nevatia, R. (2006). Recognition and segmentation of 3-d human action using HMM and multi-class adaboost. In *Proceedings of the European conference on computer vision. ECCV 2006. Vol. IV* (pp. 359–372).

Nguyen, N., Phung, D., Venkatesh, S., & Bui, H. (2005). Learning and detecting activities from movement trajectories using the hierarchical hidden Markov model. In *Proceedings of the IEEE conference on computer vision and pattern recognition. CVPR 2005. Vol. 2* (pp. 955–960).

Oliver, N., Garg, A., & Horvitz, E. (2004). Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, *96*(2), 163–180.

Padoy, N., Mateus, D., Weinland, D., Berger, M., & Navab, N. (2009). Workflow monitoring based on 3d motion features. In *Proceedings of the IEEE international conference on computer vision workshops. ICCV workshops 2009* (pp. 585–592).

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

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

Scherer, S., Oubbati, M., Schwenker, F., & Palm, G. (2008). Real-time emotion recognition from speech using echo state networks. In *Artificial neural networks in pattern recognition. Vol. 5064* (pp. 205–216).

Shi, Y., Huang, Y., Minnen, D., Bobick, A., & Essa, I. (2004). Propagation networks for recognition of partially ordered sequential action. In *Proceedings of the IEEE conference on computer vision and pattern recognition. CVPR 2004. Vol. 2* (pp. 862–869).

Skowronski, M., & Harris, J. (2007). Automatic speech recognition using a predictive echo state network classifier. *Neural Networks*, *20*, 414–423.

Turaga, P., Chellappa, R., Subrahmanian, V., & Udrea, O. (2008). Machine recognition of human activities: a survey. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*(11), 1473–1488.

Veres, G., Grabner, H., Middleton, L., & Gool, L. V. (2010). Automatic workflow monitoring in industrial environments. In *Proceedings of the Asian conference on computer vision. ACCV 2010* (pp. 200–213).

Verplancke, T., Looy, S. V., Steurbaut, K., Benoit, D., Turck, F. D., Moor, G. D., et al. (2010). A novel time series analysis approach for prediction of dialysis in critically ill patients using echo-state networks. *BMC Medical Informatics and Decision Making*, *10*(1), 414–423.

Vogler, C., & Metaxas, D. (1999). Parallel hidden Markov models for American sign language recognition. In *Proceedings of the IEEE international conference on computer vision. ICCV 1999* (pp. 116–122).

Voulodimos, A., Grabner, H., Kosmopoulos, D., Van Gool, L., & Varvarigou, T. (2010). Robust workflow recognition using holistic features and outlier-tolerant fused hidden Markov models. In *Proceedings of the international conference on artificial neural networks. ICANN 2010. Vol. 6352* (pp. 551–560).

Wang, X., Ma, K., Ng, G., & Grimson, W. (2008). Trajectory analysis and semantic region modeling using a nonparametric Bayesian model. In *Proceedings of the IEEE conference on computer vision and pattern recognition. CVPR 2008* (pp. 1–8).

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

Zelnik-Manor, L., & Irani, M. (2006). Statistical analysis of dynamic actions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(9), 1530–1535.

Zeng, Z., Tu, J., Pianfetti, B., & Huang, T. (2008). Audiovisual affective expression recognition through multistream fused HMM. *IEEE Transactions on Multimedia*, *10*(4), 570–577.

Please cite this article in press as: Voulodimos, A., et al. Online classification of visual tasks for industrial workflow monitoring. Neural Networks (2011), doi:10.1016/j.neunet.2011.06.001