SEMI-SUPERVISED DEEP LEARNING FOR OBJECT TRACKING AND CLASSIFICATION

*Nikolaos Doulamis and Anastasios Doulamis**

National Technical University of Athens, Heroon Polytechniou st. 15773 Athens, Greece,
email {ndoulam@cs.ntua.gr, adoulam@cs.ntua.gr}

ABSTRACT

A semi-supervised deep learning paradigm is proposed for object classification/tracking. The method addresses the main difficulties of deep learning, by allowing unsupervised data to initially configure the network and then a gradient descent optimization scheme is triggered to fine tune the data. Unsupervised learning transforms the input data into smaller and more abstract forms of representations and therefore improves the stability, convergence and performance of the model. Additionally, an adaptive approach is presented in a way to allow dynamic modification of the model to the current visual conditions. Adaptation is performed by exploiting both unsupervised and supervised samples, coming by the application of a combined motion/deep learning tracker activating only at frames a decision mechanisms ascertains retraining.

Index Terms— Object tracking, classification, deep networks

1. INTRODUCTION

Human brain does not work by explicitly pre-processing sensory signals but rather allows them to propagate into complex hierarchies [1], [2], implying the so called "deep learning" paradigm; methods that process and model data using multi-layers structural representations [3]. Training deep multi-layered neural network structures is in fact an arduous process since the standard learning strategies (e.g., backpropagation) converge to poor solutions for networks of more than three hidden layers [4]. The problem mainly stems from the following reasons. (i) As we add more and more hidden layers in a neural network architecture, the number of its parameters (weights) needed to be learnt during training increases, implying that we need large number of labeled data to avoid over-fitting. (ii) The gradient descent-based optimization algorithm used to estimate the network weights is often trapped to local minima, deteriorating the classifier performance. (iii) Finally, in deep multi-layered neural networks, the last (deep) layers are triggered by input data totally different from the actual sensory inputs since the latter, as they are propagating from one layer to another, they are also transforming from one space to another and noise is added.

To overcome the aforementioned limitations, Hinton et al. [3], [5] propose an *unsupervised learning* as a training method for Deep Belief Networks (DBN's). In *unsupervised*, however, learning, we like to preserve information about the input, while in *supervised tasks* the target is to learn a mapping from the inputs to a good classification. Supervised learning is suitable for object tracking and classification with the limitations mentioned above. However, combining unsupervised with supervised learning, we are able to build more efficient object classification schemes, sine the unsupervised paradigm provides a fine to coarse representation of the input data, which in the sequence improves convergence, and stability and performance of the object tracking and labeling process (supervised learning). This is mainly due to the fact that abstract forms of sensory inputs are propagated to the deep (last) layers of processing, retaining a correlation between the actual inputs and the inputs triggering the deep layers, instead of the conventional supervised approach where such correlation between input-output is lost.

Different models and training strategies have been proposed as deep learning deep structures. Researchers have been focused on DBN's [3],[5], deconvolutional architectures [6], convolutional neural networks [7], [8] non-linear autoencoders or auto-associator networks [9]-[15] and restricted Boltzmann machines (RBMs) [16]-[18].

The key concept behind the aforementioned methods is the unsupervised learning which is not suitable for image analysis problems. The principle of such unsupervised machines is to discover structures within feature data that renders supervised mechanisms or classification tasks, which are necessary for object tracking and labeling. The work of [19] proposes spike-and-slab sparse coding (called S3C) strategy as an efficient feature learning algorithm. Similarly, [20] exploits convolutional restricted Boltzmann machines for shift-invariant feature learning. In the same context, the work of [21] combines unsupervised and supervised components for efficient training deep networks.

In this work, we address such limitations in the context of object tracking and labeling by proposing a semi-supervised deep learning architecture that exploits the unsupervised and supervised learning paradigm for real-time robotic vision applications. Our approach includes adaptation mechanisms that update the network each time a considerable change of the environment is encountered. Approximates of current visual environment are provided

*This research is supported by EU Funded project eVacuate "A holistic, scenario-independent, situation-awareness and guidance system for sustaining the Active Evacuation Route for large crowd" under grant agreement 313161 and JASON "Joint synergistic and integrated use of earth observation, navigation and communication technologies for enhanced border security" project funded under the cooperation program of the Greek Secretary of Research & Technology.

through a dynamic tracker that combines motion and learning features to optimally and automatically construct pairs of representative inputs and desired outputs. Few samples are required to re-train the network, since the majority of the weight parameters are estimated through supervised learning. In deep classifiers, the data are used as inputs to the structures instead of features since the deep connections can find more complex relationships among them and the desired outputs.

2. NOTATION

Let $\mathbf{x} \in R^n$ be an input feature vector of size n. This feature vector has been extracted by analyzing an image I at a region $\Re(\mathbf{p})$ around a pixel $p=(x,y)$. The purpose of an object classification problem, coinciding with a supervised learning task, is to transform the input representation $\mathbf{x}$ into a p-dimensional output vector $\mathbf{y}$, defined as

$$\mathbf{y} = \left[\, p_{\omega_1} \, p_{\omega_2} \cdots p_{\omega_p} \, \right]^T \tag{1}$$

where p_{ω_j} denotes the probability of feature vector $\mathbf{x}$ to belong to the j-th (out of p available) class. Let us now consider that there exists a non-linear vector-valued function $\mathbf{h}(\cdot)$ that models the transformation of $\mathbf{x}$ to the p-dimensional output vector $\mathbf{y}$, that is, $\mathbf{y} = \mathbf{h}(\mathbf{x})$. However, $\mathbf{h}(\cdot)$ is actually unknown.

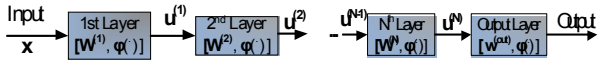

Fig. 1. A multi-layered deep architecture.

Fig. 1 presents a compact representation form of the structure of a deep multi-layered neural network able to approximate any continuous function. Speaking mathematically, let us assume a deep structure of N hidden layers plus one output layer (e.g., totally N+1 layers). Since we have a p-dimensional classification problem, the output layer consists of p neurons [see Eq. (2a)], while the number of neuron at the l-th hidden layer is denoted as m_l, l=1,2,…,N. Variable $m_0 = n$ equals the input features size,

$$\mathbf{y} = \begin{bmatrix} p_{\omega_1} \\ \vdots \\ p_{\omega_p} \end{bmatrix} \approx \boldsymbol{\varphi}(\mathbf{W}^{(out)^T} \cdot \mathbf{u}^{(N)}) \text{ with} \tag{2a}$$

$$\mathbf{u}^{(l)} = \boldsymbol{\varphi}\left(\mathbf{W}^{(l)^T} \cdot \mathbf{u}^{(l-1)} \right), \; l = 1,2,\ldots, N \text{ and } \mathbf{u}^{(0)} = \mathbf{x} \tag{2b}$$

where matrices $\mathbf{W}^{(l)} = [\mathbf{w}_1^{(l)} \; \mathbf{w}_2^{(l)} \cdots \mathbf{w}_{m_l}^{(l)}]$, with $l = 1,2,\ldots, N$ includes the weights of the l-th hidden layer while matrix $\mathbf{W}^{(out)} = [\mathbf{w}_1^{(out)} \; \mathbf{w}_2^{(out)} \cdots \mathbf{w}_p^{(out)}]$ the weights of the output layer.

2.1. Limitations of supervised training in deep networks

Although Eq. (2) is a robust mathematical framework for modeling deep architectures, the main difficulty results from the efficiency of the algorithm used to approximate the unknown coefficients $\mathbf{W}^{(l)}$ with l=1,…,N and $\mathbf{W}^{(out)}$. Minimization of a mean square error criterion is often used to optimize model parameters. Actually, back-propagation performs gradient descent analysis on the error surface by modifying each neural network weight proportionally to the negative gradient of the error surface. In complex non-linear relationships and in case of a considerable number of input variables, there are multiple local minima in the error surface, deteriorating network performance. This is particular evident for more than 3 hidden layers units. In deep learning, the error is accumulated as the inputs are propagated into deep hierarchies and being transformed from one space to another. In such cases, the actual input space at deep layers has been significantly altered from its original data making the training a really arduous task.

To address this problem, in this paper, we propose the mathematical framework of *semi-supervised deep learning by combing an unsupervised* deep learning paradigm with a supervised learning process for object tracking.

3. STACKED AUTOENCODERS TO APPROXOMATE HIDDEN LAYERS RESPONSES

To address the previously difficulties, we forwardly propagate the input data into the hidden transformation layers so that the output neurons receive a structured (abstract) form of the inputs. This way, the gradient-based learning optimization algorithms (e.g., backpropagation) exploit more "reliable" data to perform the mapping from the input to the desired (supervised) prediction. This increases the performance of the network. Instead, letting the weights to be randomly selected, the performance is deteriorated especially for large scale networks.

In particular, let us denote as $\tilde{\mathbf{x}}^{(l)}$, l=1,2,..,N a compressed (abstract) version of input vector $\mathbf{x}$ at the l-th layer of the network. It should be mentioned that the dimension of $\tilde{\mathbf{x}}^{(l)}$ is often smaller than the dimension of $\mathbf{x}$, i.e., of n. Our goal, is to pre-estimate network weights at each hidden layer, so that the output of the hidden activation neurons $\mathbf{u}^{(l)}$ be as close as possible to the compressed forms $\tilde{\mathbf{x}}^{(l)}$. Therefore, we have that the initial weight estimate for the network at the l-th layer $\mathbf{w}^{(l)}(0) = \text{vec}(\mathbf{W}^{(l)}(0))$, is given as

$$\begin{aligned} \hat{\mathbf{w}}^{(l)}(0) &= \arg\min_{\mathbf{w}^{(l)}(0)} L(\mathbf{u}^{(l)}, \tilde{\mathbf{x}}^{(l)}) = \\ &= \arg\min_{\mathbf{w}^{(l)}(0)} L(\boldsymbol{\varphi}(\mathbf{W}^{(l)^T} \cdot \mathbf{u}^{(l-1)}), \tilde{\mathbf{x}}^{(l)}) \end{aligned}, \; l=1,2,..,N \tag{3}$$

where $\hat{\mathbf{w}}^{(l)}(0)$ are the optimal initial estimates of the network weights at the l-th hidden layer.

Two difficulties are involved in solving Eq. (3). One is that the optimal initial estimates $\hat{\mathbf{w}}^{(l)}(0)$ depend on the optimal initial estimates of the previous layers. One way to overcome this is to adopt a greedy layered wise approach to pre-train the deep network [21]; pre-training one layer at a time in a greedy way. Therefore, we can say that $\mathbf{u}^{(l)} \approx \tilde{\mathbf{x}}^{(l-1)}$, while $\tilde{\mathbf{x}}^{(l-1)}$ *have been already estimated at a previous stage of the algorithm*. The second difficulty is that the current compressed signal transformations $\tilde{\mathbf{x}}^{(l)}$ are actually unknown. To overcome this problem, we exploit concepts from autoencoders. An autoencoder is a symmetrical neural network to learn the features of a dataset in an unsupervised manner. This is achieved by minimizing the reconstruction error between input data at the encoding layer and its reconstruction at the decoding layer. This iterative solution can be determined considering that $\tilde{\mathbf{x}}^{(0)} \equiv \mathbf{x}$. Fig. 2 graphically illustrates the proposed stacked autoencoding scheme.

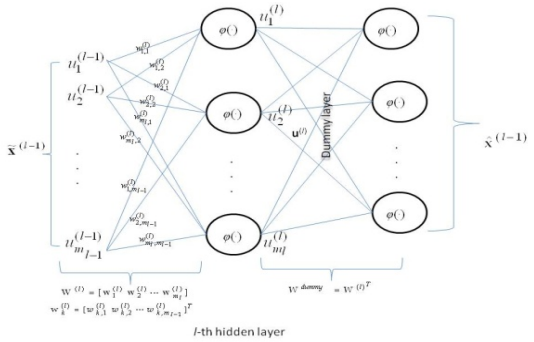

Fig. 2. The stacked autoencoder model that configure the network.

4. ADAPTIVE SEMI-SUPERVISED LEARNING

As regards the initial output weights $\hat{\mathbf{w}}^{(out)}(0)$, these are estimated using a conventional supervised learning approach, such as the backpropagation algorithm. However, in this scenario, we modify only a small number of weight parameters and the convergence of the network can be easily achieved instead of a deep network architecture. One of the difficulties of the aforementioned network structure is that network weights are considered constant throughout its operation. However, in real-life image analysis applications, the conditions that a classifier operates on are of considerable different from the ones it has been trained to; this means that network adaptation is required.

4.1. Decision Making that Ascertains Adaptation

Let us denote as $L(t) = \sum_{i=0}^{Q} L_i(t)$ a labeling mask at a frame t as being provided by the neural network. This labeling mask consists of Q foreground objects, $L_i(t)$ i=1,..,Q. It is clear that as long as the neural network output is adequate no retraining is required; in this case, the foreground masks present consistency both in area size and location. Let $\Lambda_i = \left\| area(L_i(t)) - area(L_i(t+1)) \right\|$ a metric for the area consistency. Additionally, location consistency is calculated based on the intersection between two frames which should not equal the null set. Thus, if $\Lambda_i < T$ and $L_i(t) \cap L_i(t+1) \neq \varnothing$ no retraining is needed.

4.2. Hidden Layer Weight Adaptation – The Unsupervised Paradigm

Each time a new image frame is captured a set of unlabeled data are collected. The DCT coefficients of the MPEG are used as sensory input data. These data are, then, used to update the hidden layer weight parameters. To accelerate time, a greedy layered wised method is adopted for training. The method starts from the first hidden layer and continuous with the remaining ones. A gradient descent method is used to transform the raw input into a vector consisting of activation of the first hidden units. Then, the second layer is trained with the information of the first layer and so on. The collected unlabelled data are initially compared with previous data representatives. Only in case that the new unlabelled data presents high differences with the previous ones, re-training of the stacked autoenders is performed.

4.3. Output Layer Weight Adaptation – The Supervised Paradigm

The difference in this scenario is that we need supervised pairs, which is difficult to be obtained, especially in real-life scenarios. This is approximated using the following method.

4.3.1. Combined motion based and deep learning tracking

A mixture of Gaussian model is used to approximate the background content. Background modeling is based on a new innovative approach that exploits local geometry instead of the conventional methods. The idea is to model a region of pixels instead of one pixel itself. However, such a modelling is practically impossible to be implemented. Thus, the unsupervised deep paradigm of Section 4.2 and 3 is exploited to transform inputs into more relative forms.

$$B(\mathbf{u}^{(l)}(t)) = \sum_{i=1}^{M} w_i(t) \cdot G(\mathbf{u}^{(l)}(t), \boldsymbol{\mu}_i(t), \boldsymbol{\Sigma}_i(t)) \quad (4)$$

where $\mathbf{u}^{(l)} \approx \tilde{\mathbf{x}}^{(l-1)}$ (see Section 3) the approximate of the sensory inputs provided by the stacked autoencoder.

4.3.2. Confident background/foreground labeled data

A constraint time fusion scheme is considered to estimate confident motion regions in an image. Initially, we select "good pixels" for calculating the Lucas Kanade optical flow to accelerate time and remove noise. In case that the motion vectors present no coherency the results of the tracker are not reliable and therefore no supervised pairs are selected. In case that reliable motion information is estimated the complimentary regions, after being expanding by applying a dilation morphology operator, are consider as confident background regions, $B_{conf}(t) = (\psi \to M(t))^c$.

This set produces a respective abstract form, say $\mathbf{u}_B^{(l)}$ when the sensory data of the background are fed to

the stacked autoencoders. Vector $\mathbf{u}_B^{(l)}$ are used to update the model parameters of (4). In this way, we estimate the background pairs representing the current visual environment. The respective foreground samples are estimated through a background subtraction process. In case of more than one foreground objects, the output of the deep learning architecture is exploited on each separate connected component. In case that the deep learning network before the adaptation produces a coherent output the connected component retains its labelling. Otherwise, a new labelling is assigned to the detected mask.

4.3.3. Training the network

Only the last hidden and the output layer is adaptive using the supervised training set. This way, we significantly reduce the number of parameters required for the adaptation. The remaining weight parameters are updated based on an unsupervised learning paradigm (see Section 4.2) using a conventional backpropagation algorithm.

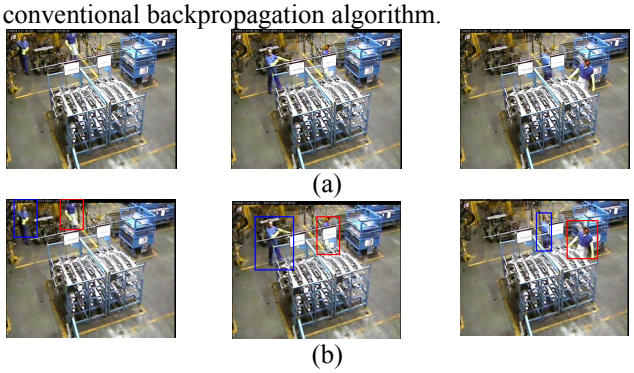

Fig. 3. The results of the proposed semi-supervised deep learning architecture on object tracking and labeling.

5. EXPERIMENTS

The Workflow Recognition dataset consists of video sequences from the production line of a major automobile manufacturer [22], created by the EU Research project SCOVIS [23] has been selected for evaluation. The selected dataset presents many computer vision challenges, such as severe occlusions, background content and luminosity changes. The performance of many known algorithms on this dataset is very weak [24]-[26].

Fig. 3 presents the performance of semi-supervised deep learning for object tracking and classification. The DCT of three color components on 32x32 blocks have been used as input features of neural network. As is observed, the proposed network adequately localize and label the two foreground objects (workers), although the existence of occlusions and background complexity. Finally, Fig. 4 presents a more complicated scenario where a different camera view has been selected for object tracking and labeling. The original frame is shown in Fig. 4(a). The performance of the neural network before the adaptation is shown in Fig. 4(b) where we observe that the network fails to identify the two foreground objects. In this case, network retraining is activated (see Section 4.1). Then, a new current training set is automatically created exploiting the tracker of Section 4.3.1 [see Fig. 4(e)]. Since the performance of the neural network before the adaptation localizes one out of two foreground objects, we assume that the same object labeling should be kept for that object; the other object are labeled as a new one. The performance of the network after retraining is shown in Fig. 4(c), where we notice the improvement performance [see also Fig. 4(d)].

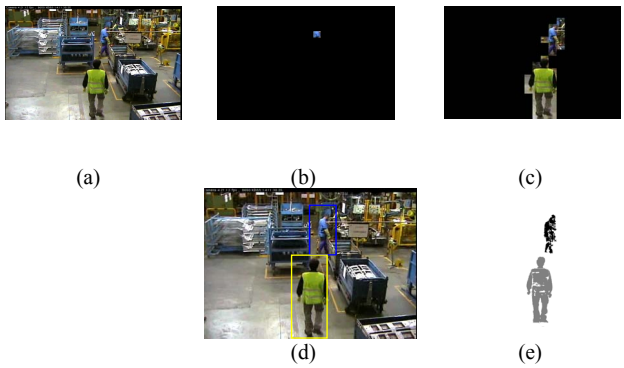

Fig. 4. Evaluation at a different camera view. (a) The original frame. (b,c) The results of the network before and after the adaptation. (d) The labeling results. (e) The training set estimated for the network updating.

Fig. 5 presents the *F1* score versus image frames. In this scenario, we have assumed that the first 10 frames present similar visual properties and thus no re-training is needed. In this figure, the "no adaptation" notion stands for conventional classification-based approaches. As is observed, the proposed method outperforms the conventional approaches especially in cases where alteration of the current visual conditions is encountered.

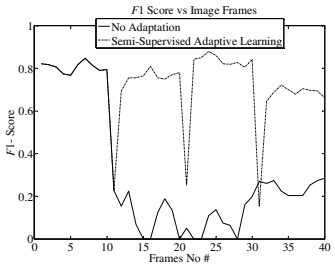

Fig. 5. Objective evaluation of the proposed deep architecture using the *F*1-score and comparison with conventional classifications schemes.

6. CONCLUSIONS

In this paper, we present an adaptive semi-supervised deep learning paradigm for object tracking and classification scenarios. Our model, exploits the principle of unsupervised learning to re-train high multi-layered neural network architectures in order to improve their convergence, stability and performance especially in image analysis problems. The presented scheme is combined with a motion based deep learning tracker that provides only few supervised pairs for network retraining, while the supervised data are used to mainly adapt the network parameters.

7. REFERENCES

[1] I. Arel, D.C Rose, T.P Karnowski, "Deep Machine Learning - A new frontier in artificial intelligence research [Research Frontier]," *IEEE Computational Intelligence Magazine*, Vol. 5, No. 4, pp. 13 – 18, 2010.

[2] Dong Yu, Li Deng, "Deep learning and its applications to signal and information processing" *IEEE Signal Processing Magazine*, Vol. 28, No. 1, pp.145-154, 2011.

[3] G.E. Hinton, and R.R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, Vol. 313. no. 5786, pp. 504 - 507, 28 July 2006.

[4] H. Larochelle, Y. Bengio, J. Louradour and P. Lamblin, "Exploring Strategies for Training Deep Neural Networks," *Journal of Machine Learning Research*, Vol. 1, pp. 1-40, 2009

[5] G. E. Hinton, S. Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, 18:1527–1554, 2006.

[6] M. Zeiler, D. Krishnan, G. Taylor, and R. Fergus, "Deconvolution networks," *Conf. Computer Vision Pattern Recognition* (CVPR), 2010.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE,* Vol. 86, pp. 2278–2324, 1998.

[8] G.W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional Learning of Spatio-Temporal Features," Proc. 11th European Conf. Computer Vision, pp. 140-153, 2010.

[9] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," *in Proc. Int. Conf. Mach. Learning (ICML)*, 2008.

[10] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle "Greedy Layer-Wise Training of Deep Networks," *Advances in Neural Information Processing Systems* 19 (NIPS'06), pages 153–160. MIT Press, 2007.

[11]M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," *in Neural Inf. Proc. Systems (NIPS)*, 2006.

[12]G. E. Hinton, A. Krizhevsky and S. D. Wang, "Transforming Auto-encoders," Artificial Neural Networks and Machine Learning – *ICANN 2011, Lecture Notes in Computer Science*, Volume 6791, 2011, pp 44-51.

[13]H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," In Zoubin Ghahramani, editor, *Twenty-fourth International Conference on Machine Learning (ICML 2007)*, pages 473–480. Omnipress, 2007.

[14] J. Tyler Rolfe and Y. LeCun, "Discriminative Recurrent Sparse Auto-Encoders," *International Conference on Learning Representations*, (ICLR2013), April 2013.

[15] R. Goroshin and Y. LeCun, "Saturating Auto-Encoders," *International Conference on Learning Representations* (ICLR2013), April 2013,

[16] R. Salakhutdinov, A. Mnih, G. Hinton "Restricted Boltzmann Machines for Collaborative Filtering," *Proc. of the 24th International Conference on Machine Learning, Corvalis*, Oregon, USA, 2007.

[17] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," *Technical Report, UTML TR 2010-003*, Department of Computer Science, 6 King's College Rd, Toronto, University of Toronto August 2010.

[18] M. Norouzi, M. Ranjbar, and G. Mori, "Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[19] I. J. Goodfellow, A. Courville, Y. Bengio, "Scaling Up Spike-and-Slab Models for Unsupervised Feature Learning," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 35, No. 8, pp. 1902- 1914, 2013.

[20] M. Norouzi, M. Ranjbar, and G. Mori, "Stacks of Convolutional Restricted Boltzmann Machines for Shift-Invariant Feature Learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[21] H. Larochelle, Y. Bengio, J. Louradour, P. Lamblin "Exploring Strategies for Training Deep Neural Networks," *Journal of Machine Learning Research*, Vol. 1 (2009) 1-40.

[22] A. Voulodimos, D. Kosmopoulos, G. Vasileiou, E. Sardis, V. Anagnostopoulos, C. Lalos, A. Doulamis, and T. Varvarigou, "A threefold dataset for activity and workflow recognition in complex industrial environments," *IEEE Multimedia Magazine,* Vol. 19, No. 3, pp42-52, 2012.

[23] A. Doulamis, D. Kosmopoulos, E. Sardis, T. Varvarigou, "An Architecture for Self Configurable Video Supervision," *ACM Workshop on Analysis and Retrieval of Events, Actions, Workflows in Video Streams in Conjunction with ACM Multimedia*, pp. 97-104, Vancouver, Canada, October 2008.

[24] C. Lalos, A. Voulodimos, A. Doulamis, and T. Varvarigou, "Efficient tracking using a robust motion estimation technique," *Multimedia Tools and Applications*, Springer, pp. 1-16, February 2012.

[25]A. S. Voulodimos, N. D. Doulamis, D. I. Kosmopoulos, and T. A. Varvarigou, "Improving multi-camera activity recognition by employing neural network based readjustment," *Applied Artificial Intelligence*, vol. 26, no. 1-2, pp. 97-118, 2012.

[26]D. I. Kosmopoulos, N. D. Doulamis, A. S. Voulodimos, "Bayesian filter based behavior recognition in workflows allowing for user feedback," *Computer Vision and Image Understanding*, Elsevier, vol. 116, no. 3, pp. 422–434, March 2012.