Spatio-Temporal Convolutional Sparse Auto-Encoder for Sequence Classification

Moez Baccouche[1]
moez.baccouche@orange.com

Franck Mamalet[1]
franck.mamalet@orange.com

Christian Wolf[2]
christian.wolf@liris.cnrs.fr

Christophe Garcia[2]
christophe.garcia@liris.cnrs.fr

Atilla Baskurt[2]
atilla.baskurt@liris.cnrs.fr

[1] Orange Labs R&D
4 rue du Clos Courtel
F-35510, France

[2] Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR 5205
F-69621, France

Abstract

We present in this paper a novel learning-based approach for video sequence classification. Contrary to the dominant methodology, which relies on hand-crafted features that are manually engineered to be optimal for a specific task, our neural model automatically learns a sparse shift-invariant representation of the local $2D+t$ salient information, without any use of prior knowledge. To that aim, a spatio-temporal convolutional sparse auto-encoder is trained to project a given input in a feature space, and to reconstruct it from its projection coordinates. Learning is performed in an unsupervised manner by minimizing a global parametrized objective function. The sparsity is ensured by adding a sparsifying logistic between the encoder and the decoder, while the shift-invariance is handled by including an additional hidden variable to the objective function. The temporal evolution of the obtained sparse features is learned by a long short-term memory recurrent neural network trained to classify each sequence. We show that, since the feature learning process is problem-independent, the model achieves outstanding performances when applied to two different problems, namely human action and facial expression recognition. Obtained results are superior to the state of the art on the GEMEP-FERA dataset and among the very best on the KTH dataset.

1 Introduction and related work

Learning machines that are able to automatically build feature extractors instead of handcrafting them is a wide research area in pattern recognition. The main benefit of these models is their high genericity since they can automatically learn to extract salient patterns directly from the raw input, without any use of prior knowledge. Yet they have been shown to yield excellent results in several tasks, e.g. object recognition [8, 10, 17, 19, 25], natural language processing [3], and audio classification [18], their extension to the video case is still an open issue. In this context, the dominant methodology in video sequence classification still relies

http://dx.doi.org/10.5244/C.26.124

on so-called *engineered* features, which are manually designed to be optimal for a specific task. Indeed, the most popular state-of-the-art features depend on the targeted application. For instance, in action recognition, we can mention *Harris-3D* [16], *Cuboid* [5], and *Hessian* features [31], while in facial expression recognition, the most popular features are local binary patterns (*LBP*) [24], *Gabor* [21] and *Haar-like* [30] features. The main purpose of this paper is to introduce a task-independent model, which can be applied to different problems.

However, direct and unconstrained learning of complex problems is difficult, since (i) the amount of required training data increases steeply with the complexity of the prediction model and (ii) training highly complex models with very general learning algorithms is extremely difficult. It is therefore common practice to restrain the complexity of the model. This is generally done either by forcing the model parameters to be identical for different input locations (as in *ConvNets* [17]), or by operating on small patches to reduce the input dimension and diversity, and to train the model in an unsupervised manner [25]. In this paper, we propose a novel model of the latter category, which is adapted to the video case.

Most unsupervised learning techniques for feature extraction rely on auto-encoders, i.e. an encoder trained to project a given input in a feature space, and a decoder which reconstruct it from its projection coordinates (a.k.a *the code*). This procedure generally produces a compact representation of the input content. However, several recent works advocate the use of *sparse-overcomplete* representations, i.e. whose dimension is larger than the input one, but where only a small number of components are non-zero. Several sparsifying procedures have been presented in the literature, including the one proposed by Ranzato *et al.* [25] for object recognition, which relies on a non-linear *sparsifying logistic*. Ranzato *et al.* have presented a learning algorithm to train the sparse model, and a specific procedure to handle shift-invariance. In this paper, we propose a solution based on the extension of this principle to the video case, with a novel approach for handling shift-invariance. A spatio-temporal convolutional sparse auto-encoder is trained to automatically build a sparse representation of local spatio-temporal patterns of sub $2D+t$ blocks in the video. The entire video sequence is then labelized considering the temporal evolution of these learned features, using a recurrent neural network.

The rest of the paper is organized as follows. Section 2 presents the proposed model for feature learning, and the corresponding training algorithm. We present in section 3 the architectures of the encoder and the decoder used in our experiments. The recurrent neural scheme for the entire sequence labelling is then described in section 4. Experimental results, on the KTH human actions [26] and the GEMEP-FERA facial expressions [29] datasets, are given in section 5. Finally, we conclude and give some perspectives of this work.

2 Unsupervised learning of sparse shift-invariant spatio-temporal features

In this section, we describe the proposed approach for unsupervised learning of sparse spatio-temporal features. We first introduce the proposed model, and then present the corresponding training algorithm.

2.1 The model

Our sequence classification model is hierarchical: each element of the sequence corresponds to a space-time block (a set of consecutive frames or of parts of frames) where each block

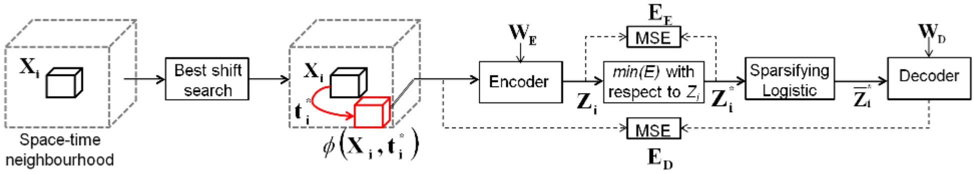

Figure 1: Overview of the proposed model for unsupervised learning of sparse shift-invariant spatio-temporal features.

itself is decomposed into a set of small space-time patches. The space-time patches are the representation level on which learning is performed. The rationale of using small patches is to reduce the diversity of the content to be encoded, since the patterns are locally less variable than if the full frame was considered. The set of training patches will be denoted by $\{X_i\}_{i\in[1..P]}$, they are of size $M\times M\times T$ pixels each (where P is the number of training sample patches, M and T are respectively the spatial and temporal sizes of the patch). The two main parts of the proposed model are: an encoder (with trainable parameters W_E), which builds a non-sparse code vector Z_i representing the spatio-temporal salient information contained in the input, and a decoder (with trainable parameters W_D), which learns to reconstruct the input volume from a sparse version $\overline{Z}_i$ of the obtained code (see Figure 1).

As in [25], the system learns a compact code which can reconstruct the input. A *sparsifying logistic* between the encoder and the decoder restrains the size of the code. It is a non linear function which can be seen as a SoftMax applied on consecutive samples of the same code unit. Given the i^{th} training sample, and its corresponding non-sparse code $Z_i = \left\{z_i^{(k)}\right\}_{k\in[1..N]}$, where N is the code size, the sparse code $\overline{Z}_i = \left\{\overline{z}_i^{(k)}\right\}_{k\in[1..N]}$ will be expressed by:

$$\overline{z}_i^{(k)} = \frac{\eta e^{\beta z_i^{(k)}}}{\xi_i^{(k)}} \quad with \quad \xi_i^{(k)} = \eta e^{\beta z_i^{(k)}} + (1-\eta)\,\xi_{i-1}^{(k)} \tag{1}$$

where η and β are positive parameters controling the code sparsity and softness (large η values increase the number of samples used to compute ξ, and large β values yield quasi-binary outputs). Due to the resulting strong non-linearities, the encoder and the decoder are learned separately: for each training sample, when updating the parameters of one of the two parts, those of the other part are kept constant. A third (and also separate) step produces optimal code, in that its sparse version is the best decoded, while being close to the original code. i.e. Z_i is considered as an additional optimizable parameter. All three steps minimize a single global objective function (see equation (2)) with respect to different parameters at each step. Details are given in 2.2.

In order to handle the spatial and temporal shift-invariance of the learned representations (i.e. the model assigns the same code to the spatially and temporally shifted versions of a given input), we introduce an additional hidden variable t_i (a three-dimensional translation vector), on which the optimization is done. The idea is to represent the spatio-temporal neighbourhood of a given input patch X_i by a single patch $\phi(X_i,t_i)$, which is the one minimizing the objective function, given the current set of parameters (W_E,W_D) (see Figure 1). Note that in practice, this procedure is started only after the first epoch (a full pass over all the training samples) to ensure the relevance of the encoder/decoder parameters.

Finally, the global objective function E is the one given by equation (2). It is a sum of two terms, representing respectively the encoder prediction and the decoder reconstruction mean square errors (MSE) on the translated patches:

$$\begin{aligned} E\left(X_i, t_i, Z_i, W_E, W_D\right) &= E_E\left(X_i, t_i, Z_i, W_E\right) + E_D\left(X_i, t_i, Z_i, W_D\right) \\ &= \left\|Z_i - Enc\left(W_E, \phi\left(X_i, t_i\right)\right)\right\|^2 + \left\|Dec\left(W_D, \overline{Z}_i\right) - \phi\left(X_i, t_i\right)\right\|^2 \end{aligned} \tag{2}$$

We describe in 2.2 how this objective function is minimized with respect to the parameters $\left(W_E, W_D, Z_i, t_i\right)$.

2.2 Training procedure

The model is trained *on-line*, i.e. the parameters are updated after considering each training sample. We aim to find, for a given input X_i, the optimal set of parameters $\left(W_E^*, W_D^*, Z_i^*, t_i^*\right)$ which minimize E. The training procedure is performed in three steps, each step aims to minimize E with regard to one of the parameters keeping the others constant, as expressed below. Here the notation $E(a|b)$ emphasizes the fact that we optimize over parameters a while keeping parameters b constant.

$$t_i^* = \arg\min_{t_i} E\left(t_i | X_i, Z_i, W_E, W_D\right) \tag{3}$$

$$Z_i^* = \arg\min_{Z_i} E\left(Z_i | X_i, t_i^*, W_E, W_D\right) \tag{4}$$

$$\left(W_E^*, W_D^*\right) = \arg\min_{W_E, W_D} E\left(W_E, W_D | X_i, t_i^*, Z_i^*\right) \tag{5}$$

Thus, learning is performed with the following algorithm, applying steps 2-3-4 for each input patch X_i:

1. $\left(W_E, W_D\right)$ are randomly initialized.
2. For a given X_i, an exhaustive search is performed in a spatio-temporal neighbourhood to find t_i^* as expressed in equation (3).
3. Given $\phi\left(X_i, t_i^*\right)$, equation (4) is minimized using steepest descent on the Z_i parameter to find the optimal code Z_i^*.
4. W_E and W_D are updated using standard *on-line* backpropagation, targeting Z_i^* and $\phi\left(X_i, t_i^*\right)$ respectively for the encoder and the decoder, as expressed in equation (5).

The third step of this training algorithm requires the calculation of the partial derivative of E with respect to Z_i, for gradient computation. This partial derivative is approximated by a finite difference, as expressed by equation (6), considering a small non-zero fixed value dZ.

$$\frac{\partial E\left(Z_i | X_i, t_i^*, W_E, W_D\right)}{\partial Z_i} = \frac{E\left(Z_i + dZ | X_i, t_i^*, W_E, W_D\right) - E\left(Z_i - dZ | X_i, t_i^*, W_E, W_D\right)}{dZ} \tag{6}$$

In order to avoid encoding non-relevant patterns (e.g. colour and texture), and to reduce the amount of data to be encoded, the model is trained only with the patches containing significant spatio-temporal information. Thus, a motion-based patch selection module is

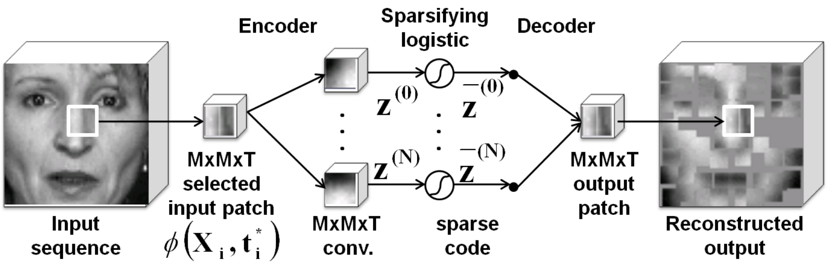

Figure 2: Architecture of the spatio-temporal convolutional sparse auto-encoder: illustration on a sample from the GEMEP-FERA facial expressions dataset.

placed before the encoder. It calculates an absolute difference between the frames $\frac{T}{2}$ and $-\frac{T}{2}$ of the input (i.e. the first and the last ones), and retains those with a percentage of moving pixels exceeding a certain threshold. This module plays the same role as the saliency detectors in the case of the hand-crafted features, but does not use complex processing since the outliers will be filtered during the training, and will not be encoded.

The next section details the architectures of the different modules.

3 The auto-encoder scheme

The functional forms of the encoder and decoder are specified with the architecture illustrated in Figure 2, which can be described as follows:

The encoder contains N trainable 3D convolution kernels of size $M \times M \times T$ each. It takes as input a small $M \times M \times T$ spatio-temporal patch and learns to compute a non-sparse code of size N corresponding to the response of each convolution, and encoding the spatio-temporal salient information of the patch. Each $M \times M \times T$ convolution kernel contains an additional trainable bias and a linear activation function, thus, the total number of trainable parameters of the encoder is $(M \times M \times T + 1) \times N$. To ensure overcompleteness, N should be greater than or equal to the input dimension, but in practice, since there is a high correlation between the T consecutive input frames, even if N is smaller than $M \times M \times T$, the representation is still overcomplete. The optimization of t_i is performed for each component among values in $[-M/4, M/4]$ for the spatial shift and $[-T/2, T/2]$ for the temporal one, i.e. the selected shifted-patch is located in a $3M/2 \times 3M/2 \times 2T$ neighbourhood around the initial position.

The sparsifying logistic described in equation (1), is replaced, as in [25], by a classical logistic function with a constant gain after training, since ξ turns to a fixed value which is its mean over all training samples.

The decoder consists of a set of $M \times M \times T$ output neurons fully connected to the sparse code layer, and to an additional trainable bias. The total number of parameters of the decoder is thus $(N \times M \times M \times T) + 1$. The output is a weighted sum of elementary spatio-temporal patches (which will be called "basis" in the following), which corresponds to the set of decoder responses, when stimulated with sparse codes that have

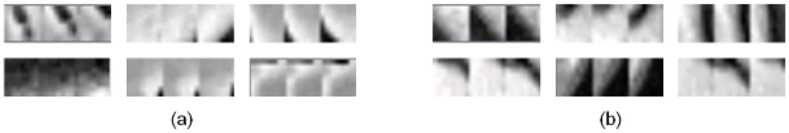

Figure 3: A subset of learned basis elements: (a)- KTH human actions dataset. (b)- GEMEP-FERA facial expressions dataset. On both illustrations, each basis element is composed of three ($T = 3$) $M \times M$ images.

only one non-zero component equal to 1. Since the code is sparse, only few elements of the basis are used to reconstruct each output (typically, in all our experiments, the number of activated units has never exceeded $N/8$).

We depict in Figures 3-(a) and 3-(b) a subset of the learned basis respectively on the KTH human actions [26] and GEMEP-FERA facial expressions [29] datasets. We can note that no element of the basis is a shifted version of another one. The auto-encoder can reconstruct each input patch by combining a few elements of this basis. For each reconstructed patch, the coefficients of the basis elements correspond to the coordinates in the projection space of the input patch. These coordinates are used hereafter as features to represent the patch neighbourhood content. In the next section, we will present our proposed method to learn the temporal evolution of these features, using a recurrent neural network trained for classifying the entire sequences.

4 Sequence classification considering the temporal evolution of learned features

Entire sequences are labeled with a particular recurrent neural network classifier, namely *Long Short-Term Memory Recurrent Neural Network* (hereafter LSTM-RNN), in order to take benefits of its ability to use the temporal evolution of features for classification. We first present some LSTM-RNN fundamentals, and then describe the LSTM-RNN architecture for sequence classification.

4.1 Long short-term memory recurrent neural networks

Intuitively, a Recurrent Neural Network (RNN) can be seen as a particular neural network which can remember previous inputs and use them to influence the network output. This can be done by using recurrent connections in the hidden layers, and makes them particularly suited for temporal analysis of data, because of their ability to take into account the context. Nevertheless, even if they are able to learn tasks which involve short time lags between inputs and corresponding teacher signals, this short-term memory becomes insufficient when dealing with "real world" sequence processing, e.g. video sequences.

The LSTM-RNN architecture was introduced by Hochreiter and Schmidhuber [11] in order to alleviate this problem by adding a special node, called *constant error carousel* (CEC). This allows for constant error signal propagation through time, and thus, provides remedies for the RNN's problem of *exponential error decay* [11]. Recently, improved versions of the

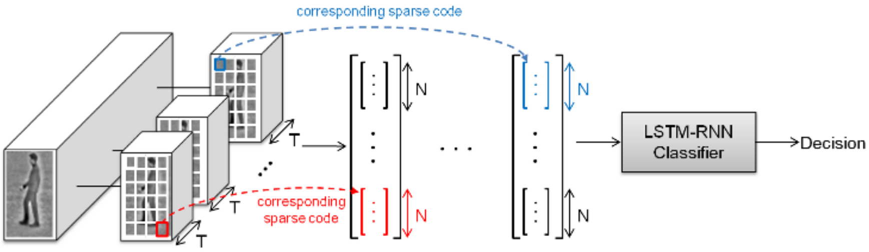

Figure 4: Illustration of the feature sequence generation and the LSTM-RNN classification.

LSTM-RNN were proposed [9], in which multiplicative gates are added to control the access to the internal state of the CEC. Graves and Schmidhuber [9] also introduced a so-called bidirectional LSTM-RNN model, i.e. two separate LSTM-RNNs, both connected to an output layer, and to which each training sequence is presented in forward and backward directions. This permits to introduce more context, and thus to improve the classification performances [9]. LSTM-RNN have been tested in many applications, like phoneme classification [9] or action recognition [1], and generally outperformed existant methods. In this paper, the classification part of all experiments was carried out using the model proposed by Graves and Schmidhuber in [9], fed with the sparse spatio-temporal features.

4.2 The proposed architecture

If we denote by H and W respectively the number of rows and columns of the original video frame, by L the sequence length, and by N the sparse code size, the LSTM-RNN takes as input a sequence of L feature vectors of size $W/M \times H/M \times N$, each one corresponding to the concatenated responses of the $M \times M \times T$ patches placed at the $W/M \times H/M$ grid of possible locations in each space-time block. Figure 4 illustrates this principle on a sample taken from the KTH human actions dataset. The LSTM cells are fully connected to these inputs and have also auto-recurrent connections. For the output layer, we used the SoftMax activation function, which is standard for 1 out of K classification tasks. The SoftMax function ensures that the network outputs are all between 0 and 1, and that their sum is equal to 1 at every timestep. These outputs can then be interpreted as the posterior probabilities of the actions at a given timestep. In all our experiments, the hidden layer contains 5 LSTM neurons for each direction (forward and backward). The network was trained with standard *online backpropagation through time with momentum* [9, 11].

5 Experimental results

5.1 Human action recognition

The KTH dataset [26] is the most commonly used dataset for human action recognition. It contains 6 types of actions (boxing, clapping, waving, walking, jogging and running) performed by 25 subjects in 4 different scenarios. As in [7], we rename the KTH dataset in two ways: the first one (the original one) where each person performs the same action 3 or 4 times in the same video, is named KTH1 and contains 599 long sequences with

	Boxing	Clapping	Waving	Walking	Jogging	Running	Avg.
KTH1	98.00	98.00	98.00	99.00	93.00	89.00	**95.83**
KTH2	97.67	94.00	96.98	99.23	89.22	85.36	**93.74**

Table 1: Obtained results on the KTH human actions dataset.

Dataset	Features	Method	Accuracy
KTH1	Learned	**Ours**	**95.83**
		Jhuang *et al.* [13]	91.70
	Hand-crafted	Gao *et al.* [7]	96.33
		Chen and Hauptmann [2]	95.83
		Liu and Shah [22]	94.20
KTH2	Learned	**Ours**	**93.74**
		Ji *et al.* [14]	90.20
		Taylor *et al.* [28]	90.00
	Hand-crafted	Kim *et al.* [15]	95.33
		Ikizler *et al.* [12]	94.00
		Gao *et al.* [7]	92.45

Table 2: Comparison of obtained results on KTH dataset with the state-of-the-art.

several “empty” frames between action iterations. The second, named KTH2, is obtained by splitting videos into smaller ones with no empty frame, and contains 2391 sequences. For each original video, we extracted the person-centered bounding box as in [13], and performed spatial down-sampling by a factor of 2 horizontally and vertically. The resulting frame dimensions are $H = 54$ and $W = 34$. The model was trained as described above, with input patches of size $8 \times 8 \times 3$, encoded to a sparse code of size 192. For the *sparsifying logistic*, η and β were fixed respectively to 0.02 and 1.5 for all our experiments. The LSTM-RNN input size is 4608 per time step, which corresponds to the concatenated responses of the 4×6 possible locations of the input patches in the original space-time block. We performed leave-one-out cross validation as recommended by Gao *et al.* in [7] and reported the average accuracies per class in Table 1. We can notice that the best performances are obtained for the class walking, because of the characteristic low-speed motion existing in this action, which is a highly discriminative information for the LSTM-RNN model. We also observe that the long sequences (KTH1) achieve better performances than the short ones (KTH2), confirming the fact that LSTM-RNNs are more suited for long sequences. We also compare in Table 2 obtained results to related works on the KTH dataset. Among the methods using automatically learned features, to our knowledge, our method obtained the best results, both on KTH1 (95.83%) and KTH2 (93.74%). More generally, according to the survey made by Gao *et al.* in [7], we obtain the second best results for KTH1, and the third for KTH2, even when compared with approaches relying on hand-crafted features designed for the KTH dataset.

5.2 Facial expression recognition

The GEMEP-FERA dataset [29] is a recent facial expressions dataset that was initially presented for the *facial expression recognition and analysis challenge* (FERA 2011). The *emotion sub-challenge* dataset contains 5 types of facial expressions (anger, fear, joy, relief and sadness) displayed by 10 actors, while uttering a meaningless phrase or a sustained vowel

	Anger	Fear	Joy	Relief	Sadness	Avg.
PI	100.0	93.33	95.00	68.75	46.67	**80.75**
PS	92.31	100.0	100.0	100.0	100.0	**98.46**
Overall	96.30	96.00	96.77	80.77	68.00	**87.57**

Table 3: Obtained results on the GEMEP-FERA facial expressions dataset.

Method	PI	PS	Overall
Ours	**80.75**	**98.46**	**87.57**
Yang and Bhanu [32]	75.23	96.18	83.78
Tariq *et al.* [27]	65.50	100.0	79.80
Littlewort *et al.* [20]	71.40	83.70	76.10
Dhall *et al.* [4]	64.80	88.70	73.40
Meng *et al.* [23]	60.90	83.70	70.30
Valstar *et al.* [29]	44.00	73.00	56.00

Table 4: Comparison of obtained results on GEMEP-FERA dataset with the state-of-the-art.

"aaa". The training set contains 155 video sequences and includes 7 subjects, while the test set contains 134 sequences and 6 subjects, 3 of which are not present in the training set. This enables to test the method on both person independent (PI) and person specific (PS) settings, to evaluate the generalization power. The dataset is particularly challenging since a high intra-class confusion exists even for a human (between joy/relief and anger/fear/sadness). In addition, the original videos are not face-centered and include several non-facial movements (hands, body...), which implies that a face detection process should be applied (which is not straightforward since high blurs are present). The face-centered bounding box was extracted from the original 3-channel frames of size 720×567 using the the *convolutional face finder* [8] (which achieved near-perfect results), and consecutive face-centered images were aligned using the approach proposed in [6]. Some alignement imprecision still exists, but will be handled by the shift-invariance property of the proposed system. The extracted images finally underwent grayscale transform and size rescaling to 64×64. The model was trained with input patches of size $8 \times 8 \times 3$, encoded into a sparse code of size 128, with the same parameter values used for human actions recognition. The LSTM-RNN input size is 8192 per timestep (8×8 codes of size 128 each). We followed the same benchmark procedure as for the original challenge (which is described in the baseline paper [29]). Average accuracies per class, both for person-independant and person-specific configurations, are reported in Table 3. We also present in Table 4 a comparison with the best results obtained in the FERA 2011 Challenge. Up to our knowledge, no other approach gives a better performance on this dataset (87.57% for the overall classification rate), with +3.79 percentage points improvement on the result presented by Yang and Bhanu [32], which are the winners of the challenge. Moreover, the obtained high score on the person-independent configuration is a positive evidence of the high generalization of our approach, and that it eliminates the person-specific effect and captures the facial expression salient information.

6 Conclusion and future work

In this paper, we have presented a neural model for sequence classification, which differs from the dominant methodology in that the feature construction process is learning-based and fully automated. We have introduced the spatio-temporal convolutional sparse auto-encoder architecture, which builds a sparse representation of $2D+t$ patterns present in the video. We have described how the model is trained only with the patches containing relevant space-time information, to avoid encoding uninteresting patterns. We have also presented a novel approach for handling both spatial and temporal shift-invariance of the representation, by including an additional hidden variable to the objective function. Finally, we have shown how the temporal evolution of these features is used to classify the sequences, using a recurrent neural network model. Experimental results on two different problems, namely human actions and facial expressions recognition, confirms the high genericity of the model since it achieves the best results among related works, even when compared to methods using hand-crafted features. Future work will address scale invariance and applications to different problems.

Acknowledgements The authors would like to thank Michel Valstar (University of Nottingham) for his help in the evaluation of our model on the GEMEP-FERA dataset.

References

[1] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential Deep Learning for Human Action Recognition. In *International Workshop on Human Behavior Understanding*, pages 29–39, 2011.

[2] M.Y. Chen and A. Hauptmann. Mosift: recognizing human actions in surveillance videos date of original version. Technical Report CMU-CS-09-161, Carnegie Mellon University, 2009.

[3] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning*, pages 160–167, 2008.

[4] A. Dhall, A. Asthana, R. Goecke, and T. Gedeon. Emotion recognition using phog and lpq features. In *International Conference on Automatic Face & Gesture Recognition*, pages 878–883, 2011.

[5] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

[6] S. Duffner and C. Garcia. Robust face alignment using convolutional neural networks. In *International Conference on Computer Vision Theory and Applications*, 2008.

[7] Z. Gao, M.Y. Chen, A. Hauptmann, and A. Cai. Comparing evaluation protocols on the kth dataset. In *International Workshop on Human Behavior Understanding*, pages 88–100. 2010.

[8] C. Garcia and M. Delakis. Convolutional face finder: A neural architecture for fast and robust face detection. *Transactions on Pattern Analysis and Machine Intelligence*, 26 (11):1408–1423, 2004.

[9] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

[10] G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

[12] N. Ikizler, R.G. Cinbis, and P. Duygulu. Human action recognition with line and flow histograms. In *International Conference on Pattern Recognition*, pages 1–4, 2008.

[13] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *International Conference on Computer Vision*, pages 1–8, 2007.

[14] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. In *International Conference on Machine Learning*, pages 495–502, 2010.

[15] T.K. Kim, S.F. Wong, and R. Cipolla. Tensor canonical correlation analysis for action classification. In *Computer Vision and Pattern Recognition*, pages 1 –8, 2007.

[16] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition*, pages 1 –8, 2008.

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278 –2324, 1998.

[18] H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *International Conference on Machine Learning*, pages 609–616, 2009.

[19] H. Lee, P. Pham, Y. Largman, and A. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Neural Information Processing Systems*, pages 1096–1104, 2009.

[20] G. Littlewort, J. Whitehill, T.F. Wu, N. Butko, P. Ruvolo, J. Movellan, and M. Bartlett. The motion in emotion - a cert based approach to the fera emotion challenge. In *International Conference on Automatic Face & Gesture Recognition*, pages 897–902, 2011.

[21] C. Liu and H. Wechsler. Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *Transactions on Image Processing*, 11 (4):467–476, 2002.

[22] J. Liu and S. Mubarak. Learning human actions via information maximization. In *Computer Vision and Pattern Recognition*, pages 1 –8, 2008.

[23] H. Meng, B. Romera-Paredes, and N. Bianchi-Berthouze. Emotion recognition by two view svm 2k classifier on dynamic facial expression features. In *International Conference on Automatic Face & Gesture Recognition*, pages 854–859, 2011.

[24] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[25] M.A. Ranzato, F.J. Huang, Y.L. Boureau, and Y. Lecun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[26] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *International Conference on Pattern Recognition*, volume 3, pages 32 – 36, 2004.

[27] U. Tariq, K.H. Lin, Z. Li, X. Zhou, Z. Wang, V. Le, T.S. Huang, X. Lv, and T.X. Han. Emotion recognition from an ensemble of features. In *International Conference on Automatic Face & Gesture Recognition*, pages 872–877, 2011.

[28] G.W. Taylor, R. Fergus, Y. LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European Conference on Computer Vision*, pages 140–153, 2010. ISBN 3-642-15566-9, 978-3-642-15566-6.

[29] M.F. Valstar, B. Jiang, M. Mehu, M. Pantic, and K. Scherer. The first facial expression recognition and analysis challenge. In *International Conference on Automatic Face & Gesture Recognition*, pages 921–926, 2011.

[30] P. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[31] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *European Conference on Computer Vision*, pages 650–663, 2008.

[32] S. Yang and B. Bhanu. Facial expression recognition using emotion avatar image. In *International Conference on Automatic Face & Gesture Recognition*, pages 866 –871, 2011.