Residual Stacked RNNs for Action Recognition

Mohamed Ilyes Lakhal[1(✉)], Albert Clapés[2], Sergio Escalera[2], Oswald Lanz[3], and Andrea Cavallaro[1]

[1] CIS, Queen Mary University of London, London, UK
{m.i.lakhal,a.cavallaro}@qmul.ac.uk
[2] Computer Vision Centre, University of Barcelona, Barcelona, Spain
aclapes@gmail.com, sergio.escalera.guerrero@gmail.com
[3] TeV, Fondazione Bruno Kessler, Trento, Italy
lanz@fbk.eu

Abstract. Action recognition pipelines that use Recurrent Neural Networks (RNN) are currently 5–10% less accurate than Convolutional Neural Networks (CNN). While most works that use RNNs employ a 2D CNN on each frame to extract descriptors for action recognition, we extract spatiotemporal features from a 3D CNN and then learn the temporal relationship of these descriptors through a stacked residual recurrent neural network (Res-RNN). We introduce for the first time residual learning to counter the degradation problem in multi-layer RNNs, which have been successful for temporal aggregation in two-stream action recognition pipelines. Finally, we use a late fusion strategy to combine RGB and optical flow data of the two-stream Res-RNN. Experimental results show that the proposed pipeline achieves competitive results on UCF-101 and state of-the-art results for RNN-like architectures on the challenging HMDB-51 dataset.

Keywords: Action recognition · Deep residual learning · Two-stream RNN

1 Introduction

An important challenge in action recognition is to effectively model temporal dynamics and long-term dependencies. If the temporal evolution of actions is unaccounted for, similar actions (e.g. *answering phone* and *hanging-up phone*) may be confused. Also to encode discriminative spatiotemporal descriptors from the high-dimensional input video stream is key to address the recognition task.

Action recognition methods based on hand-crafted features [1,2] have recently been surpassed, in terms of recognition accuracy, by end-to-end trainable deep architectures [3]. Earlier deep learning approaches use convolutional neural networks to classify individual frames based on appearance information only, and then average their class scores to expose the video prediction [4]. To

L. Leal-Taixé and S. Roth (Eds.): ECCV 2018 Workshops, LNCS 11130, pp. 534–548, 2019.
https://doi.org/10.1007/978-3-030-11012-3_40

complement appearance information, low-level motion descriptors can be produced by feeding short clips as input to the network [5] instead of single frames, or by exploiting optical flow [6]. Integrating flow and appearance at the appropriate stage of the network can boost performance significantly [7]. Moreover, 3D ConvNets extend the convolutions in time to extract features from video segments [3,5,8]. To deal with the increased complexity of 3D ConvNets, pre-trained 2D image recognition-based CNNs can be inflated to 3D (Two-Stream Inflated 3D ConvNets (I3D) [3]). Residual networks [9] use skip connections to address the *degradation problem* (increasing the number of layers in a feed-forward network leads to a decrease in performance on both test and training data), allowing to effectively increase the depth of the network without augmenting the number of parameters.

The temporal evolution of the output of deep convolutional networks that use frames can be modeled by sequence models, e.g. Long-Short Term Memory (LSTM) [10]. To increase the depth of sequential models, LSTM layers can be stacked [11], but might suffer from the *degradation problem*. Networks with a large number of parameters or layers starve with action datasets, because of the limited training data (e.g. 10K videos in the most popular benchmarking datasets, UCF-101 and HMDB-51). It is therefore important not to increase the number of optimizable parameters.

In this paper, we extend multi-layer LSTMs with residual connections and explicitly learn *appearance* and *motion* in two separate streams. To the best of our knowledge, this is the first time Residual LSTMs are applied to video action recognition, as they were only applied to speech processing [12]. We use as input features extracted from a 3D CNN, which provides a much richer representation than a 2D CNN and reduces redundancy observed by the LSTMs on top. The predictions of the two-stream residual LSTMs are late fused using element-wise dot aggregation, as opposed to adding the final score predictions. The proposed architecture achieves competitive results on the UCF-101 dataset and outperforms state-of-the-art methods for RNN-like architecture on the HMDB-51 dataset.

The paper is organized as follows: Sect. 2 reviews related work. Section 3 introduces the proposed two-stage pipeline and data generation. Section 4 covers the experimental protocol and discusses the results. Finally, Sect. 5 concludes the paper.

2 Related Work

Spatiotemporal features can be learned by extending over time the connectivity of a 2D CNN and fusing responses from subsequent frames in short fix-length clips. This fusion leads to marginal gains over averaging predictions from a set of uniformly sampled frames [4]. 3D CNN [5] extend convolutional kernels and pooling operations over time, e.g. on 16-frame clips [13]. Increasing the temporal interval can improve recognition, except for actions with periodic patterns, by encoding longer temporal dynamics [8]. Improved results are obtained by combining long-term convolution networks with different temporal intervals, e.g. 60

and 100 frames [8]. Averaging predictions from a few random frames (or clips) may select samples that are uninformative for the action class. Max-pooling can be applied to the last convolutional layer map responses of a 2D CNN across all the frames of the video (*Conv Pooling*) [11]. Subaction patterns shared by actions classes may uniquely be assigned to a particular class, instead of to multiple classes, using frames or clips classification. To address this problem, local spatiotemporal descriptors can be aggregated over the video by softly assigning them to subaction anchors, as in ActionVLAD [14].

Temporal dynamics can be handled by sequence modeling via recurrent neural networks. Each frame can be processed in a shallow 3D CNN and the output responses modeled with an LSTM trained separately [15]. End-to-end trained architectures exists that combine a 2D CNN and a 5-layer LSTM [11]. Gated-recurrent Units (GRU) [16], a simpler variant of LSTMs with similar performance [17], consist of two gates only (*update* and *reset* gates) and the internal state (*output state*) is fully exposed. We refer the reader to [18] for an in-depth survey on recent advances in the subject.

Convolutional LSTMs [19] replace the fully connected layers in input-to-state and state-to-state transitions with convolutional layers, thus maintaining the spatial structure of input frames or convolutional maps from input to output. Spatial data redundancy can be reduced by substituting the fully connected layer with convolutions. Each pair of frames can be passed through two 2D CNNs with shared weights, whose outputs are then connected in a 3D convolutional layer and passed to a ConvLSTM [20]. ConvALSTM [21] combines the benefits of both ConvLSTM and Attention-LSTM [22,23], but relies on a shallow convolutional network for the soft-attention mechanism, unlike the Attention-LSTM. L^2STM [24] extends the LSTM formulation to learn independent hidden state transitions of memory cells for individual spatial locations.

Motion information (e.g. optical flow) and appearance (e.g. RGB data) can be modeled in two separate classification models and then the two independent softmax predictions are (late) fused. Examples of architectures include a two-stream 2D CNN [6], a long-term convolutional network [8], and ActionVLAD [14]. Moreover, spatial and temporal streams can also be fused after the last convolutional layer [7]. Cross-modal training is also possible on sequential models: L^2STM [24] modifies the gradient update equations for the input and forget gates to jointly adapt their parameters accounting for the loss of both streams during back-propagation. Motion information is easier for training from scratch with limited training data [6,8], and is useful for attentional mechanisms and action localization [21].

3 Residual Learning with RNNs

In this section we present the proposed stacked residual network (depicted in Fig. 1) and the data generation process. We also present our approach for model selection and evaluation.

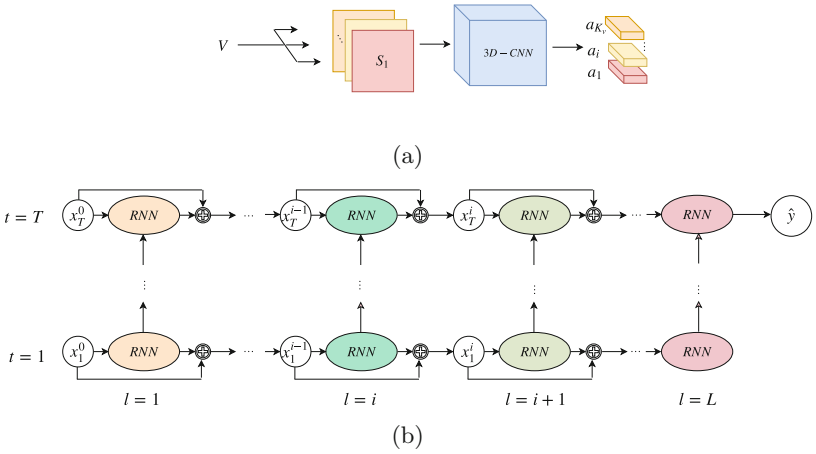


Fig. 1. The proposed residual stacked RNN architecture. The input video, V, is divided into clips from which spatiotemporal features are extracted with a 3D CNN. The residual stacked RNN learns temporal dependencies between an action class and elements of a sampled subset of features, x^0, of duration T.

3.1 The Two-Stage Pipeline

Let $V \in \mathbb{R}^{m_x \times m_y \times l_v}$ be a video of duration l_v that represents an action and whose frames have size $m_x \times m_y$ pixels. Let $K_v = \lfloor \frac{l_v}{r} \rfloor - 1$, be the number of clips in V, where r is the stride between each clip. Let each clip $S_i \in \mathbb{R}^{m_x \times m_y \times s}$ have a duration s empirically defined so that S_i captures a gesture in V. A gesture is a movement of body parts that can be used to control and to manipulate, or to communicate. A temporal concatenation of gestures composes an action [25]. Unlike frame-level predictions [11], we aim to encapsulate each S_i with a latent representation of much lower dimension, $a_i \in \mathbb{R}^{d_f}$, which is expected to encode a gesture.

Let the spatiotemporal features of S_i be extracted[1] with a 3D CNN [3,26], as $a_i = E(S_i) \in \mathbb{R}^{d_f}$, thus obtaining $a = \{a_i | i = 1 \dots K_v; a_i \in \mathbb{R}^{d_f}\}$, from which we extract a subset x^0 of size T: $x^0 = \{x_t^0 = a_{\sigma(t)} | t = 1, \dots, T; a_{\sigma(t)} \in a\}$, where $\sigma(t) = 1 + \lfloor (t-1)\frac{K_v - 1}{T-1} \rfloor$.

The final step learns the temporal dependencies between the input sequence x^0 using the residual recurrent neural network. Instead of fitting an underlying mapping $H(x)$, the same stack of layers in residual learning generates $H(x)$ to fit another mapping $F(x) = H(x) + x$. The network learns a residual mapping $F(x)$ to approximate $H(x)$ rather than $F(x)$. Given a stack of recurrent neural units of depth L, at layer l and timestep t the input is x_t^{l-1}. The previous memory and cell states for layer l from timestep $t-1$ are m_{t-1}^l and c_{t-1}^l, respectively.

[1] Note that our architecture is independent from the particular CNN structure and other models can be used as extractor, e.g. TSN [26] or I3D [3] (see Fig. 1(a)).

Then, m_t^l and c_t^l are calculated using the recurrent equations *e.g.* LSTM [27], and the input to layer $l+1$ at timestep t is $x_t^l = m_t^l + x_t^{l-1}$.

Each RNN layer in the residual RNN part has index $l \in \{1, \ldots, L\}$. The dimension of the input at time t in layer l must be the same as the memory m_t^l since the addition in the residual equation is performed element-wise. The overall structure is the same as in [28] (see Fig. 1(b)).

Let Θ^l be the parameter of the recurrent model at layer l, and L the total number of LSTM layers. If P is total number of action classes, $m_t^l, x_t^l \in \mathbb{R}^{d_f}$, $y \in \mathbb{R}^P$, and $W_y \in \mathbb{R}^{d_f \times P}$ is a fully connected layer, then the recurrent part of our hierarchical residual RNN model is updated using:

$$\begin{aligned} c_t^l, m_t^l &= LSTM_l(c_{t-1}^l, m_{t-1}^l, x_t^{l-1}; \Theta^l) \\ x_t^l &= m_t^l + x_t^{l-1} \end{aligned} \tag{1}$$

where T is the number of time steps. At the l-th layer we obtain the hidden state from $LSTM_l$ using the input x_t^{l-1}, the input at the $(l+1)$-th layer is the residual equation: $x_t^l = m_t^l + x_t^{l-1}$. We obtain the final score $\hat{y}$ through a softmax layer at the last time step T using

$$\hat{y} = \texttt{softmax}((m_T^L)^\top . W_y). \tag{2}$$

Under this formulation, our residual RNN model will learn the temporal dependencies of each clip S_i and perform a video level prediction by adding a softmax layer on top of the last LSTM unit at the last time step T (see Fig. 1(b)).

3.2 Fusion

We extend the multi-layer residual LSTM to handle two streams, namely the RGB video, V, and the optical flow, which has been shown to be useful for CNN based models [7]. Girdhar [14] explored three fusion methods to combine RGB and Flow CNN models.

Given two feature vectors $u, v \in \mathbb{R}^m$, we consider for fusion the element-wise sum and the element-wise product. For the element-wise sum, $\oplus$, the fusion is $u \oplus v = (u_1 + v_1, \ldots, u_m + v_m)$. This method is expected to help if the two responses have high scores, whereas small perturbations in either vector will be ignored. The element-wise product, $\odot$, is $u \odot v = (u_1.v_1, \ldots, u_m.v_m)$. This scheme encourages the highest scores and tends to diminish small responses.

We feed the input video, V, to a pre-trained 3D-CNN, $x^c = \text{3D-CNN}^c(V)$, and the optical flow, V^f, through a pre-trained flow model, $x^f = \text{3D-CNN}^f(V^f)$ (see Fig. 2(a)). With *mid fusion*, $x^0 = \{x_t^0 = x_t^c \circ x_t^f | t = 1, \ldots, T\}$, where $\circ \in \{\oplus, \odot\}$. We then train a Res-LSTM model using x^0 as input. With *late fusion*, we use the input x^c (x^f) to train a Res-LSTM for the appearance (optical flow) network. Once the models are trained we obtain the softmax predictions, $\hat{y}^c$ and $\hat{y}^f$, for each modality network. The final prediction is $\hat{y} = \hat{y}^c \circ \hat{y}^f$, where $\circ \in \{\oplus, \odot\}$ (see Fig. 2(b)). Results of these fusion methods are shown in Table 2.

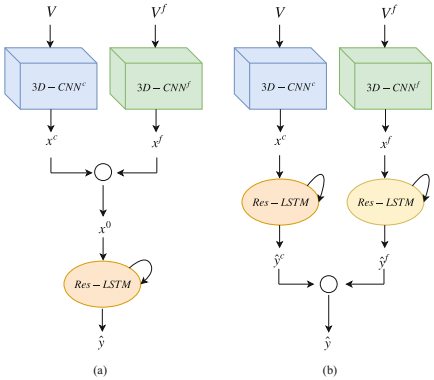


Fig. 2. Fusion schemes on two-stream Res-LSTM: (a) in *mid fusion* after obtaining the spatiotemporal features from the RGB(Flow) network we combine the features using either element-wise or dot product to produce a single vector x^0 that serves as the input to a Res-LSTM model, (b) *late fusion* on the other hand trains a separate Res-LSTM for each modality input, the predictions of each modality network are then combined for a final prediction.

3.3 Data Generation

We consider two strategies of data augmentation [24], depending on the CNN architecture used. The first strategy consists of fixing the spatial dimension and varying the temporal axis of the input stream,

$$V_S = \{V_{clip}^{(1)}, V_{clip}^{(2)}, \ldots, V_{clip}^{(v)} | V_{clip}^{(i)} \in \mathbb{R}^{m_x \times m_y \times l_t}\}, \tag{3}$$

where $l_t \leq l_v$ and we let r to be a fixed stride between each clip $V_{clip}^{(i)} \subset V$. The second strategy consists of sampling a fixed set of spatial crops with a temporal length of $T < l_v$. We select 10 crops from four corners and the center of each frame, along with their mirrors. We thus sample a new set $V_S \in \mathbb{R}^{10 \times m_x \times m_y \times T}$ for a given input video V. After a forward pass through the CNN, we obtain our spatiotemporal matrix of descriptors $\mathbf{A} \in \mathbb{R}^{10 \times T \times d_f}$, where d_f is the spatiotemporal feature dimension. Finally, the input to the recurrent network is obtained as follows:

$$x^0 = \{x_t^0 \in \mathbb{R}^{d_f} | t = 1, \ldots, T\}, \tag{4}$$

where

$$x_t^0 = \frac{1}{K_c} \sum_{k=1,\ldots,K_c} \mathbf{A}(k, t, :), \tag{5}$$

where K_c is the number of crops (in our case $K_c = 10$). We found that the mean over the features of crops is preferable to just considering each crop separately.

4 Experimental Results

We will first introduce the datasets, evaluation measures, and implementation details. Then we will discuss the choice of key parameters of the proposed method and compare with both static (non-sequential) and sequential state-of-the-art approaches.

We carried out the experiments on HMDB-51 [29] and UCF-101 [30]. HMDB-51 consists of 6,849 videos of 51 categories, each category containing at least 101 instances. Actions can be categorized into 5 groups: facial actions, facial actions involving objects, body movements, body movements involving objects, and human interactions. UCF-101 provides 13,320 videos from 101 action categories, that can be categorized into: human-object interaction, body-motion only, human-human interaction, playing musical instruments, and sports. Results will be reported in terms of accuracy (%) over the split-1 on both datasets, as in [13].

For the parameter discussion in the next section we used as feature extractor a C3D pre-trained on Sports-1M [13]. Input clips were of temporal length $|S_i| = 16$, with a stride of $r = 8$ between them. For our final architecture features were extracted using the TSN [26] model, and the sparse features were of size 25. For training the residual recurrent network we used the RMSProp optimizer [31] with a learning rate of $\epsilon = 10^{-3}$.

4.1 Discussion on Parameters

In this section we discuss the choice of the hidden layer size, the depth, the duration, and the fusion strategy for the deep residual network.

We varied the hidden layer size h, given the 4,096-dimensional spatiotemporal features extracted by the pre-trained C3D [13]. The best validation performance of the model increased before $h = 1024$ and stagnated after this value (Fig. 3(a)). We therefore used $h = 1024$ as a base parameter to our models.

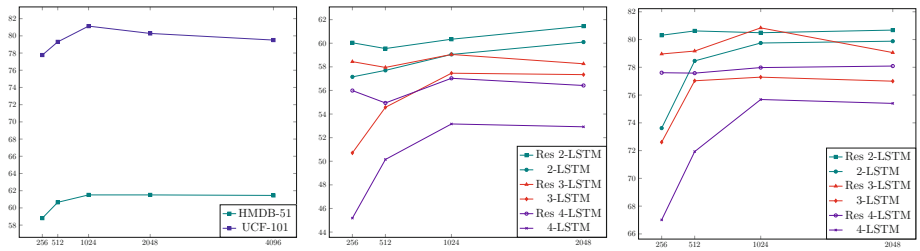


Fig. 3. Influence of size and depth on the performance of the LSTM. (a) Dimension of the hidden layers; (b) residual depth on HMDB-51; (c) residual depth on UCF-101.

A number of works [3,32,33] have shown that longer temporal window over the input of the 3D-CNN input leads to better performance. In Table 1, we can

see that even for LSTM, longer time inputs lead to better performance. However, we may face the vanishing gradient problem using this class of model for very long sequence input. As for the temporal duration, $T = 25$ and $T = 35$[2] are the best choices for HMDB-51 and UCF-101, respectively. After these values the model starts overfitting.

To analyze the impact of the *residual connections* in the stacked recurrent neural networks context, we reduce the dimensionality of the 4,096 input features. The dimensionality of input and output need to match in order to perform the residual addition. To do so, we apply PCA over the initial feature of shape 4,096 (extracted from a pre-trained C3D model) to fit with the dimension of the residual RNN. We select a set of dimensions $\mathcal{D} = \{\mathbb{R}^{d_m} | d_m \in [256, 512, 1024, 2048]\}$, and we train our hierarchical RNN, with and without residual part[3]. Figure 3 shows that the residual connections help generalization in both datasets: even when dropping on performance, the residual RNN still performs better.

We tested stacking 2, 3, and 4 recurrent layers (Fig. 3(b)–(c)): stacking only two layers provides the best depth for the residual model as working with 3D-CNN reduces the number of feature samples per-video and thus a model with more layers is more likely to overfit the data. In contrast, for 2D-CNN feature extraction, the dataset is quite large because each frame is a feature and therefore the residual RNN model will have enough data to tune its parameters for more layers. This is why the authors in [11] were able to train 5-layers RNN.

Table 1. Impact of the value of the time step T on accuracy.

T	5	15	25	35
HMDB-51	59.5	60.2	61.5	61.4
UCF-101	77.9	79.9	79.5	80.9

Finally, late fusion outperforms mid fusion on HMDB-51 using Res-LSTM. For point-wise addition the gain is near 6% and for point-wise product it is 13%, with a clear benefit of the product aggregation "$\odot$". We use the same weights as the original TSN [26], *i.e.*, $(w_1, w_2) \in (1.5, 1)$ (see Table 2).

Table 2. Impact on accuracy of different mid and late fusion strategies on the 2-layer Res-LSTM on the HMDB-51 dataset.

Strategy	Mid fusion	Late fusion
$\oplus$ (element-wise sum)	59.3	65.2
$\odot$ (element-wise product)	56.5	68.0
$w_1.Flow + w_2.RGB$	–	63.3

[2] Note however, that the final scores we report are for $T = 25$ to allow for a fair comparison with the TSN model [26].

[3] We discarded the 4,096 feature vector because of computational requirements.

4.2 Final Model Evaluation

We evaluate our model on coarse UCF-101 categories as well as on complex action classes, following [24]. Table 3 shows that our model outperforms L^2STM in the coarse categories they reported and also in *Mixing Batter* with a gain of 4.41. However, the performance drops for the complex classes *Pizza Tossing* and *Salsa Spins*, probably because of the speed of the actions that our model was not able to capture well. Figure 4 shows the classification of some of the examples with the top confidences for each video example.

Table 3. Comparison on Split-1 of UCF-101 with complex movements.

Data types	L^2STM	Res-LSTM	Gain
Human-object interaction	86.7	88.2	↑ **1.5**
Human-human interaction	95.4	96.9	↑ **1.5**
Body-motion only	88.6	90.7	↑ **2.1**
Playing instrument	-	97.3	-
Sports	-	93.2	-
Pizza tossing	72.7	66.7	↓ 6.0
Mixing batter	86.7	91.1	↑ **4.4**
Playing dhol	100	100	≡
Salsa spins	100	97.7	↓ 2.3
Ice dancing	100	100	≡

Also, we compare our final model to other RNN-like architectures. Table 4 lists the performances and pre-training used by each solution. Our Res-LSTM outperforms the LSTM solutions in HMDB-51, while still being close to L^2STM [24] and Pre-RNN [34] performances on UCF-101.

In addition, Fig. 5 reports the confusion matrices of our model on UCF-101 and HMDB-51. The classes in both datasets are rearranged using the coarse category labels provided in each dataset.

We also combined our method with IDT, following other state-of-the-art approaches [8,14,21]. From the combination, we obtained an improvement of accuracy of, respectively, +0.5% and +8.9% in UCF-101 and HMDB-51. In order to analyze the larger improvement in HMDB-51, we illustrate the confusion matrix for the combination in Fig. 6a and the subtraction of the confusion matrices before and after the combination in Fig. 6b. Finally, Fig. 6c illustrates the per-class accuracy improvement on the HMDB-51 categories after the combination with IDT which improved (or maintained) the accuracy on 45 of the 51 categories, while only getting worse performance on 7.

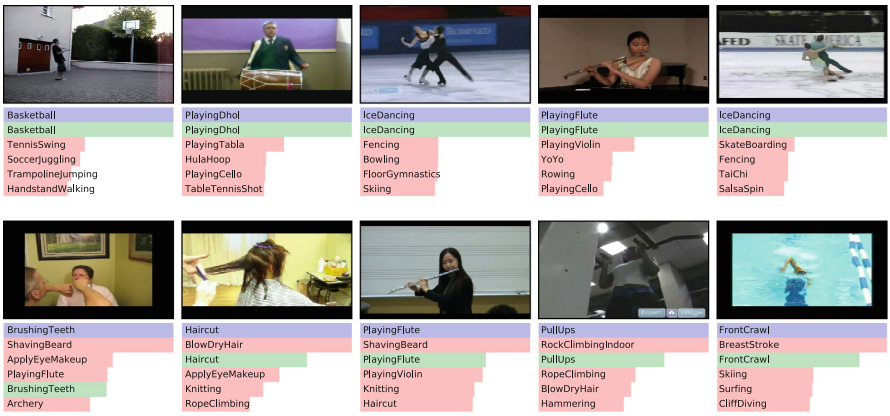
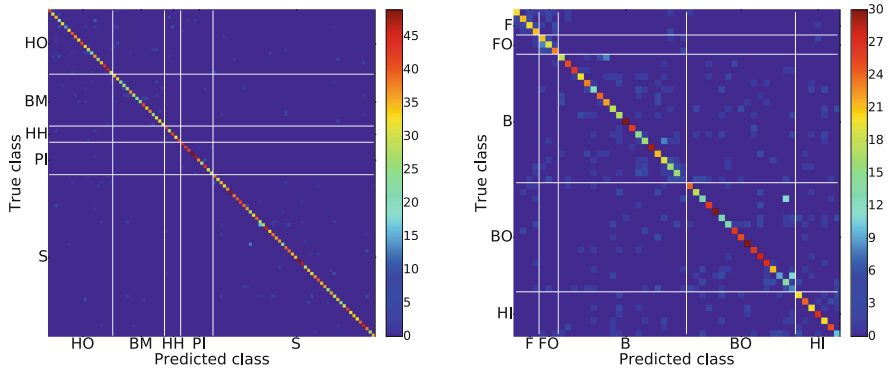


Fig. 4. Sample video classification results showing the top-5 class predictions based on confidence. First row: correctly classified videos; second row: miss-classified videos. Key – blue bar: ground-truth class; green bar: correct class prediction; red bar: incorrect class prediction. (Color figure online)



(a) (b)

Fig. 5. Confusion matrices with rearranged classes to group coarse categories. For UCF-101: human-object interaction (HO), body-motion only (BM), human-human interaction (HH), playing musical instrument (PI), and sports (S). For HMDB-51: facial actions (F), facial actions w/ object manipulation (FO), body movements (BM), body movements w/ object interaction (BO), and body movements for human interaction (HH). (a) Res-LSTM confusion matrix on UCF-101, (b) Res-LSTM confusion matrix on HMDB51

Finally, we compare to a broader set of works, either sequential or non-sequential (static) models in Table 5. Most of them only report results over the three splits in both UCF-101 and HMDB-51. Those that provide the accuracy in Split-1 are marked with '*'.

Table 4. Performance comparison of RNN-like architectures. UCF-101 accuracies are over split-1, except for [24] that only reports accuracy over the three splits. '*' indicates that the method may or not use a pre-trained model, depending on the CNN used.

Method	Pre-training		UCF-101	HMDB-51
	ImageNet	1M Sports		
TwoLSTM [35]	✓	✓	88.3	-
VideoLSTM [21]	✓	✗	89.2	56.4
L^2STM [24]	✓	✗	**93.6**	66.2
Pre-RNN [34]	✓	✗	**93.7**	-
Res-LSTM	*	*	92.5	**68.0**

Table 5. Comparison on UCF-101 and HMDB-51.

Model	Method	UCF-101	HMDB-51
Static models	FST-CNN [36]	88.1	59.1
	TDD [37]	90.3	63.2
	KV-CNN [38]	93.1	63.3
	LTC [39]	91.7	64.8
	TDD + IDT [37]	91.5	65.9
	ST-ResNet [40]	93.4	66.4
	STM-ResNet [41]	94.2	68.2
	LTC + IDT [39]	92.7	67.2
	TSN [26]	94.2	69.4
	ST-ResNet + IDT [40]	94.6	70.3
	STM-ResNet + IDT [41]	94.9	72.2
	STC-ResNext [33]	95.8‡	72.6‡
	I3D [3]	**98.0**	**80.7**
Sequential models	LRCN [42]	82.9	-
	AttLSTM [23]	77.0*	41.3
	UnsuperLSTM [43]	84.3*	44.0‡
	RLSTM-g3 [44]	86.9	55.3
	TwoLSTM [35]	88.3*	-
	VideoLSTM [21]	89.2*	56.4
	VideoLSTM + IDT [21]	91.5*	63.0
	L^2STM [24]	**93.6**	66.2
	PreRNN [34]	**93.7***	–
	Res-LSTM (ours)	92.5*	68.0*
	Res-LSTM (ours) ⊙ IDT	93.0*	**76.9***

‡Only RGB modality is used
*Evaluation on split-1

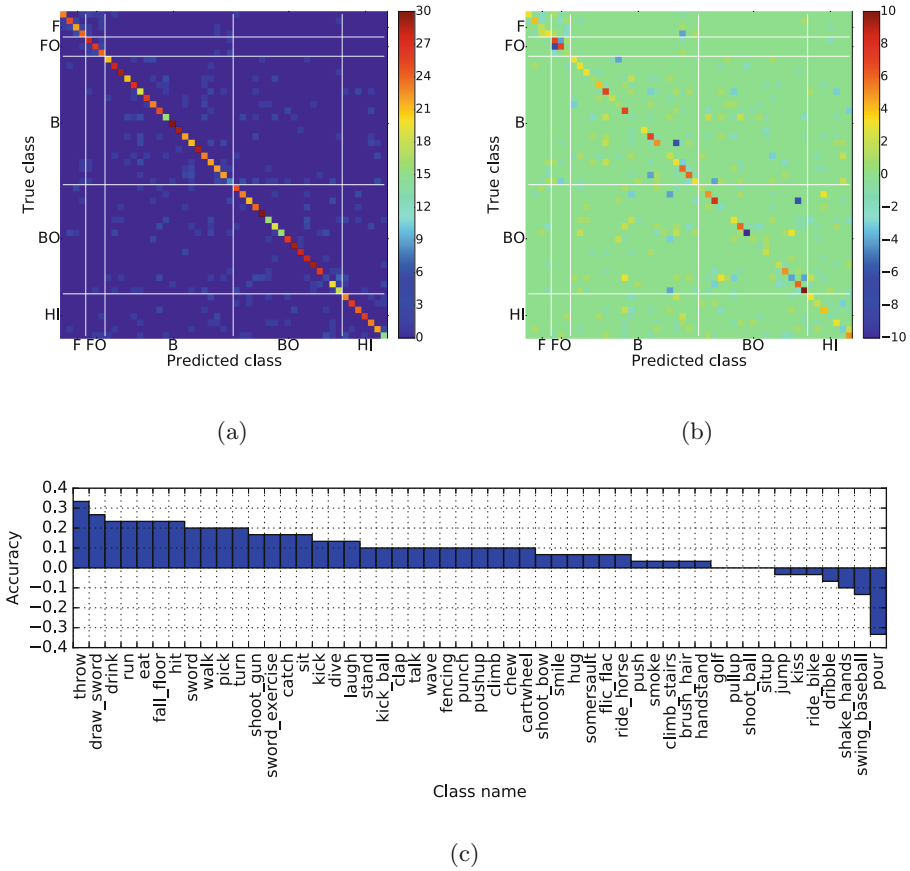


Fig. 6. Confusion matrices and per-class accuracy improvement after combining our Res-LSTM with IDT on HMDB-51. The ordering of classes in (a) and (b) is rearranged as in (a). (a) Res-LSTM ⊙ IDT confusion matrix on HMDB-51, (b) Subtraction of Res-LSTM ⊙ IDT and Res-LSTM confusion matrices on HMDB-51, (c) Per-class accuracy improvement after combining with IDT.

5 Conclusion

We have shown the benefits of the residual connection and of the fusion of appearance and motion features for an action recognition pipeline with a stacked recurrent neural network. Our solution obtained state-of-the-art against LSTM solutions on the HMDB-51 dataset.

Compared to CNN-like state-of-the-art, the RNN based models are still challenging. As future work, we will investigate the joint learning strategy by merging the spatiotemporal features from the 3D CNN directly with the RNN layer in an end-to-end learning pipeline. The end-to-end learning framework between CNN and RNN has been successfully applied to sound classification [45], we believe that this may help to reduce the gap for RNN solution in action recognition problem as well.

Acknowledgements. This work has been partially supported by the Spanish project TIN2016-74946-P (MINECO/FEDER, UE) and CERCA Programme/Generalitat de Catalunya. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU used for this research.

References

1. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: The IEEE International Conference on Computer Vision, ICCV, pp. 3551–3558 (2013)
2. Laptev, I.: On space-time interest points. Int. J. Comput. Vis. **64**(2–3), 107–123 (2005)
3. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2017
4. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725–1732 (2014)
5. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 221–231 (2013)
6. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems, December 2014
7. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1933–1941 (2016)
8. Varol, G., Laptev, I., Schmid, C.: Long-term temporal convolutions for action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **40**(6), 1510–1517 (2018). https://doi.org/10.1109/TPAMI.2017.2712608
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2016
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Ng, J.Y.-H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2015
12. Kim, J., El-Khamy, M., Lee, J.: Residual LSTM: design of a deep recurrent architecture for distant speech recognition. arXiv preprint arXiv:1701.03360 (2017)
13. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. In: IEEE International Conference on Computer Vision, December 2015
14. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: ActionVLAD: learning spatio-temporal aggregation for action classification. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2017
15. Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., Baskurt, A.: Sequential deep learning for human action recognition. In: Salah, A.A., Lepri, B. (eds.) HBU 2011. LNCS, vol. 7065, pp. 29–39. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25446-8_4
16. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. arXiv:1409.1259 (2014)

17. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555 (2014)
18. Asadi-Aghbolaghi, M., et al.: A survey on deep learning based approaches for action and gesture recognition in image sequences. In: IEEE International Conference on Automatic Face & Gesture Recognition, pp. 476–483 (2017)
19. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W., Woo, W.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in Neural Information Processing Systems, December 2015
20. Sudhakaran, S., Lanz, O.: Convolutional long short-term memory networks for recognizing first person interactions. In: IEEE International Conference on Computer Vision Workshops, October 2017
21. Li, Z., Gavrilyuk, K., Gavves, E., Jain, M., Snoek, C.G.: VideoLSTM convolves, attends and flows for action recognition. Comput. Vis. Image Underst. **166**, 41–50 (2018)
22. Xu, K., et al.: Show, attend and tell: neural image caption generation with visual attention. In: International Conference on Machine Learning, pp. 2048–2057 (2015)
23. Sharma, S., Kiros, R., Salakhutdinov, R.: Action recognition using visual attention. arXiv preprint arXiv:1511.04119 (2015)
24. Sun, L., Jia, K., Chen, K., Yeung, D.Y., Shi, B.E., Savarese, S.: Lattice long short-term memory for human action recognition. In: IEEE International Conference on Computer Vision, October 2017
25. Borges, P.V.K., Conci, N., Cavallaro, A.: Video-based human behavior understanding: a survey. IEEE Trans. Circ. Syst. Video Technol. **23**, 1993–2008 (2013)
26. Wang, L., et al.: Temporal segment networks: towards good practices for deep action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 20–36. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_2
27. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
28. Wu, Y., et al.: Google's neural machine translation system: bridging the gap between human and machine translation. CoRR (2016)
29. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: IEEE International Conference on Computer Vision, November 2011
30. Soomro, K., Zamir, A.R., Shah, M.: UCF101: a dataset of 101 human actions classes from videos in the wild. CoRR (2012)
31. Tijmen, T., Geoffrey, H.: Lecture 6.5-RmsProp: divide the gradient by a running average of its recent magnitude. In: COURSERA: Neural Networks for Machine Learning (2012)
32. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 2018
33. Diba, A., et al.: Spatio-temporal channel correlation networks for action classification. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11208, pp. 299–315. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01225-0_18
34. Yang, X., Molchanov, P., Kautz, J.: Making convolutional networks recurrent for visual sequence learning. In: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 2018

35. Ng, J.Y.H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2015
36. Sun, L., Jia, K., Yeung, D.Y., Shi, B.E.: Human action recognition using factorized spatio-temporal convolutional networks. In: Proceedings of the 2015 IEEE International Conference on Computer Vision, ICCV, ICCV 2015, Washington, DC, USA, pp. 4597–4605. IEEE Computer Society (2015)
37. Wang, L., Qiao, Y., Tang, X.: Action recognition with trajectory-pooled deep-convolutional descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition, June 2015
38. Zhu, W., Hu, J., Sun, G., Cao, X., Qiao, Y.: A key volume mining deep framework for action recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 1991–1999, June 2016
39. Varol, G., Laptev, I., Schmid, C.: Long-term temporal convolutions for action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **PP**(99), 1 (2017)
40. Feichtenhofer, C., Pinz, A., Wildes, R.: Spatiotemporal residual networks for video action recognition. In: Advances in Neural Information Processing Systems, pp. 3468–3476 (2016)
41. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal multiplier networks for video action recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, July 2017
42. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. IEEE Trans. Pattern Anal. Mach. Intell. **39**(4), 677–691 (2017)
43. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using LSTMs. In: Blei, D., Bach, F. (eds.) Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, JMLR Workshop and Conference Proceedings, pp. 843–852 (2015)
44. Mahasseni, B., Todorovic, S.: Regularizing long short term memory with 3D human-skeleton sequences for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3054–3062 (2016)
45. Choi, K., Fazekas, G., Sandler, M., Cho, K.: Convolutional recurrent neural networks for music classification. In: IEEE International Conference on Acoustics, Speech and Signal Processing, March 2017