IMPROVING HUMAN ACTION RECOGNITION BY TEMPORAL ATTENTION

Zhikang Liu[1], *Ye Tian*[2], *Zilei Wang*[1]

[1] Department of Automation, University of Science and Technology of China
[2] Institute for Computational and Mathematical Engineering, Stanford University

ABSTRACT

Recently, deep learning methods have been extensively applied for action recognition in videos. Most existing deep networks equally treat every video frame and directly assign a video label to all the frames sampled from it. However, discriminative action may occurs sparsely in a few key frames in a video, and other frames are less relevant or even irrelevant to the action class. Equally treating all the frames will hurt performance. To address this issue, we propose a temporal attention model which learns to recognize human actions in videos while focusing selectively on the informative frames. Our model does not need explicit annotations regarding such informative frames during training and testing. Specifically, we adopt Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) unit and attaches higher importance to the frames which are discriminative for the task at hand. Our method consistently improves on no-attention methods, with both RGB and optical flow based deep ConvNets. We achieve state-of-the-art performance on two challenging datasets of UCF101 and HMDB51.

Index Terms— action recognition, informative frames, temporal attention, RNN, ConvNets

1. INTRODUCTION

Human action recognition in videos has become an active topic and received a lot of research interests [1, 2, 3]. Particularly, in the past several years, action recognition methods taking advantage of deep convolutional networks (ConvNets) have received a great amount of attention. Inspired by the success of image classification using ConvNets [4, 5, 6], researchers attempted developing ConvNet based video analysis algorithms. Some methods took advantage of the image-based ConvNets by directly applying them to sampled video frame [1, 7], while some others tried to learn 3D spatio-temporal convolutional filters for short video clips [8]. In order to represent the entire video sequences, max pooling or average pooling was applied on the top of multiple per-frame or per-clip ConvNets [1, 2, 8, 9].

However, these pooling methods consider all the frames or clips equally and are not able to separate human actions from irrelevant video shots, *i.e.*, the frames or clips that are not correspond to the target action [10, 11]. This is problematic for action recognition from TV and movie material, where an action may be dispersedly portrayed in a video that also contains several video shots for advancing dialog and setting the scene as is observed in [12]. Equally treating all the sampled frames or clips will bring in a lot of noise and hurt performance as noted by many algorithms [10, 12, 13].

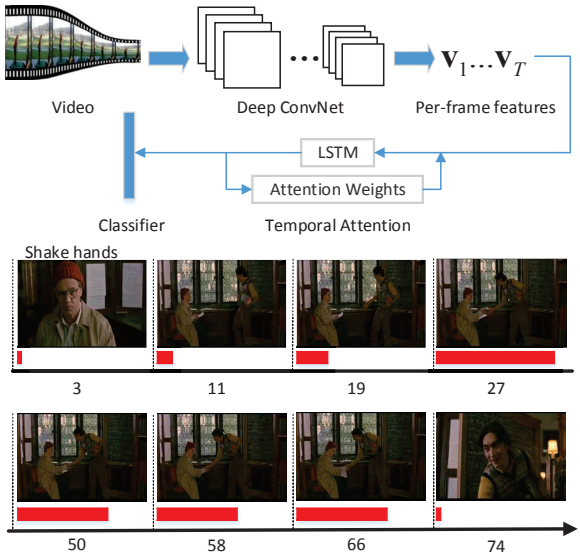

Fig. 1. (Top) Illustration of the high-level overview of our method. Per-frame features $\{\mathbf{v}_1 \ldots \mathbf{v}_T\}$ which are extracted by deep ConvNet are passed through temporal attention model. The temporal attention model assigns weights to the features according to their importance. Class label is generated by the classifier. (Bottom) Predicted attention weights for a video on HMDB51 ran through our temporal attention model. The red bars on the bottom indicate the predicted weights, and the timeline gives the related frame position in the video.

In this paper, we propose a temporal attention model which learns to predict action categories while adaptively select discriminative deep per-frame features. Specifically, our model uses recurrent Long Short-term memory units (LSTMs), and dynamically adjusts attention weights for the input

features by going through multiple iterations. The proposed algorithm is weakly supervised in a way that labels are provided at video-level and not at frame-level [12]. Our model is end-to-end learnable, and no additional inference step is in need during learning. The proposed temporal attention model is able to function with any per-frame or per-clip ConvNets as well as other types of feature representations.

Our contributions can be summarized as below:

(1) We propose a novel temporal attention model for action recognition in videos that is able to select informative frames among the inputs. Our attention model is designed to be fully differentiable, allowing end-to-end training along with the final objective of discriminative classification.

(2) We validate the proposed method on two challenging video benchmark datasets and show that our method consistently outperforms relevant non-attention baselines and achieves state-of-the-art performance when combined with complimentary ConvNet representations of videos.

2. PROPOSED APPROACH

We now describe the proposed approach in detail. As illustrated in Figure 1, our model consists of three components that are connected to each other sequentially. Deep ConvNets extract per-frame features for the input video. Then, the features are sent to the proposed temporal attention model, which recursively adjusts the weights of the features by taking into account their importance which are estimated inside the recurrent network. Softmax classifier is adopted on the top of the temporal attention model to predict the class label. We now describe the feature extraction method and the temporal attention model and thereafter provide details regarding the loss function and the learning procedure.

2.1. Feature Extraction method

We follow the two-stream deep ConvNets [14] fashion to extract both motion and appearance features from video data. We denote a video as

$$Z = [\mathbf{z}_1, \ldots, \mathbf{z}_T], \mathbf{z}_t \in \mathbb{R}^{m \times n \times K}, \quad (1)$$

with each $\mathbf{z}_t$ represents a RGB image ($K = 3$) or a stacked of optical flow images of neighbouring frames [2] (5 frames, $K = 10$ in our experience). Where m and n denote the height and width of the videos respectively. T is the total number of RGB images or stacked optical flow images in the video. Given a training set

$$\mathcal{D} = \{(Z_i, y_i)\}_{i=1}^{N} \subset \mathbb{R}^{m \times n \times K \times T} \times \{1, \ldots C\}, \quad (2)$$

where the Z_i is a training video and y_i is the class label from one of the C possible action categories. The feature extractor consists of all the convolutional layers along with the first fully connected layers (FC-6) of the VGG-16 network [5].

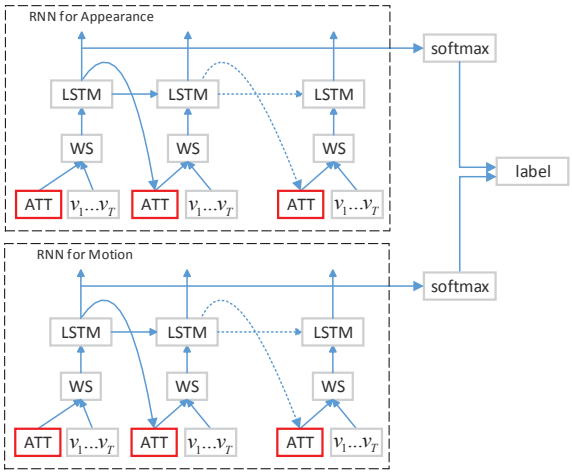

Fig. 2. An illustration of proposed temporal attention model. The block LSTM indicates a LSTM cell. The block ATT represents the attention weight calculating operation. $\mathbf{v}_1...\mathbf{v}_T$ represent the per-frame appearance features or motion features for a video. The LSTM provides the attention weights for the features at each time-step. The weighted sum (WS blocks in the figure) of the features is then given as input to the next iteration allowing the LSTM to adjust the attention weights. Softmax function is applied on the top of our model to generate class label.

This framework is responsible for extracting per-frame features from each $\mathbf{z}_t$, resulting in a fixed dimensional feature vector, denoted as $\mathbf{v}_t = \phi(\mathbf{z}_t) \in \mathbb{R}^{4096}$. In general, Our method can use any deep convolution network [2, 15, 16] for feature extraction.

2.2. Temporal Attention Model

This is the key component of the proposed approach which dynamically weights the per-frame features of a video. It does a temporal scan over the input feature sequence and weight the features by referring to the importance of the current frame. To obtain a comprehensive summarization for the input sequence, we take advantage of the Recurrent Neural Network (RNN) framework with LSTM unit, and the attention weights are adjusted through the iterations of RNN. Figure 2 describes the overall architecture of the proposed temporal attention model. At each iteration, the LSTM takes the entire feature sequence as an input. Next, previous LSTM iteration outputs are used to decide the attention weights for features in an adaptive fashion. Our model learns the weights that model how the previous LSTM output (*i.e.*, the abstraction of the sequence information in the previous round) can give rise to better attention weights in the next iteration.

More specifically, we follow the LSTM implementation

in [17], which is given as follows:

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{g}_t &= \tanh(\mathbf{W}_g\mathbf{x}_t + \mathbf{U}_g\mathbf{h}_{t-1} + \mathbf{b}_g) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned} \tag{3}$$

where $\mathbf{f}_t$ is the forget gate, $\mathbf{i}_t$ is the input gate, $\mathbf{o}_t$ is the output gate, $\mathbf{c}_t$ is the internal memory cell, $\mathbf{h}_t$ is the hidden state at t, $\mathbf{x}_t$ is the input feature at t. σ is the element-wise logistic sigmoid function and $\odot$ is an element-wise multiplication. The main idea of LSTM models lies on the internal memory cell $\mathbf{c}_t$, which records the history of the inputs observed up to t. The forget gate $\mathbf{f}_t$ is to selectively forget the previous memory while $\mathbf{i}_t$ is to selectively accept the current input.

Note that our usage of LSTM is different from previous approaches [1, 18] which adopt LSTM to capture sequential per-frame feature changes. In our case, the goal of each LSTM iteration is to adjust the attention weights to match the video input. Instead of the averaged per-frame features $\mathbf{x}_t = \frac{1}{T}\sum_{i=1}^{T}\mathbf{v}_i$, the input to the LSTM at each time-step t is the dynamic weighted sum of the features such that

$$\mathbf{x}_t = \sum_{i=1}^{T}\alpha_i^t\mathbf{v}_i, \tag{4}$$

where α_i^t is the attention weight for the i-th feature at the time-step t. The relevance score e_i^t for the i-th feature at time-step t is a function of previous iteration LSTM output $\mathbf{h}_{t-1}$ which summarized the video information, and the related per-frame feature $\mathbf{v}_i$:

$$e_i^t = tanh(\mathbf{w}_a^\top\mathbf{h}_{t-1} + \mathbf{u}_a^\top\mathbf{v}_i + b_a) \tag{5}$$

wherein, $\mathbf{w}_a$,$\mathbf{u}_a$ and b_a are the transformation parameters. Once the relevance score e_i^t for all the frames are computed, we normalize them to get the attention weight α_i^t:

$$\alpha_i^t = \frac{\exp(e_i^t)}{\sum_{j=1}^{T}\exp(e_j^t)} \tag{6}$$

through iterations, the temporal attention model allows the LSTM to selectively focus on only a subset of discriminative features by increasing the corresponding attention weight.

For classification, the learnt hidden state $\mathbf{h}_t$ is further fed into the top softmax classifier to generate the predicted label vector $\hat{\mathbf{y}}_t$:

$$\hat{\mathbf{y}}_t = softmax(\mathbf{W}_y\mathbf{h}_t + \mathbf{b}_y) \tag{7}$$

2.3. Parameter Learning

We learn all the parameters of the three portions of our temporal attention model (*i.e.*, the LSTM unit, the attention mechanism, and the top softmax classifier) by minimizing the conditional negative log-likelihood of the training data:

$$\mathcal{L} = -\sum_{t=1}^{D}\sum_{i=1}^{C}\mathbf{y}_{t,i}\log\hat{\mathbf{y}}_{t,i} + \gamma\sum_i\sum_j\theta_{t,j}^2 \tag{8}$$

where $\mathbf{y}_t$ is the label vector, $\hat{\mathbf{y}}_t$ is the class probabilities at time-step t. D is the number of time-steps and C is the number of the action categories. γ is the decay coefficient, and θ represents all the model parameters.

Our temporal attention model is designed to be fully differentiable, and all the parameters can be trained in an end-to-end manner. The gradient can be back-propagated through hidden representations of LSTM and the attention mechanism successively using back-propagation through time algorithm [19].

3. EXPERIMENTS

3.1. Datasets and evaluation protocal

We conduct experiments on the UCF101 [20] dataset and the HMDB51 [21] dataset. These two datasets are among the largest available human action recognition datasets. Specifically, the UCF101 is composed of 101 action categories and there are at least 100 video clips in each category. The whole dataset contains $13,320$ video clips. HMDB51 contains $6,700$ videos from 51 action categories, and each category has at least 100 videos. For both datasets, we use the training/testing splits provided by the corresponding organizers. The performance is measured by the mean of accuracies across all the splits in each dataset.

3.2. Implementation Details

We use the training data in UCF101 to train VGG-16 two-stream ConvNets for UCF101 dataset. The spatial net is built on single frame images $(224 \times 224 \times 3)$ therefore its architecture is the same as the VGG-16 [5] in image domain. The input to the temporal net is 5 frames stacking of optical flow images $(224 \times 224 \times 10)$ and thus the convolutional filters in the first layer are $(3 \times 3 \times 10)$. The training procedure is similar to [14]. We use mini-batch stochastic gradient descent with momentum (0.9). For spatial net, the learning rate starts at 0.001, decreases to its $1/10$ every $4,000$ iterations, stops after $10,000$ iterations. For temporal net, the initial learning rate is 0.005, decreases to its $1/10$ every $10,000$ iterations, stops after $30,000$ iterations. The parameters of the spatial net and the temporal net were initialized from a pre-trained ImageNet model and then fine-tuned on UCF101. We use a similar procedure to train the VGG-16 feature extractors for HMDB51. Theano toolbox is adopted for RNN and temporal attention mechanism implementation and the model is trained by using Adadelta [22]. For both training and testing, our

model takes 30 successive frames at a time, *i.e.*, each video in the datasets gets split into multiple 30-length video samples. We use two stacked LSTM Layers, each with 512 memory cells. Following the LSTM layers, a softmax classifier makes a prediction at every time-step.

In test, we give the video samples as input to the ConvNet feature extractors to get appearance features and motion features. The appearance features and the motion features loop 30 time-steps in the RNN for appearance and the RNN for motion, respectively. Prediction scores of the last 10 time-steps of both RNN are weighted average ($1/3$ for appearance stream, $2/3$ for motion stream) to generate the sample-level prediction. The class scores for the whole video are then obtained by averaging the scores across all the samples.

3.3. Evaluation of the temporal attention

Comparison with baselines. As the baselines, we test our model without the temporal attention mechanism, *i.e.*, we rewrite the input to LSTM unit in Equation 4 as $\mathbf{x}_t = \frac{1}{T}\sum_{i=1}^{T}\mathbf{v}_i$. We evaluated the performances of (1) individual appearance stream with RNN, (2) individual motion stream with RNN, and (3) two-stream framework with RNN. Table 1 compares our results with baselines on UCF101 and HMDB51. It can be seen that our temporal attention model can improve baseline performance on both datasets.

Table 1. Comparison with baselines on UCF101 and HMDB51.

Model	UCF101	HMDB51
appearance stream baseline	83.4%	52.3%
appearance stream attention	**84.2%**	**54.2%**
motion stream baseline	88.5%	60.7%
motion stream attention	**90.4%**	**62.9%**
two-stream baseline	91.6%	63.3%
two-stream attention	**93.3%**	**65.0%**

Comparison with exist video attention methods. Table 2 compares our results with existing attention methods for action recognition. Our temporal attention model achieves better performance than exist attention methods. Although our model is much simpler than joint attention [23], our method outperforms it a lot, especially on the HMDB51 dataset. Our method significantly surpasses the inefficiency spatial attention methods [24] [25] as well. This could be the case since spatial attention-based methods have no ability to find out irrelevant video frames.

Comparison with the state-of-the-art methods. Table 3 compares our results with several state-of-the-art methods on UCF101 and HMDB51 datasets. We first compare with hand-crafted features like Improved Dense Trajectories (iDT) [3] or MIFS [26]. Our proposed temporal attention model outperforms all these methods on these two datasets. Then, we perform comparison with deep learning methods such as DeepNet [9] and two-stream CNN [25]. We see that our model achieves performance competitive with the current state-of-the-art methods.

Table 2. Comparison with existing attention methods on UCF101 and HMDB51.

Model	UCF101	HMDB51
Spatial appearance attention [24]	N/A	41.3%
Joint attention [23]	90.6%	61.7%
Spatial attention [25]	92.7%	64.3%
Our method	**93.3%**	**65.0%**

Table 3. Comparison with state-of-the-art methods on UCF101 and HMDB51.

Model	UCF101	HMDB51
iDT [3]	85.9%	57.2%
iDT+HSV [27]	88.0%	61.1%
MIFS [26]	88.5%	63.8%
Two-steam CNN [25]	88.0%	59.4%
DeepNet [9]	65.4%	N/A
3D Convolutional [8]	83.4%	53.9%
Key-volume mining CNN [13]	93.1%	63.3%
CNN+LSTM [1]	88.6%	N/A
Deep two-stream [14]	91.4%	N/A
Our method	**93.3%**	**65.0%**

4. CONCLUSION

In this paper, we propose a temporal attention model for action recognition in videos. We aim to make deep ConvNets focus on the most discriminative frames. Our temporal attention mechanism is designed to be differentiable, allowing end-to-end learning with underlying RNN model. As demonstrated by the experimental results on two challenge datasets, our model is able to improve the two stream frameworks significantly and achieve state-of-the-art performance.

Acknowledgement. This work is supported partially by the National Natural Science Foundation of China under Grant 61673362 and 61233003, Youth Innovation Promotion Association CAS, and the Fundamental Research Funds for the Central Universities.

5. REFERENCES

[1] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici, "Beyond short snippets: Deep networks for video classification," in *CVPR*, 2015.

[2] Karen Simonyan and Andrew Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NIPS*, 2014.

[3] Heng Wang and Cordelia Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[5] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[6] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[7] Mihir Jain, J Gemert, Cees GM Snoek, et al., "University of amsterdam at thumos challenge 2014," 2014.

[8] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *ICCV*, 2015.

[9] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[10] Basura Fernando, Efstratios Gavves, José Oramas, Amir Ghodrati, and Tinne Tuytelaars, "Rank pooling for action recognition," *TPAMI*, 2016.

[11] Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould, "Dynamic image networks for action recognition," in *CVPR*, 2016.

[12] Yang Wang and Minh Hoai, "Improving human action recognition by non-action classification," *CVPR*, 2016.

[13] Wangjiang Zhu, Jie Hu, Gang Sun, Xudong Cao, and Yu Qiao, "A key volume mining deep framework for action recognition," in *CVPR*, 2016.

[14] Limin Wang, Yuanjun Xiong, Zhe Wang, and Yu Qiao, "Towards good practices for very deep two-stream convnets," *arXiv preprint arXiv:1507.02159*, 2015.

[15] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu, "3d convolutional neural networks for human action recognition," *TPAMI*, 2013.

[16] Limin Wang, Yu Qiao, and Xiaoou Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *CVPR*, 2015.

[17] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[18] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[19] Paul J Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, 1988.

[20] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[21] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre, "Hmdb: a large video database for human motion recognition," in *ICCV*, 2011.

[22] Matthew D Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[23] Jialin Wu, Gu Wang, Wukui Yang, and Xiangyang Ji, "Action recognition with joint attention on multi-level deep features," *arXiv preprint arXiv:1607.02556*, 2016.

[24] Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov, "Action recognition using visual attention," in *NIPS Time Series Workshop*. 2015.

[25] Yilin Wang, Suhang Wang, Jiliang Tang, Neil O'Hare, Yi Chang, and Baoxin Li, "Hierarchical attention network for action recognition in videos," *arXiv preprint arXiv:1607.06416*, 2016.

[26] Zhengzhong Lan, Ming Lin, Xuanchong Li, Alex G Hauptmann, and Bhiksha Raj, "Beyond gaussian pyramid: Multi-skip feature stacking for action recognition," in *CVPR*, 2015.

[27] Xiaojiang Peng, Limin Wang, Xingxing Wang, and Yu Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *CVIU*, 2016.