# ARNets: Action Recurrent Networks for Human Action Recognition


Guangjun Zhang
School of Computer Science and Technology, Beijing Institute of Technology, China
zgj18811414604@163.com

Xiaobo Cai
Shenzhen Youzhichuangxin Technologies Co., Ltd., China
caixiaobo@utarn.com

Guangyu Gao*
School of Computer Science and Technology, Beijing Institute of Technology, China
guangyu.ryan@gmail.com

Zihua Yan
Shenzhen Youzhichuangxin Technologies Co., Ltd., China
yanzhihua@utarn.com

Liang Shu
Shenzhen Youzhichuangxin Technologies Co., Ltd., China
shuliang@utarn.com

Zhihui Hu
Shenzhen Youzhichuangxin Technologies Co., Ltd., China
huzhihui@utarn.com



## ABSTRACT

Recurrent neural networks (RNNs) are powerful and robust neural networks that capture time dynamics via cycles in the graph. Compared to Convolutional Neural Networks (CNNs), RNNs consider both the current and the historical inputs which leads them to the standard approach for practitioners to address machine learning tasks involving sequential data. Compared to the achievements in these sequential learning tasks, RNNs pale in presenting a promising performance on human action recognition. Do we miss something? In this paper, we will conduct a detailed investigation of the RNNs for the human action recognition task. Several alternative recurrent architectures are designed and compared. Even simply, we find that adding the supervision signal at the end of an RNN can provide us with a comparable if not better performance than the previous state-of-the-art. We further analyze the different combinations of modalities and the effect of the length of the sequence. Compared to CNN-based recognition architecture, RNNs focus more on the action itself. Going deeper, we also find that when modeling long and difficult actions, RNNs perform much better than CNNs. It means a complex video action dataset may be required.



## CCS CONCEPTS

• **Computing methodologies**; • **Artificial intelligence**; • **Computer vision**; • **Computer vision tasks**; • **Activity recognition and understanding**;

## KEYWORDS

action recognition, recurrent networks, visualization



*Corresponding Author.





*ICIGP 2023, January 06–08, 2023, Chongqing, China*





ACM ISBN 978-1-4503-9857-2/23/01…$15.00
https://doi.org/10.1145/3582649.3582668

**ACM Reference Format:**
Guangjun Zhang, Xiaobo Cai, Guangyu Gao, Zihua Yan, Liang Shu, and Zhihui Hu. 2023. ARNets: Action Recurrent Networks for Human Action Recognition. In *2023 The 6th International Conference on Image and Graphics Processing (ICIGP 2023), January 06–08, 2023, Chongqing, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3582649.3582668


## 1 INTRODUCTION

In human action recognition problems, besides the spatial information each independent frame provides, the temporal consistency between the frames matters as well [22]. Practically, people apply the 2D Convolutional Neural Networks (CNNs) on frames within the video regularly and assemble the prediction from each frame for the final video recognition. Obviously, we miss the temporal information of the video and prediction from a single randomly sampled video frame can be significantly disturbed by viewpoint variations and background. In consequence, two-stream CNN [8] is proposed to learn motion patterns by applying another CNN to optical flows. Combining the spatial information from RGB frame and at the same time, the temporal information from optical flows, much better performance on human action recognition can be achieved.

Compared to 2D CNNs, 3D CNNs do consider a sequence of images by convolving along the temporal domain. However, in our opinion, this is a simple combination rather than a reasoning. 3D CNNs do well in combining the information both from spatial and temporal domain by using 3D convolution. However, in the action recognition tasks, reasoning is essential.

Do we have models or algorithms for the sequential reasoning? Recurrent neural networks (RNNs) are a type of neural network designed to recognize patterns in sequences of data. RNNs can use their internal memory to process sequences of inputs. Crucially, the output of the recurrent networks at each time step are influenced not only by the input you just fed in, but also on the entire history of inputs you have fed in in the past. Among RNNs, Long Short-Term Memory (LSTM) performs best. Indeed, RNNs have already been applied to human action recognition task [9, 10, 20], nevertheless, their capability for video classification has not been fully explored. The illustration of 2D CNNs, 3D CNNs and RNNs for human action recognition is shown in Figure 1.

The contributions of this paper are summarized below:

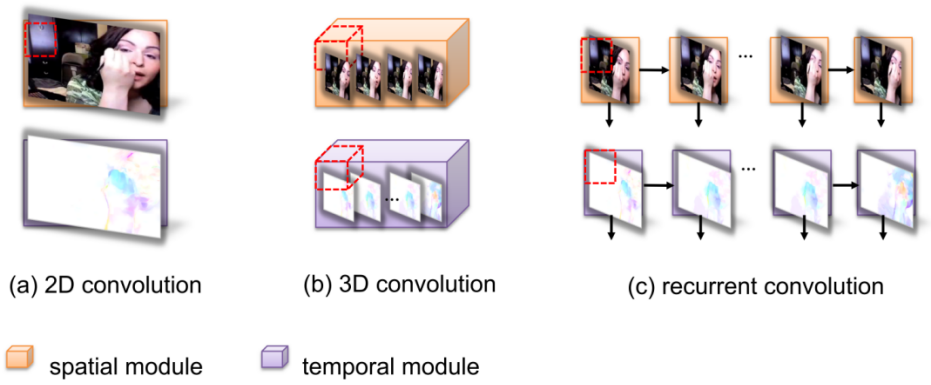



**Figure 1: Illustration of 2D CNNs, 3D CNNs and RNNs for action recognition**

- We propose a general architecture, that is Action Recurrent Networks (ARNets) for human action recognition using recurrent networks.
- Based on the ARNets, we implement several alternative designs. At the same time, quantitative and qualitative investigations of these recurrent architectures is provided for human action recognition tasks.
- Extensive quantitative and qualitative experimental results are presented to analyze the property of ARNets.
- Interestingly and surprisingly, simple recurrent networks can achieve state-of-the-art performance on popularly benchmark datasets. ARNets show their capabilities in modeling complex videos.

## 2 RELATED WORK

By processing each frame within the video using a CNN classifier such as VGG or ResNet which is pre-trained on ImageNet, the action recognition problem can be cast into an image classification problem. The final prediction is derived by averaging the probability from each frame. This kind of method could not get good performance because it just looks at one frame once independently. On the contrary, human actions in videos are three-dimensional (3D) spatio-temporal signals. Therefore, [5] proposes to use a two-stream architecture to handle the spatio-temporal information simultaneously. In this design, the spatial network is processing RGB frames and at the same time, the temporal network is processing optical flows, the probability will be averaged or weighted averaged at the end. RGB images and optical flows provide the spatial and temporal information, respectively, therefore, better performance can be achieved using this architecture [1, 5, 6, 15].

Recent attempts have been made to extend 2D CNNs to 3D CNNs to incorporate temporal information. [12] directly extends 2D CNNs to 3D CNNs for action recognition. The Inception 3D (I3D) [1] is one of the latest state-of-the-art models. They inflate all the 2D convolution filters by duplicating weights that were pretrained on ImageNet classification over the temporal dimension into 3D convolutions. [21] use Fourier temporal features of skeleton joints for recognition. mathematic based method like discrete cosine transform [23] are proposed for action recognition.

Some RNN-style two-stream networks have been proposed for action recognition. VideoLSTM [6] applies convolutional operations within LSTM on sequences of images or feature maps. [20] use siamese LSTM with metric learning for action recognition. Additionally, an attention model is stacked on top of the ConvLSTM [16] to further refine the temporal features. Sun et.al [10] also propose a lattice LSTM for the long and complex temporal modeling. Even attracting and promising, these two-stream LSTMs do not beat previous none RNNs methods and no detailed investigations are provided.

## 3 ARNETS: ACTION RECURRENT NETWORKS

LSTM units [3] are recurrent modules that have the capability of learning long-term dependencies. Different from vanilla RNNs, usually, an LSTM contains a cell memory to remember, an input, a forget and an output gate to control. Due to its superior performance, we adopt convolutional LSTM (ConvLSTM) [16] as the basic unit for Action Recurrent Networks (ARNets). To simplify the presentation, in the rest of the paper, the abbreviation LSTM instead of ConvLSTM will be used. Formally, let's denote $X_t$ as two input images at time $t$. Starting from the cell memory $\tilde{C}_t$ at time $t$ which maintains the information over time,

$$\widetilde{C}_t = tanh\left(W_{xc} * X_t + W_{hc} * H_{t-1}\right) \tag{1}$$

where $W_{xc}$ and $W_{hc}$ are, respectively, the weights for the input and hidden states. The symbol $*$ denotes the convolution operation and $H_{t-1}$ is the hidden output at time step $t-1$.

The input gate $i_t$ and forget gate $f_t$ at time step t are computed as

$$i_t = \sigma\left(W_{xi} * X_t + W_{hi} * H_{t-1}\right), \quad f_t = \sigma\left(W_{xf} * X_t + W_{hf} * H_{t-1}\right) \tag{2}$$

where $W_{x.}$, $W_{h.}$ are distinct weights for the input gate and forget gate. At the end of time step t, we can obtain the updated memory cell $C_t$ from the previous memory cell $C_{t-1}$ and $\tilde{C}_t$:

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t \tag{3}$$

where '$\circ$' denotes the Hadamard product. The input gate and forget gate together determine the amount of dynamic information entering/leaving the memory cell. The final hidden output $\boldsymbol{H}_t$ is

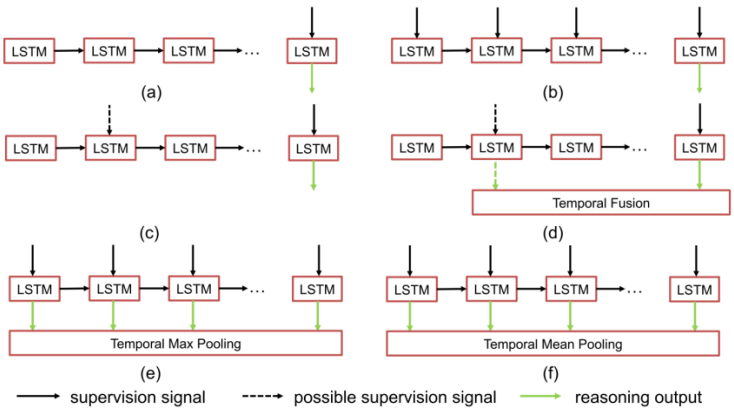



**Figure 2: Illustration of recurrent designs for sequential modeling.**

controlled by output gate $o_t$,

$$o_t = \sigma\left(W_{xo} * X_t + W_{ho} * H_{t-1}\right),\ H_t = o_t \circ tanh\left(C_t\right) \quad (4)$$

where $W_{xo}$, $W_{ho}$ are distinct weights for the output gate.

In the recurrence, we already obtain a sequence of reasoning representations as $\boldsymbol{H} = \{\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots, \boldsymbol{H}_t\}$. The post-processing functions $g$ will be applied on top of the sequence $\boldsymbol{H}$ thereafter.

$$\hat{\boldsymbol{H}} = g\left(\boldsymbol{H}\right) \quad (5)$$

$g$ can be the temporal maximum function, temporal mean function or even simple selective function.

Then SoftMax function takes an N-dimensional vector and transforms it into a vector of real number in range (0,1) which add up to 1, where $N$ is the number of categories.

$$p_i = \frac{e^{H_i}}{\sum_j e^{H_j}} \quad (6)$$

$L_t$ is the cross-entropy loss at time $t$, while $y$ denotes the correct target class number:

$$L_t = -\sum\nolimits_i y_i \log p\left(t, i\right) \quad (7)$$

## 3.1 Alternative Architecture Designs

In ARNets, the function $g$ can be any functions, here we provide six alternative designs for $g$ as shown in Figure 2. (a), (b), (c), (d), (e), (f). Solid black arrow is the supervision signal, solid green arrow is the reasoning signal and dash black arrow is the optional supervision signal. (a): the supervision signal is only added to the last time step. Only the last output is used for reasoning. (b): the supervision signal is added to each time step. Only the last output is used for reasoning. (c): the supervision signals are added to the last time step and a randomly chosen previous step. Only the last output is used for reasoning. (d): the supervision signals are added to the concatenated representations of the last time step and a randomly chosen previous step. The output from the last and chosen time step is concatenated for reasoning. (e): the supervision signals are added to the temporal max pooled representations of all time steps. The output from temporal max pooled representations of all time steps is used for reasoning. (f): the supervision signals are added to the temporal average pooled representations of all time steps. The output from temporal average pooled representations of all time steps is used for reasoning. In the following paper, we will provide the detailed results of these designs quantitatively and qualitatively.

# 4 EXPERIMENTS

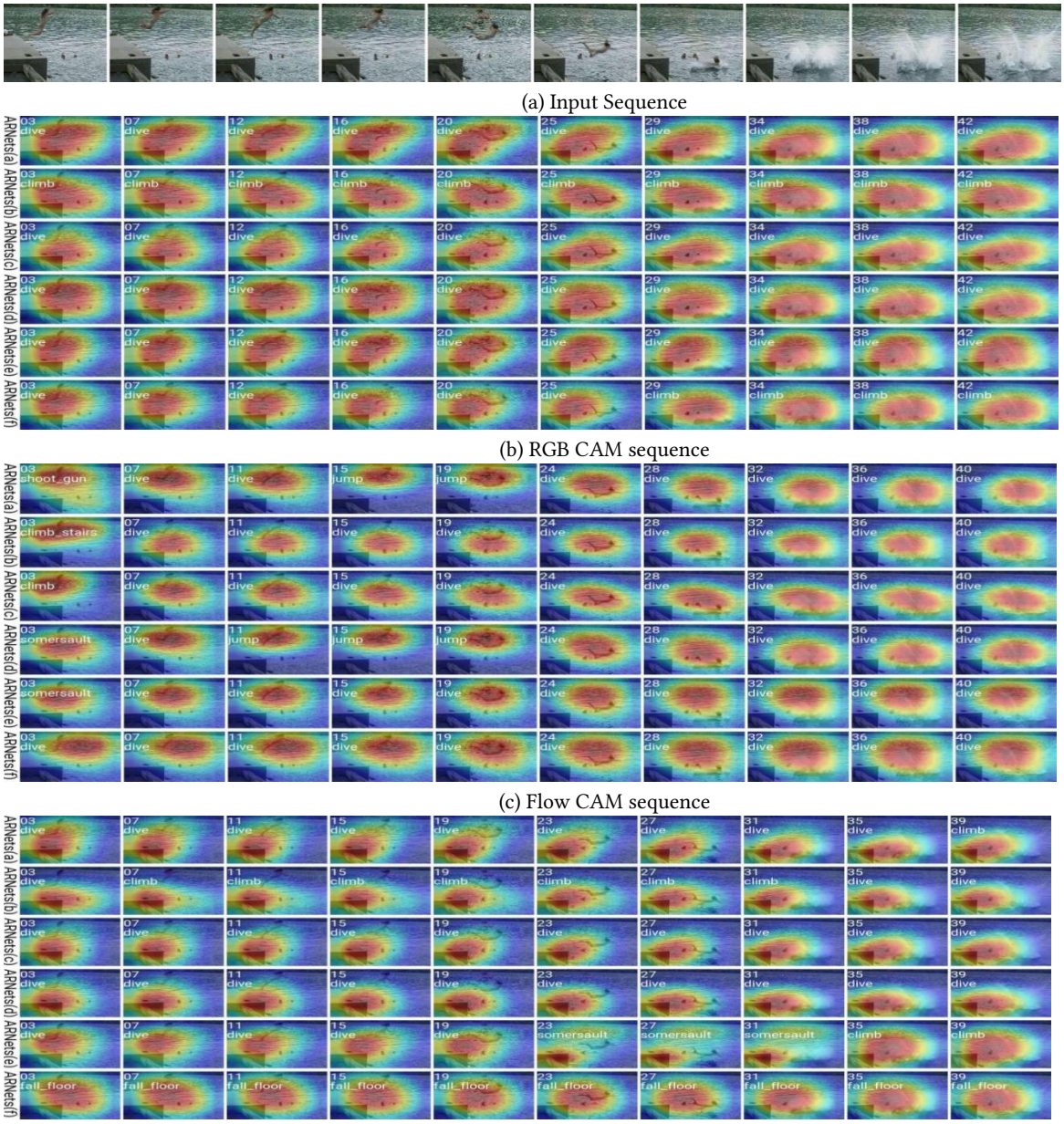
We apply our proposed methods on several large-scale human action recognition benchmark datasets, *e.g.*, UCF-101, HMDB-51, Kinetics, Something in Something, Moments in Time.

## 4.1 Implementation Details

In order to obtain the fair comparison, like TSN [15], BNinception [4] is used as backbone networks, and features from the last convolutional layers will be fed into the RNNs. RGB frames and corresponding optical flows, and RGB differences will be passed through this backbone simultaneously. For each RNN, it is a two-layer LSTM. Within convolutions, all the kernels are $3 \times 3$ and the number of hidden feature maps is 512. Batches of ten sequential frames will be fed into system for training. The whole system can be trained end-to-end using SGD. The initial learning rate for RNN is 1e-2 and for backbone is 1e-3. The momentum is set to 0.9 and weight decay is 5e-4. When testing, we regularly sample four clips and average their probabilities.

## 4.2 Performance of Alternative Designs

We evaluate the effect of different $g$ with six alternative designs on split 1 of UCF-101 and HMDB-51, as shown in Table 1. Surprisingly, when the $g$ is just a simple function by selecting the output from the last step, it achieves the best performance. So, in later analysis, we use ARNet-a as our model. Except for the individual performance of RGB, optical flow and RGB difference, we also provide the performance of their combination. We can see that both pairs can provide an attractive performance. Overall performance is the combination of RGB, optical flow and RGB difference. Interesting to see that ARNet can take advantage of RGB, optical flows and RGB difference which is different from the statement in TSN [15] that RGB difference is distracting the whole performance.




(a) Input Sequence

(b) RGB CAM sequence

(c) Flow CAM sequence

(d) RGBDiff CAM sequence

**Figure 3: Illustration of CAM for alternative design in Figure 2**

## 4.3 Temporal Focusing Analysis on Alternative Designs

In this section, we illustrate what the network focuses on in temporal inference using class activation maps (CAM) [19]. CAM is generated for each input frame of all alternative designs. So, we can visualize the recognition process by generating CAMs in sequence and explore the properties of ARNets. As shown in Figure 3, the white number on the CAM picture is indexing the frame sampled from the original video and the white text is the prediction made by our model. In ARNet-a, the focusing region is all the action regions that happened in the videos.

With the supervision signal added to each time step, the model trained by ARNet-b becomes more concentrated and only 'watch' the middle of the frame. ARNet-c behaves similarly to ARNet-b. ARNet-d presents that combining the information from the previous can somehow distract the current prediction. The prediction from ARNet-d becomes dilated when the time increase. ARNet-e and ARNet-f have a max and mean temporal operation, which produces

**Table 1: Performance evaluation (%) on split 1 of UCF-101 and split 1 of HMDB-51**

| | UCF-101/HMDB-51 | | | | | |
|---|---|---|---|---|---|---|
| Training setting | S-Net | T-Nets (Flow) | T-Nets (Diff) | S-Nets + T-Nets (Flow) | S-Nets + T-Nets | Overall |
| Clarifai [5] | 72.7/40.5 | 81.0/54.6 | 87.0/- | 87.0/- | -/- | -/- |
| VGGNet-16 [15] | 79.8/- | 85.7/- | 89.0/- | 90.9/- | -/- | -/- |
| BN-Inception [15] | 84.5/- | 87.2/- | 83.8/- | 92.0/- | 87.3/- | 91.7/- |
| BN-Inception+TSN [15] | 85.7/54.4 | 87.9/62.4 | 86.5/- | 93.5/69.5 | -/- | -/- |
| ARNet-a | 84.4/56.7 | 87.1/65.4 | 88.1/60.9 | 93.2/67.8 | 91.6/64.4 | 94.9/70.7 |
| ARNet-b | 81.6/47.5 | 86.6/57.5 | 85.9/50.9 | 90.9/60.0 | 88.7/54.5 | 91.6/63.0 |
| ARNet-c | 83.6/51.6 | 87.7/62.8 | 87.5/56.2 | 91.9/65.1 | 89.4/60.3 | 92.6/66.4 |
| ARNet-d | 83.4/55.5 | 87.4/64.0 | 87.5/60.8 | 91.8/67.0 | 89.6/63.7 | 92.8/70.1 |
| ARNet-e | 83.3/51.6 | 85.8/59.0 | 87.3/56.9 | 91.2/65.9 | 88.3/61.7 | 92.5/67.7 |
| ARNet-f | 83.1/53.9 | 86.9/61.0 | 86.6/57.5 | 91.0/66.1 | 89.1/61.5 | 91.9/67.8 |

a generally high density and confident prediction along the whole sequence.

Besides the above observation for each design, we find that the CAM generated from the RGB sequence is very different from that generated from the RGB difference and optical flow. CAMs from the temporal information (optical flows or RGB difference) have denser hot areas focusing on body movements. In particular, in Figure 3(c), ARNet-a the focusing attention follows the jumping person perfectly. On other side, it proves that optical flow or RGB difference can provide more accurate prediction which provides complementary information to RGB frames.

## 4.4 More Investigation About Recurrence

Since the parameters are shared between different time steps. After training the ARNets, theoretically we can feed any number of frames into our networks. In this section, we will investigate the relationship between the number of frames fed into the system and prediction accuracy.

It is common sense that we can obtain more reliable reasoning for action when we can see more frames. However, we never or cannot explore the effect of frames in traditional neural networks. On one hand, it may not consider the temporal relationship. On the other hand, it may not allow changing the number of frames feeding into the network after the training is completed. Therefore, in this research, we feed the variable number of frames (3-16) into our ARNets, and test on RGB, RGBDiff and flow model. Experiments show that for RGB and RGBDiff model, the recognition accuracy increases quickly when the frame gradually feeds into the system (3-10), then after 10 frames it is getting stable. When more frames are fed in, similar if not better performance can be achieved. The flow model behaves quite interestingly, accuracy increases from 3 to 11 frames, but then drop to 60% with more input. The accuracy drop may be caused by the long-term video dependence, the short-term optical flow may break at a certain length and can provide an accurate prediction.

Moreover, the intermediate result also shows ARNets try to reason the prediction progressively. Figure 4 shows this progressive prediction. We sampled frames from origin video with a step of 5, thus frame 3, 8, 12, 17, 22, 26, 31, 36, 40, 45 are fed into our ARNets. The upper left word of the figure donates the frame number and predicts the result based on frames from the first input frame to the latest frame. From the beginning, frame 3 (the first picture in Figure 4) shows two people standing together, so our model 'thinks' it is a *stand* action, then more frames arrive, the two people start running, punching each other, and finally the right person *fall floor*. Our model follows this natural action sequence and progressively predicts the best fitted result, that is *fall floor*.

In order to vividly visualize the prediction step by step, we draw the representation of the ARNet using t-SNE [14] in sequential manner in Figure 5. When we only get one frame, the category prediction is a little random and all the category is intertwined with each other. When the time step increase, more reliable reasoning can be achieved. Therefore, videos from the same category will be concentrated and different categories can be well separated. The t-SNE sequence illustrate the progressive process of a RNN model processing behavior, as the network digesting the temporal information, more accurate result can be provided. We believe that long-frame capacity make our ARNets easy to recognize complex actions.

## 4.5 TSN vs. ARNets

We discuss other different results that vary with modality. Different action representations, such as RGB, RGBDiff, and Flow, may have different contributions to recognition accuracy. Since RGB representation is more similar to humans, Flow representation is more abstract, since its data are generated from two continuous frames. To examine the difference between these representations in our ARNets and TSN, we analyze those modalities' top 5 hard classes accuracy in HMDB51. Figure 6 shows the top 5 hard class accuracy. Firstly, we select the top 5 hard classes from each modality, then we picked the most 6 common classes and draw their accuracy together. Our ARNets common hard classes is *smile*, *swing baseball*, *run*, *throw*, *walk* and *turn*. For TSN its common hard classes are very unbalanced through modality, which is *throw*, *hit*, *catch*, *pick*, *sword* and *swing baseball*. TSN's unbalance result shows its data representation depended on characteristics, and for the hardest classes, the overall accuracy is also much higher than TSN. Our model is better in dealing with different kinds of action representations, and give

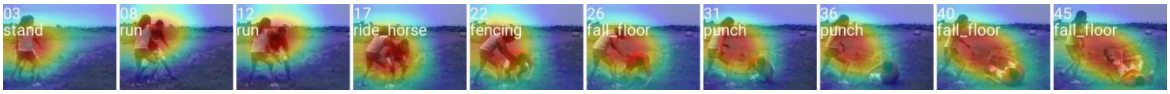
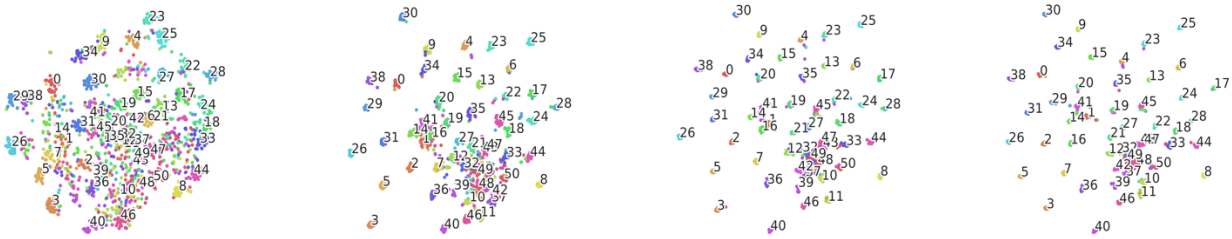



**Figure 4: Progressive temporal digestion**



(a) Frame1 (b) Frame 4 (c) Frame 7 (d) Frame 10

**Figure 5: T-SNE visualization of action recognition process with video frame from 1 to 10.**

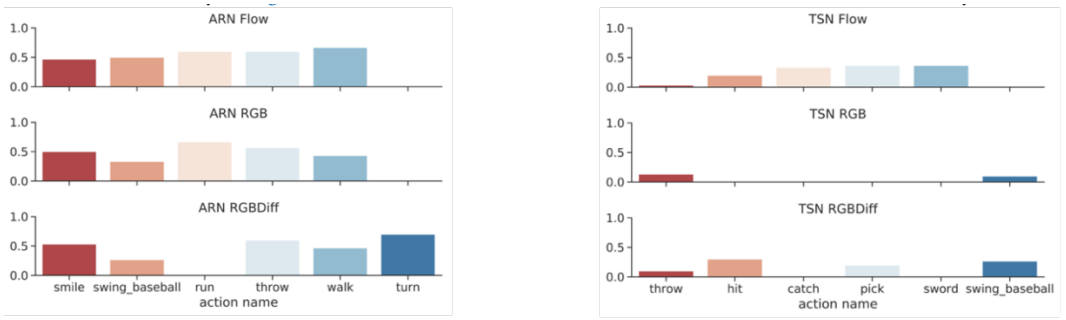



**Figure 6: shows the top 5 hard classes in ARNets and TSN based on Flow classes.**

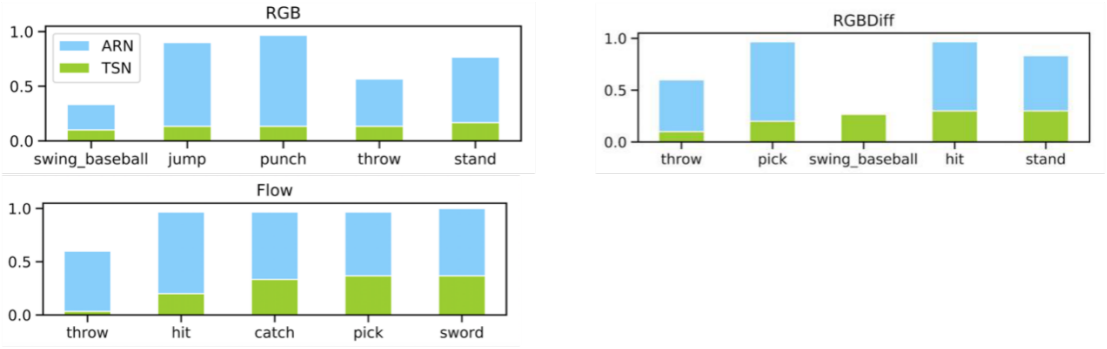



**Figure 7: Labels corrected by our ARNets. The blue part means labels are corrected by our ARNets which is wrongly predicted by TSN.**

stable result and better performance not only in overall accuracy, also keep stable in single class performance.

Compared to state-of-the-art TSN, our ARNets gain a high accuracy in classes which is difficult for TSN. Top 5 Hard classes accuracy in Figure 7. Our model has more than half the hard classes of TSN's accuracy are nearly 90%.

**Table 2: Mean accuracy on the UCF-101 and HMDB-51 datasets**

| UCF-101 | | HMDB-51 | |
|---|---|---|---|
| EMC-CNN [17] | 86.4 | EMV-CNN [17] | - |
| Two Stream [5] | 88.0 | Two Stream [5] | 59.4 |
| $F_{ST}CN(SCI\ Fusion)$ [11] | 88.1 | $F_{ST}CN(SCI\ Fusion)$ [11] | 59.1 |
| C3D (3 nets) [12] | 85.2 | C3D (3 nets) [12] | - |
| Feature amplification [8] | 89.1 | Feature amplification [8] | 54.9 |
| VideoLSTM [6] | 89.2 | VideoLSTM [6] | 56.4 |
| TDD+FV [16] | 90.3 | TDD+FV [16] | 63.2 |
| $L^2STM$ [10] | 93.6 | $L^2STM$ [10] | 66.2 |
| ST-ResNet [2] | 93.4 | ST-ResNet [2] | 66.4 |
| TSN [15] | 94 | TSN [15] | 68.5 |
| ARNets | 94.1 | ARNets | 70.6 |

**Table 3: Mean accuracy on the Kinetics and Something-V2 datasets and Moments in Time**

| Kinetics | | Something-V2 | | Moments in Time | |
|---|---|---|---|---|---|
| TSN [15] | 73.9 | TSN [15] | 30.0 | TSN [15] | 25.3 |
| I3D [1] | 74.1 | MultiScale TRN [18] | 48.8 | MultiScale TRN [18] | 28.3 |
| R(2+1)D [13] | 75.4 | two-stream TRN [18] | 55.5 | Modalities Ensemble [7] | 30.4 |
| ARNets | 76.2 | ARNets | 64.1 | ARNets | 34.7 |

## 4.6 Compared to state-of-the-art

We also compare ARNets to other state-of-the-art methods. We list the performance on relatively small datasets, UCF-101, and HMDB-51 in Table 2. While the performance on the latest large-scale datasets, Kinetics, Something-v2, and Moments in Time is provided in Table 3. Even simply, our ARNets can beat the state-of-the-art CNNs architecture. In particular, ARNets perform pretty well on Something-v2 and Moments in Time dataset. It means that ARNets can perform well on complex video datasets, more complex more gain.

## 5 CONCLUSION

In this paper, we deeply investigate recurrent networks for video human action recognition. We propose Action Recurrent Networks (ARNets) with several alternative designs. Meanwhile, quantitative and qualitative investigation of these recurrent architectures are provided for human action recognition task. Surprisingly, even using the BNInception backbone with a simple function, our ARNets can achieve the state-of-the-art performance in all benchmark datasets. Extensive experiments and visualization are provided in the paper. We find that compared to standard convolutional networks, ARNets perform better in complex and long videos. This paper will push forward the research in recurrent networks for human action recognition.

## REFERENCES


[1] João Carreira and Andrew Zisserman. 2017. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. CVPR (2017).

[2] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. 2017. Spatiotemporal Multiplier Networks for Video Action Recognition. In CVPR.

[3] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Computation 9, 8 (Nov 1997), 1735–1780.

[4] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.. In ICML

[5] A. Zisserman K. Simonyan. 2014. Two-Stream Convolutional Networks for Action Recognition in Videos. In NIPS.

[6] Zhenyang Li, Kirill Gavrilyuk, Efstratios Gavves, Mihir Jain, and Cees G. M. Snoek. 2018. VideoLSTM convolves, attends and flows for action recognition. Computer Vision and Image Understanding 166 (2018), 41–50.

[7] Mathew Monfort, Alex Andonian, Bolei Zhou, *et al.* 2019. Moments in Time Dataset: one million videos for event understanding. IEEE transactions on pattern analysis and machine intelligence 42 (2), 502–508.

[8] Eunbyung Park, Xufeng Han, Tamara L. Berg, and Alexander C. Berg. 2016. Combining multiple sources of knowledge in deep CNNs for action recognition.. In WACV

[9] Sarabu A, Santra A K. 2021. Human Action Recognition in Videos using Convolution Long Short-term Memory Network with Spatio-temporal Networks. Emerging Science Journal, 5(1), 25-33.

[10] Lin Sun, Kui Jia, Kevin Chen, Dit-Yan Yeung, Bertram E. Shi, and Silvio Savarese. 2017. Lattice Long Short-Term Memory for Human Action Recognition. In ICCV

[11] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E. Shi. 2015. Human Action Recognition Using Factorized Spatio-Temporal Convolutional Networks. In ICCV.

[12] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning Spatiotemporal Features with 3D Convolutional Networks. In ICCV.

[13] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2018. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In CVPR.

[14] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. Journal of Machine Learning Research 9 (2008), 2579–2605. http://www.jmlr.org/papers/v9/vandermaaten08a.html

[15] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In ECCV.

[16] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In NIPS.

[17] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. 2016. Realtime Action Recognition with Enhanced Motion Vector CNNs. In CVPR.

[18] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. 2018. Temporal Relational Reasoning in Videos. In ECCV.

[19] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning Deep Features for Discriminative Localization. In CVPR.

[20] Seyma Yucer and Yusuf Sinan Akgul, "3D Human Action Recognition with Siamese-LSTM Based Deep Metric Learning," Journal of Image and Graphics, Vol. 6, No. 1, pp. 21-26, June 2018. doi: 10.18178/joig.6.1.21-26

[21] Naresh Kumar and Nagarajan Sukavanam, "Motion Trajectory for Human Action Recognition Using Fourier Temporal Features of Skeleton Joints," Journal of Image and Graphics, Vol. 6, No. 2, pp. 174-180, December 2018. doi: 10.18178/joig.6.2.174-180

[22] Muhammad Hassan, Tasweer Ahmad, Nudrat Liaqat, Ali Farooq, Syed Asghar Ali, and Syed Rizwan hassan, "A Review on Human Actions Recognition Using

Vision Based Techniques," Journal of Image and Graphics, Vol. 2, No. 1, pp. 28-32, June 2014. doi: 10.12720/joig.2.1.28-32
[23] Tasweer Ahmad, Junaid Rafique, Hassam Muazzam, and Tahir Rizvi, "Using Discrete Cosine Transform Based Features for Human Action Recognition," Journal of Image and Graphics, Vol. 3, No. 2, pp. 96-101, December 2015. doi: 10.18178/joig.3.2.96-101