Attention-Aware Sampling via Deep Reinforcement Learning for Action Recognition

Wenkai Dong,[1,3] **Zhaoxiang Zhang,**[1,2,3*] **Tieniu Tan**[1,2,3]
[1]Center for Research on Intelligent Perception and Computing (CRIPAC),
National Laboratory of Pattern Recognition (NLPR)
[2]Center for Excellence in Brain Science and Intelligence Technology (CEBSIT),
Institute of Automation, Chinese Academy of Sciences (CASIA)
[3]University of Chinese Academy of Sciences
wenkai.dong@cripac.ia.ac.cn, zhaoxiang.zhang@ia.ac.cn, tnt@nlpr.ia.ac.cn

Abstract

Deep learning based methods have achieved remarkable progress in action recognition. Existing works mainly focus on designing novel deep architectures to achieve video representations learning for action recognition. Most methods treat sampled frames equally and average all the frame-level predictions at the testing stage. However, within a video, discriminative actions may occur sparsely in a few frames and most other frames are irrelevant to the ground truth and may even lead to a wrong prediction. As a result, we think that the strategy of selecting relevant frames would be a further important key to enhance the existing deep learning based action recognition. In this paper, we propose an attention-aware sampling method for action recognition, which aims to discard the irrelevant and misleading frames and preserve the most discriminative frames. We formulate the process of mining key frames from videos as a Markov decision process and train the attention agent through deep reinforcement learning without extra labels. The agent takes features and predictions from the baseline model as input and generates importance scores for all frames. Moreover, our approach is extensible, which can be applied to different existing deep learning based action recognition models. We achieve very competitive action recognition performance on two widely used action recognition datasets.

Introduction

Action recognition has drawn a large amount of attention due to its potential in practical applications like video surveillance and behavior analysis. Since deep learning has achieved great success in the task of object recognition with competitive performance achieved on ImageNet (Krizhevsky, Sutskever, and Hinton 2012), recent works have focused on learning deep representations from a large number of labeled video data. These works can be mainly divided into the following categories: two-stream based methods (Simonyan and Zisserman 2014; Wang et al. 2016; Diba, Sharma, and Van Gool 2017; Feichtenhofer, Pinz, and Wildes 2017; Du et al. 2018), Recurrent Neural Networks based methods (Donahue et al. 2015), 3D ConvNets based methods (Tran et al. 2015) and their combinations (Yue-Hei Ng et al. 2015; Carreira and Zisserman 2017). Among these works, two-stream based methods play the most important role because they can exploit information from video data efficiently. To test a video efficiently, these methods first sample frames sparsely and uniformly from the whole video. Then they perform recognition for each frame and average all the frame-level predictions to obtain the video-level prediction.

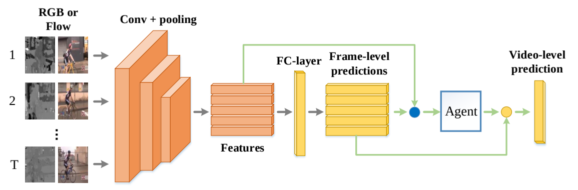

Figure 1: Flow-chart of our proposed method for action recognition. Each stream takes a sequence of frames or optical flow stacks as input. Then, a agent trained by deep reinforcement learning generates importance scores for all frames, which finds the attentions of the video for recognition.

However, discriminative actions may occur sparsely in a few frames and most other frames are irrelevant, which may lead to a wrong video-level prediction. Most existing two-stream based works treat every sampled frame equally and pay little attention to the different importance of frames at the testing stage. We think that the irrelevant frames, which will bring in large proportion of noise, should be discarded by hard attention. Motivated by this, we propose an attention-aware sampling method (AS) to identify discriminative frames and discard the irrelevant frames simultaneously. The agent in our method takes a sequence of features and predictions from the baseline action recognition model as input and outputs a sequence of probabilities for each frame. Since hard attention model is non-differentiable, which cannot be trained end-to-end with the whole network, we apply deep reinforcement learning (DRL) to train the at-

[*]Corresponding Author (zhaoxiang.zhang@ia.ac.cn).

tention agent.

Our method owns the following advantages. The first advantage is that it needs no extra labels. The agent is trained with a supervision reward guided by the classifier of the baseline action recognition model, which means only video-level labels are needed. Secondly, since our agent takes information from a trained baseline model as input, the agent is independent from the training procedure of the baseline model, which means our method is compatible to most existing two-stream based methods. Thirdly, our method is simple and can boost the performance of the baseline model effectively. The main contributions of this paper are summarized as follows:

- We discover a novel problem for action recognition, which is that irrelevant frames should be differentiated and discarded during the testing stage.
- To address this issue, we propose an attention-aware sampling method to select discriminative frames in videos, where the agent is trained by DRL.
- We conduct experiments on two widely used benchmark datasets to demonstrate the effectiveness of our method and achieve competitive results.

Related Work

Deep learning based Action Recognition. Since it achieved great success on ImageNet (Krizhevsky, Sutskever, and Hinton 2012), deep learning has been widely used in action recognition. These works can be mainly divided into the following categories: two-stream-based (Simonyan and Zisserman 2014; Wang et al. 2016), LSTM-based (Donahue et al. 2015), 3D ConvNets-based (Tran et al. 2015), and their combination (Yue-Hei Ng et al. 2015; Carreira and Zisserman 2017). Karparthy et al. (Karpathy et al. 2014) first evaluated ConvNets on a large dataset, called Sports-1M. However, the deep model achieved lower performance than handcraft features (Wang and Schmid 2013) because it failed to capture the complicated temporal information, although they tried different fusion methods including early fusion, late fusion and slow fusion. To overcome this problem, Simonyan et al. (Simonyan and Zisserman 2014) designed the two-stream ConvNets composed of spatial and temporal ConvNets. The spatial and temporal networks take RGB frames and stacked optical flow as input, respectively. The final predictions are obtained by averaging the outputs of two streams. Due to the great success of two-Stream ConNets, many improved methods based on it have been proposed and achieved state-of-the-art performances. Feichtenhofer et al. (Feichtenhofer, Pinz, and Zisserman 2016) improved it by changing the position and method of fusion. Zhang et al. (Zhang et al. 2016) replaced optical flow with motion vector due to the high computational cost of optical flow. Feichtenhofer et al. (Feichtenhofer, Pinz, and Wildes 2016; 2017) applied Residual Networks in two-stream model and injected residual connections between two streams, aiming to achieve spatio-temporal interaction between two streams. Zhu et al. (Zhu et al. 2016) proposed a key volume mining deep framework to identify key volumes and conduct classification simultaneously. Wang et al. (Wang et al. 2016) modeled long-term temporal structure by proposing a temporal segment network using multiple clips sparsely sampled from the whole video. In (Diba, Sharma, and Van Gool 2017), Diba et al. proposed a temporal linear encoding layer to aggregated frame-level information into video-level representation. Du et al. (Du et al. 2018) proposed an interaction-aware spatio-temporal pyramid attention layer inspired by PCA.

Most existing works focus on designing novel deep architectures to model long-term temporal structure and learn discriminative spatio-temporal features. However, they ignore the different importance of frames and treat every frame equally during the testing stage, which can hurt the final performance. Therefore, we propose an attention-aware sampling method to discard the irrelevant frames and preserve the key frames.

Deep reinforcement learning. Reinforcement learning has been originated from the understanding of humans' decision making process, which aims to enable the agent to decide the behavior from its experiences. Unlike conventional supervised machine learning method, reinforcement learning is supervised through the reward signals of actions. Deep reinforcement learning is a combination of deep learning and reinforcement learning, which has been applied in various tasks in recent years. For example, Mnih et al. (Mnih et al. 2013) combined reinforcement learning with CNN and achieved the human-level performance in the Atari game. Rao et al. (Rao, Lu, and Zhou 2017) applied DRL to face recognition and Zhou et al. (Zhou and Qiao 2017) to video summarization. More recently, it has been applied to tracking (Dong et al. 2018), segmentation (Han et al. 2018) and person search (Chang et al. 2018).

However, little progress has been made in reinforcement learning for action recognition. The process of selecting key frames is a kind of hard attention mechanism in our method. Considering that there are no labels for the key frames and our hard attention model is non-differentiable, we formulate the process of mining key frames as a Markov descision process and apply deep reinforcement learning to train the attention model.

Method

The pipeline of our framework is shown in Figure 1. Our attention-aware sampling framework is composed of two parts: a baseline action recognition model and an attention agent. The baseline model takes a sequence of frames as input and generates features and frame-level predictions for each frame, which are the inputs of the agent. The output of the agent is a sequence of probabilities. At inference stage, the probabilities are viewed as importance scores for each frame and are used to pick the most discriminative frames. We first introduce the baseline model as follows.

Baseline model

We build our architecture on one of the best two-steam models: Temporal Segment Networks (Wang et al. 2016). In this work, the two streams are trained separately: the spatial stream takes RGB frames as input to exploit appearance

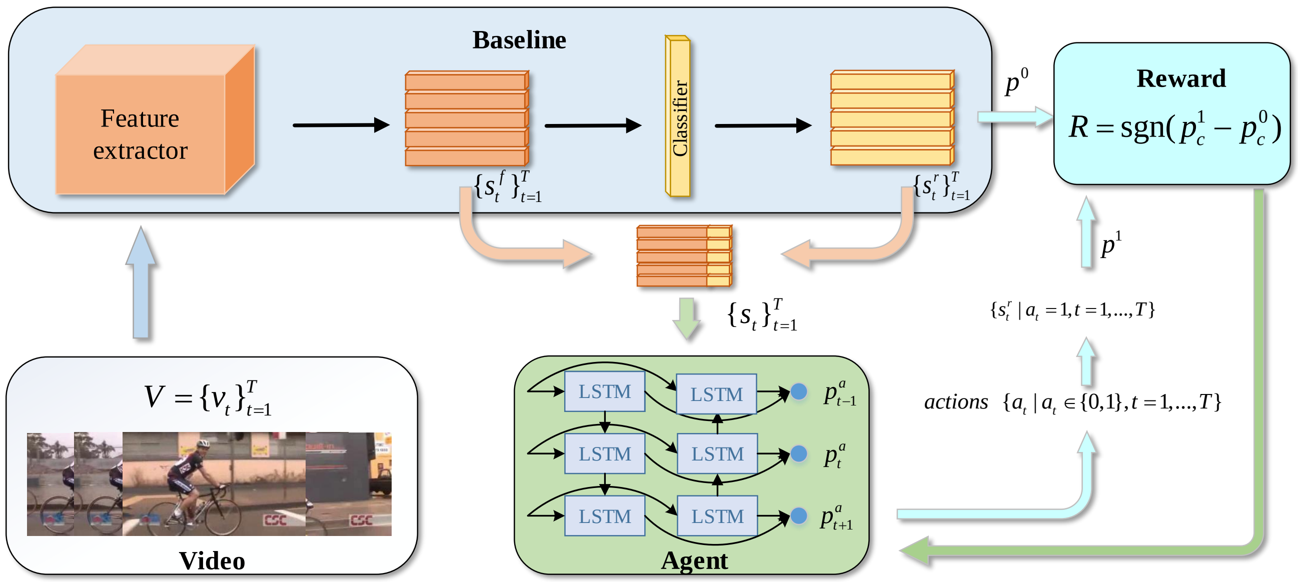

Figure 2: Process of training attention-aware sampling (AS) agent via deep reinforcement learning (spatial stream). The AS agent receives state $\{s_t\}_{t=1}^T$ and takes **action** $\{a_t\}_{t=1}^T$ to select part of the video. The feedback reward is computed based on the difference between the original video-level prediction p^0 and the new prediciton p^1.

information; the temporal stream takes stacks of optical flow images as input to exploit motion information. The final prediction is obtained by averaging the outputs of two streams. For convenience, we take one stream for an example to illustrate our method.

The classical action recognition framework is composed of a feature extractor $F(\cdot)$ and a classifier $CL(\cdot)$. Given a video V, first we sample T frames $\{v_1, v_2, ..., v_T\}$ with equal temporal spacing between them. Then we obtain a sequence of features $\{s_t^f\}_{t=1}^T$ and predictions $\{s_t^r\}_{t=1}^T$ calculated by $F(v_t)$ and $CL(s_t^f)$, respectively. We train the baseline model with the standard categorical cross-entropy loss and the parameters of the baseline model are fixed when training the attention agent.

Attention-aware sampling

Discriminative actions may occur sparsely in a few frames. Hence, not all frames in a video are helpful to recognition. It is necessary to discard irrelevant frames to avoid the adverse effect from them.

Attention models have been widely used in various computer vision tasks. In our approach, we consider the process of seeking the most discriminative frames as the process of finding temporal attentions, which is formulated as a one-step Markov decision process (MDP). Figure 2 provides a sketch map of this process. The agent, interacting with an environment which provides rewards and updates its state, learns by maximizing the total expected reward to select the frames, finally resulting in m most discriminative frames. The state, agent, action and reward of the MDP are elaborated below.

State. The state s_t contains two components: (s_t^f, s_t^r). s_t^f is the feature of frame v_t extracted by $F(v_t)$ and s_t^r is the prediction calculated by the classifier $CL(s_t^f)$. These two components are essential and complementary because s_t^f provides visual content for the agent while s_t^r indicates how accurate the frame is.

Agent. The agent is a bidirectional long-short term memory network (BiLSTM) topped with a fully connected (FC) layer. For a video with length T, the BiLSTM takes as input the state $\{s_t\}_{t=1}^T$ and produces corresponding hidden states $\{h_t\}_{t=1}^T$. Each h_t is the concatenation of the forward hidden state h_t^f and the backward hidden state h_t^b, which encapsulate the future information and the past information with a strong emphasis on the parts surrounding the t^{th} frame. The FC layer that ends with the sigmoid function predicts for each frame an action selection probability $p_t^a = \sigma(Wh_t)$.

Action. The **action** is the selection of each sampled frame. We define 2 types of **action** as 'discarding' and 'remaining'. As shown in Figure 2, the agent emits a vector $p^a \in R^{T\times 1}$ for each video, where $p_t^a \in [0, 1]$ represents the probability of choosing **action** 'remaining' for the t^{th} frame. More specifically, the FC layer of the agent outputs a probability p_t^a for each frame v_t, where a frame-selection action a_t is sampled:

$$a_t \sim \text{Bernoulli}(p_t^a), \tag{1}$$

where $a_t \in \{0, 1\}$ indicates whether the t^{th} frame is selected or not. The selected frames is defined as $\{s_t^r | a_t = 1, t =$

$1, 2, ..., T\}$.

Reward. The reward reflects how good the **action** taken by the agent is with regard to the state s. We generate the reward with the pre-trained classifier $CL(\cdot)$. We first define the r_0 reward as follows:

$$r_0 = sgn(p_c^1 - p_c^0), \tag{2}$$

where c is the ground truth label of the video and p_c represents the probability of predicting the video as class c. p^0 represents the video-level prediction of all frames and p^1 is decided by the hard attention model:

$$p^0 = \frac{\sum_{t=1}^{T} s_t^r}{T}, \ p^1 = \frac{\sum_{t=1}^{T} a_t s_t^r}{\sum_{t=1}^{T} a_t}. \tag{3}$$

The reward r_0 takes value in $\{-1, 1\}$, reflecting the predicted possibility variation of the ground truth if the predicted action does not change after one iteration. Moreover, if it changes, a strong stimulation of $r = \Omega$ is enforced when it turns from incorrect to correct, and a strong punishment of $r = -\Omega$ if the turning goes otherwise. We set Ω as 10 in the paper. Thus, the final form of the reward R can be written as:

$$R = \begin{cases} \Omega & \text{, if stimulation} \\ -\Omega & \text{, if punishment} \\ r_0 & \text{, otherwise} \end{cases} \tag{4}$$

Training with Policy Gradient

The goal of our attention-aware sampling agent is to learn a policy function π_θ with parameters θ by maximizing the expected reward

$$J(\theta) = E_{p_\theta(a_{1:T})}[R], \tag{5}$$

where $p_\theta(a_{1:T})$ denotes the probability distributions over possible action sequences, and R is computed by Eq. 4. As defined above, our **action** sets consists of the different choices of selecting the frames. There are 2 **actions** for each frame and the exponential size 2^T is computationally infeasible for deep Q-learning. Thus, we employ the policy gradient method.

Following the REINFORCE algorithm proposed by Williams (Williams 1992), we can compute the derivative of the expected reward $J(\theta)$ w.r.t. the parameters θ as

$$\bigtriangledown_\theta J(\theta) = E_{p_\theta(a_{1:T})}[R \sum_{t=1}^{T} \bigtriangledown_\theta log \ \pi_\theta(a_t|h_t)], \tag{6}$$

where a_t is the **action** taken by the agent for the frame v_t and h_t is the hidden state from the BiLSTM.

Since Eq.6 involves the expectation over high-dimensional action sequences, which is hard to compute directly, we approximate the gradient by running the agent for N episodes on the same video and then taking the average gradient

$$\bigtriangledown_\theta J(\theta) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} R_n \bigtriangledown_\theta \ log \ \pi_\theta(a_t|h_t), \tag{7}$$

where R_n is the reward computed at the n^{th} episode.

Algorithm 1 Attention-aware sampling agent training

```
Input: Frames $\{v_t\}$ from training set $\{V\}$
Output: Parameters $\theta$ of the hard attention model
 1: Initialize $\theta$ with random values
 2: for $i \leftarrow 1, 2, ..., M$ do
 3:     Sample a sequence of $T$ frames $\{v_t\}_{t=1}^{T}$ from
          training set $\{V\}$
 4:     Compute features $\{s_t^f\}_{t=1}^{T}$ and predictions $\{s_t^r\}_{t=1}^{T}$
 5:     Get a video-level prediction $p^0$
 6:     Compute probabilities $\{p_t^a\}_{t=1}^{T}$
 7:     for $episode \leftarrow 1, 2, ..., N$ do
 8:         Sample $\{a_t\}_{t=1}^{T}$ from policy $\pi_\theta$
 9:         Get the new video-level prediction $p^1$
10:         Compute reward $R_n$
11:     end for
12:     Update $\theta$ with gradient $\bigtriangledown_\theta J(\theta)$ and $\bigtriangledown_\theta L_{sampling}$
13:     Update the moving average of rewards $b$ for the cor-
        responding video
14: end for
15: return $\theta$
```

Although the gradient in Eq.7 gives the direction of updating θ, it may contain high variance which will make the network hard to converge. Therefore, we normalize the reward through subtracting the reward by a constant baseline b, so the gradient becomes

$$\bigtriangledown_\theta J(\theta) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} (R_n - b) \bigtriangledown_\theta \ log \ \pi_\theta(a_t|h_t), \tag{8}$$

where b is simply computed as the moving average of rewards experienced so far for each video.

Besides the $J(\theta)$ term to maximize the expected reward, we minimize the following term during training to help the agent select m frames from the whole T frames when testing,

$$L_{sampling} = \beta \cdot || \frac{1}{T} \sum_{t=1}^{T} p_t^a - \frac{m}{T} ||, \tag{9}$$

where β is the hyperparameter that balances the weighting and the best choice of m will be discussed in the Experiments section.

The details of training the proposed attention-aware sampling agent are summarized in **Algorithm** 1.

Inference

Given a sequence of T frames $\{v_t\}_{t=1}^{T}$ of a test video V, we first employ $F(\cdot)$ and $CL(\cdot)$ to compute their features $\{s_t^f\}_{t=1}^{T}$ and recognition predictions $\{s_t^r\}_{t=1}^{T}$, and then we apply a trained hard attention model π to predict the frame-selection probabilities as importance scores. Finally, we select m frames according to the importance scores, which are the temporal attentions of the testing video. The inference of attention-aware sampling is summarized in **Algorithm** 2.

Discussion about the non-selected frames

In our attention-aware sampling method, the agent selects the key frames to obtain a video-level prediction and dis-

Algorithm 2 Attention-aware sampling agent testing

```
Input: Frames $\{v_t\}$ of testing video $\{V\}$
Output: Video-level prediction $p^1$
1: Initialize $\theta$ from the trained attention model
2: Get features $\{s_t^f\}_{t=1}^T$ and predictions $\{s_t^r\}_{t=1}^T$
3: Compute probabilities $\{p_t^a\}_{t=1}^T$ through policy $\pi_\theta$
4: Sort the frames in descending order by probabilities
5: Select the first $m$ frames
6: Get the video-level prediction $p^1$
7: return $p^1$
```

cards the irrelevant frames. To further demonstrate the effectiveness of our method, we preserve some information of the irrelevant frames:

$$a_t \in \{\alpha, 1\}, \tag{10}$$

where $\alpha \in (0, 1)$ is a hyperparameter. Instead of discarding the irrelevant frames, we set a small weight for them. However, in our experiments, the performance is worse than the pure hard attention model. We think that there are two main reasons leading to this phenomenon. Firstly, the agent in our attention model can discriminate key frames from irrelevant ones. Secondly, the irrelevant frames are not helpful to recognition, which should be discarded at the testing stage.

Experiments

We conducted experiments on two widely used datasets to evaluate our proposed attention-aware sampling method, and compared it with state-of-the-art action recognition approaches. The following describes the details of the experiments and results.

Datasets and Experiment Settings

Datasets. We evaluate our approach on two challenging action recognition datasets: UCF101 (Soomro, Zamir, and Shah 2012) and HMDB51 (Kuehne et al. 2011). The UCF101 dataset consists of 101 action classes with 13,320 video clips and provides videos with a fixed resolution of 320×240. We follow the provided evaluation protocol and report the average accuracy over three splits. The HMDB51 dataset is a large collection of realistic videos from various sources, such as movies and web videos. The dataset is composed of 6,766 video clips from 51 action categories. We also evaluate our approach using three training/testing splits and report average accuracy over these splits.

Implementation Details. We first train the two streams separately. We use the mini-batch stochastic gradient descent algorithm to learn the baseline model parameters, where the batch size is set to 128, weight decay set to 5×10^{-4} and momentum set to 0.9. We initialize network weights with pre-trained models from ImageNet and Kinetics. For the models pre-trained on ImageNet, our temporal stream uses optical flow stack with $L = 5$ frames and is trained with dropout 0.7. The learning rate is initialize as 0.001 and decreases to its $\frac{1}{10}$ after 190 and 300 epochs. The maximum epoch is set as 340. For the extraction of optical flow, we choose the TVL1 (Zach, Pock, and Bischof 2007) optical flow algorithm implemented in OpenCV with CUDA. For spatial network, the learning rate is initialized as 0.001 and decreases to its $\frac{1}{10}$ after 30 and 60 epochs. The whole training procedure stops at 80 epochs. During training, we use the same augmentations as (Wang et al. 2016). The input RGB frames or optical flow frames are of size 256×340, and the width and height of cropped region are randomly sampled from $\{256, 224, 192, 168\}$, which will be resized to 224×224 for training. To train the models pre-trained on Kinetics, the learning rate is initialize as 0.001 and decreases to its $\frac{1}{10}$ after 10 and 20 epochs. The maximum epoch is set as 30 for both streams. We use Pytorch to train our deep neural network, and specifically, 4 GTX 1080Ti GPUs are used for the parallel computing.

For the attention-aware sampling network, we use Adam (Kingma and Ba 2014) as the optimization algorithm. The learning rate is initialize as 10^{-5} and decreases to its $\frac{1}{10}$ after 20 and 40 epochs. The whole training procedure stops at 45 epochs. The weight decay and hyperparameter β are set to 10^{-5} and 0.1, respectively.

Finally, we present our testing scheme for our method. Following the testing scheme of the original two-stream ConvNets (Simonyan and Zisserman 2014), we first sample 25 RGB frames or optical flow stacks from the videos. Meanwhile, we crop 4 corners and 1 center, and their horizontal flippings. Then our attention agent takes each 25-volume as inputs to select m frames or optical flow stacks and averagely aggregates predictions of $10 \times m$ regions to get the video-level prediction for single stream. For the fusion of spatial and temporal stream networks, we take a average of them. The value of m is seleted from $\{4, 8, 12, 16, 20\}$ and $m = 12$ leads to the best performance according to our experiments.

Ablation Study

We first compare our method with several baselines that differ in feature extraction. Then we analyze our method with different experiment settings, including the input state s_t, the weight α and the number of the frames we select from $T = 25$ frames.

Performance evaluation of our approach with different baselines. In this subsection, we verify the effectiveness of our attention-aware sampling method. First we reimplement original two-stream and TSN works whose networks are adapted from BN-inception. Besides, we compare two different pre-trained models, which are pre-trained on ImageNet and Kinetics, respectively. Different from ImageNet, Kinetics is a very large benchmark dataset for action recognition, which consists of approximately $300,000$ videos from 400 action categories. In testing, this baseline averagely aggregates predictions of 25×10 (equal to the number of volumes when doing 10 views testing at 25 temporal locations) sampled volumes to get the video-level prediction. However, discriminative actions may occur sparsely in a video and 25 sampled frames probably contain some irrelevant frames which may lead to a wrong prediction. Therefore, we propose the attention-aware sampling (AS) method to select key frames and discard irrelevant frames. For a fair comparison, we sample the same 250 volumes as the input to our framework and our attention model selects $m \times 10$ volumes according to the state $\{s_t\}_{t=1}^{T}$.

As shown in Table 1, the performance of pre-training on Kinetics is much better than that of pre-training on ImageNet ($69.9\% vs. 75.8\%$), because a large amount of training data on Kinetics can reduce the risk of over-fitting and help both streams learn more robust features. Then it is shown that our AS method can boost the recognition performance of models pre-trained on ImageNet and Kinetics to 71.2% and 77.3%, respectively. We also evaluate the performance of randomly selecting frames and this brings worse recognition accuracy ($75.4\% vs. 75.8\%$) because this will discard key frames and destroy the completeness of the action. The results demonstrate the effectiveness of our AS method. The learned agent is able to discriminate key frames from irrelevant frames and seek different attentions of videos for the recognition.

TSN	Kinetics	Random	AS	Acc.
				66.7
✓				68.5*
✓				69.9
✓	✓			75.8
✓	✓	✓		75.4
			✓	68.6
✓			✓	71.2
✓	✓		✓	**77.3**

Table 1: Two-stream accuracy(%) with different baseline models on HMDB51 dataset (3 splits) when $m = 12$. The line without ✓ denotes the result of the two-Stream ConvNets (Simonyan and Zisserman 2014) with BN-inception backbone. 68.5* is the result reported in (Wang et al. 2016) and AS denotes our attention-aware sampling method.

Performance evaluation of our approach with different state settings. In this subsection, we focus on the study of the state which the agent take as input. s_t^f is the output of the feature extractor $F(v_t)$, which contains the visual content of frame v_t. s_t^r is the recognition output of the classifier $CL(s_t^f)$, which indicates how accurate the recognition output is and how important the frame v_t is to the recognition of the video. Results on comparing the performance of different states are reported in Table 2. It is shown that s_t^r can help the agent learn a better policy ($71.3\% vs. 72.2\%$).

Performance evaluation of our approach with different numbers of frames selected by the agent. The attention-aware sampling method mines the key frames in videos. Experimental results in Table 1 demonstrate that AS method improves the performance. We tried different m in our experiments, which is the number of frames the agent selects given T sampled frames. A small m will increase the risk of discarding key information while a large m will increase the risk of introducing irrelevant noise. Therefore, we need to find a balance between keeping important information and discarding irrelevant noise, which is decided by m. Results in Table 3 show the influence of different

Method	State	HMDB51
Original	—	67.1
TSN	—	70.9
Original + AS	s_t^f	68.0
Original + AS	s_t^f, s_t^r	**69.1**
TSN + AS	s_t^f	71.3
TSN + AS	s_t^f, s_t^r	**72.2**

Table 2: The two-stream accuracy (%) with different state settings on HMDB51 dataset(split1).

m. It is noted that $m = 25$ stands for the baseline and means the agent selects all sampled frames. We observe that $m = 12$ yields the best performance under 2 different selection schemes. Besides, when $m \leq 16$, the performance of AS method is better than that of randomly sampling, which also demonstrates that our agent can learn a good policy. However, AS method gets a worse performance when $m = 20$. We believe that this is because the agent mix up the key information and noise due to the loss $L_{sampling}$.

Then we will introduce and analyze the difference between the **frame-based selection** and **region-based selection** testing schemes in details. Given 250 (1 center, 4 corners and their flippings at 25 temporal locations) volumes sampled from a testing video, we put each 25 volumes into our framework and the agent outputs 25 importance scores. Hence, we can get 250 importance scores from a video. Then there are 2 selection schemes: frame-based selection and region-based selection. The first one is that we average the 10 importance scores of the same frame and obtain 25 (equal to 25 temporal locations) importance scores. Then we select m frames and aggregate the predictions of the m frames ($10m$ volumes) to get a video-level prediction. The second one is that we select m volumes from each 25 volumes of the same region. Then we aggregate the predictions of the $10m$ volumes. The difference between them is that region-based selection may contain more temporal locations than frame-based selection. Moreover, region-based selection performs spatial attention slightly. Therefore, the best performance of region-based selection is better than that of frame-based selection (64.3% vs. 64.1%).

m	Random	Frame	Region
4	61.5	63.3	63.6
8	62.7	63.4	63.7
12	63.1	**64.1**	**64.3**
16	63.1	63.8	63.7
20	63.6	63.3	63.3
25	63.5	63.5	63.5

Table 3: Accuracy (%) of the spatial stream of TSN (pre-trained on Kinetics) with different m and selection schemes on HMDB51 (split1). Frame means the frame-based selection and region means the region-based selection.

Influence of the non-selected frames. To analyze the influence of the frames discarded and further demonstrate the effectiveness of our method, we preserve some information of the irrelevant frames and the results are summarized in Table 4. We can see that the pure hard attention achieves the best performance. We think that there are two main reasons leading to this phenomenon. Firstly, the agent in our attention model can discriminate key frames from irrelevant ones. Secondly, the irrelevant frames are not helpful to recognition, which should be discarded at the testing stage.

α	0	0.1	0.2	0.3
Acc.	**71.24**	71.00	71.15	70.94

Table 4: TSN (pre-trained on ImageNet) accuracy(%) with different α.

Comparison with state-of-the-art results. After exploring the experiment settings and analyzing the effect of attention-aware sampling method, we compare our method with state-of-the-art methods action recognition methods, which are presented in Table 5. We see that among the two-stream based methods without 3D convolutional kernels, such as TLE (Diba, Sharma, and Van Gool 2017) (71.1%), Spatiotemporal Pyramid (Wang et al. 2017) (68.9%) and ISTPAN (Du et al. 2018) (70.7%), our method achieves the best performance on the HMDB51 (71.2%) and outperforms some methods with 3D convolutional kernels: ST-ResNet (Feichtenhofer, Pinz, and Wildes 2016) and STM-ResNet (Feichtenhofer, Pinz, and Wildes 2017). The reason why the performance of I3D (Carreira and Zisserman 2017) (80.2%) is better is that their method uses two 3D ConvNets and takes 64 frames as the input when training, which leads to a high temporal resolution and can capture fine-grained temporal structure of actions. However, 3D ConvNets tend to overfitting without a large amount of training data (Kinetics) and the performance of our method is much better than that of I3D with the models pre-trained on ImageNet (72.2% vs. 66.4% on HMDB51 split1).

We also observe that the improvement of AS method on UCF101 is lower than that on HMDB51 (0.4% vs. 1.3%). We believe that videos in UCF101 are of higher quality than those in HMDB51. Therefore, videos in UCF101 contain less noise and it is harder for the agent to select the key frames.

Conclusion

In this paper, we discover a novel problem for action recognition, which is that irrelevant frames should be differentiated and discarded during the testing stage. To address this issue, we proposed a novel attention-aware sampling method for action recognition to discard misleading frames and select discriminative frames simultaneously. We formulate the process of mining the key frames of videos as a Markov decision process and train the attention model through deep reinforcement learning without extra labels. Moreover, our approach is extensible, which can be applied to different existing deep learning based action recognition models. Our method achieves very competitive performance of action recogntion on widely used UCF101 and HMDB51 datasets.

Method	UCF101	HMDB51
IDT(2013)	85.9	57.2
Two-Stream ConvNet(2014)	88.0	59.4
Two-Stream+LSTM (2015)	88.6	—
TDD (2015)	90.3	63.2
Two-stream Fusion (2016)	92.5	65.4
Key Volume Mining (2016)	93.1	63.3
TSN* (2 modalities) (2016)	94.0	68.5
TLE (2017)	95.6	71.1
Spatiotemporal Pyramid (2017)	94.6	68.9
ISTPAN (2018)	95.5	70.7
C3D (2015)	85.2	—
STM-ResNet (2017)	94.2	68.9
MF-Net + Kinetics (2018)	96.0	74.6
I3D + Kinetics (2017)	**97.9**	**80.2**
TSN	94.2	69.9
TSN + Kinetics	96.5	75.8
TSN + AS	94.6	71.2
TSN + AS + Kinetics	**96.8**	**77.3**

Table 5: Performance comparison with the state-of-the-art. The first part contains methods without using 3D convolutional kernels and methods of the second part benefit a lot from 3D convolutional kernels.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (No. 2018YFB1004600), the National Natural Science Foundation of China (No. 61761146004, No. 61773375), and the Beijing Municipal Natural Science Foundation (No. Z181100008918010).

References

Carreira, J., and Zisserman, A. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 4724–4733. IEEE.

Chang, X.; Huang, P.-Y.; Shen, Y.-D.; Liang, X.; Yang, Y.; and Hauptmann, A. G. 2018. Rcaa: Relational context-aware agents for person search. In *The European Conference on Computer Vision (ECCV)*.

Chen, Y.; Kalantidis, Y.; Li, J.; Yan, S.; and Feng, J. 2018. Multi-fiber networks for video recognition. In *The European Conference on Computer Vision (ECCV)*.

Diba, A.; Sharma, V.; and Van Gool, L. 2017. Deep temporal linear encoding networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1.

Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; and Darrell, T. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634.

Dong, X.; Shen, J.; Wang, W.; Liu, Y.; Shao, L.; and Porikli, F. 2018. Hyperparameter optimization for tracking with continuous deep q-learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Du, Y.; Yuan, C.; Li, B.; Zhao, L.; Li, Y.; and Hu, W. 2018. Interaction-aware spatio-temporal pyramid attention networks for action classification. In *The European Conference on Computer Vision (ECCV)*.

Feichtenhofer, C.; Pinz, A.; and Wildes, R. 2016. Spatiotemporal residual networks for video action recognition. In *Advances in neural information processing systems*, 3468–3476.

Feichtenhofer, C.; Pinz, A.; and Wildes, R. P. 2017. Spatiotemporal multiplier networks for video action recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7445–7454. IEEE.

Feichtenhofer, C.; Pinz, A.; and Zisserman, A. 2016. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1933–1941.

Han, J.; Yang, L.; Zhang, D.; Chang, X.; and Liang, X. 2018. Reinforcement cutting-agent learning for video object segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 1725–1732.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; and Serre, T. 2011. Hmdb: a large video database for human motion recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2556–2563. IEEE.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Rao, Y.; Lu, J.; and Zhou, J. 2017. Attention-aware deep reinforcement learning for video face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3931–3940.

Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, 568–576.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, 4489–4497.

Wang, H., and Schmid, C. 2013. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, 3551–3558.

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, 20–36. Springer.

Wang, Y.; Long, M.; Wang, J.; and Philip, S. Y. 2017. Spatiotemporal pyramid network for video action recognition. In *CVPR*, volume 6, 7.

Wang, L.; Qiao, Y.; and Tang, X. 2015. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4305–4314.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.

Yue-Hei Ng, J.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; and Toderici, G. 2015. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4694–4702.

Zach, C.; Pock, T.; and Bischof, H. 2007. A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*, 214–223. Springer.

Zhang, B.; Wang, L.; Wang, Z.; Qiao, Y.; and Wang, H. 2016. Real-time action recognition with enhanced motion vector cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2718–2726.

Zhou, K., and Qiao, Y. 2017. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. *arXiv preprint arXiv:1801.00054*.

Zhu, W.; Hu, J.; Sun, G.; Cao, X.; and Qiao, Y. 2016. A key volume mining deep framework for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1991–1999.