OCSampler: Compressing Videos to One Clip with Single-step Sampling

Jintao Lin[1] Haodong Duan[2] Kai Chen[3,4] Dahua Lin[2] Limin Wang[1] ✉
[1] State Key Laboratory for Novel Software Technology, Nanjing University, China
[2] The Chinese University of Hong Kong [3] SenseTime Research [4] Shanghai AI Laboratory
jintaolin@smail.nju.edu.cn dh019@ie.cuhk.edu.hk chenkai@sensetime.com
dhlin@ie.cuhk.edu.hk lmwang@nju.edu.cn

Abstract

Videos incorporate rich semantics as well as redundant information. Seeking a compact yet effective video representation, e.g., sample informative frames from the entire video, is critical to efficient video recognition. There have been works that formulate frame sampling as a sequential decision task by selecting frames one by one according to their importance. In this paper, we present a more efficient framework named OCSampler, which explores such a representation with one short clip. OCSampler designs a new paradigm of learning instance-specific video condensation policies to select frames only in a single step. Rather than picking up frames sequentially like previous methods, we simply process a whole sequence at once. Accordingly, these policies are derived from a light-weighted skim network together with a simple yet effective policy network. Moreover, we extend the proposed method with a frame number budget, enabling the framework to produce correct predictions in high confidence with as few frames as possible. Experiments on various benchmarks demonstrate the effectiveness of OCSampler over previous methods in terms of accuracy and efficiency. Specifically, it achieves 76.9% mAP and 21.7 GFLOPs on ActivityNet with an impressive throughput: 123.9 Video/s on a single TITAN Xp GPU.

1. Introduction

With the explosive popularity of social media platforms as well as bountiful online video content, there comes wider attention on effective and scalable approaches that can deal with actions or events recognition in the face of the video data deluge. To this end, most efforts have been devoted to exploring a complicated temporal module to capture relationships across the time dimension by densely applying 2D-CNNs [11, 20, 22, 29, 34, 42] or 3D-CNNs [3,6,7,10,28,31,32]. Though achieving superior performance, the exorbitant computational expense limits the application of these models in real-world scenarios where the deployment is resource-constrained and requires to process high data volumes with stringent latency and throughput requirements.

Figure 1. **Comparisons of other methods and our proposed OCSampler.** Most existing works reduce computational cost by regarding the frame selection problem as a sequential decision task, while OCSampler aims to perform efficient inference by making one-step decision with holistic views. Our method achieves excellent performance on accuracy, theoretical computational expense, and actual inference throughout.

To mitigate this issue, a large body of research has been focusing on designing light-weighted modules [9, 22, 27, 28, 33, 33, 40, 47] to bring efficiency improvements. Being unaware of the complexity of video contents and instance-specific difficulties for video recognition, these models treat

✉: Corresponding author.

all videos equally and adopt naive sampling strategies. To overcome this limitation, extensive studies [8, 12, 14, 39, 41] have been conducted to devise adaptive mechanisms of frame selection on a per-video basis by either determining which frame to observe next, or conditional early exiting in a deterministic order. These approaches all model the frame selection problem as a sequential decision task and prefer to make per-frame decisions individually, leaving out the subsequent parts of the video. Thus, these methods require more inference time even with theoretical computational efficiency and lead to sub-optimal results. Recent methods [19, 25, 26, 30, 35, 38] rely on designing different preset transformations (*e.g.*, process at a specific spatial resolution [25], process at a specific patch [35], *etc.*) and determining which action should be taken on each frame or network module to alleviate computational burden. However, the key to video recognition is aggregating features across different frames. Most of these methods rely on the assumption that several salient frames are equally important to an effective video representation for video recognition, which may introduce temporal redundancy and lack specific consideration for temporal modeling.

A promising alternative to reduce the computational complexity of video analysis, without sacrifice of recognition accuracy, is representing videos with one clip in a single step. Clip-level features [3, 10, 18, 32] commonly used in 3D-CNNs methods reveal the superiority owing to its spatio-temporal information extraction. However, traditional clip-level sampling requires to average the predictions of multiple clips, and clips containing visual redundancy will pollute the final results. Inspired by that, we design an efficient video recognition framework that compresses trimmed/untrimmed videos into a single clip by evaluating a clip-based reward on a per-video basis in one pass. As shown in Figure 1, our basic idea is that modeling the selection problem as a one-step decision task can yield significant savings in both theoretic computation and actual inference time, and sampling an integrated clip is more reasonable than evaluating several frames individually.

Particularly, in this paper, we propose a novel OCSampler to dynamically localize and attend to the instance-specific condensed clip of each video. More specifically, our method first takes a quick skim over the whole video with a light-weight CNN to obtain coarse global information. Then we train a simple yet effective policy network to select the most valuable combination of the clip for the subsequent recognition. This module is learnt with reinforcement learning due to its non-differentiability. Finally, we activate a high-capacity classifier to process the selected clip. Inference on clips constructed with a small number of frames, considerable computation overhead can be saved. Our method allocates computation unevenly across the temporal locations of videos according to their contributions to the recognition task, leading to a significant improvement in efficiency yet still with preserved accuracy.

The vanilla OCSampler framework processes videos using the same number of frames, while the only difference lies in the temporal locations of selected frames. We show that our method can be extended via an adaptive frame budget to reduce the computation spent on "easy" videos, which can be classified precisely with few frames, owing to discriminative backgrounds or objects. This is achieved by introducing an additional budget network that estimates how many frames should be used for a video, which is optimized by pseudo-labels in a self-supervised way.

We evaluate the effectiveness of OCSampler on four efficient video recognition benchmarks, namely ActivityNet [2], Mini-Kinetics [17], FCVID [15], Mini-Sports1M [16]. Experimental results show that OCSampler consistently outperforms all the state-of-the-art by large margins in terms of accuracy and efficiency. Especially, we achieve 76.9% mAP and 21.7 GFLOPs on ActivityNet with an impressive throughput: 123.9 Video/s on a single TITAN Xp GPU. We also demonstrate that the frames sampled by our method can be generalized to boost the efficacy and efficiency of an arbitrary classifier.

2. Related Work

Video recognition. In the context of deep neural networks, there exist two families of models for video recognition, namely 2D-CNN approaches and 3D-CNN approaches. For 2D-CNN approaches, they commonly equip the state-of-the-art 2D-CNN models with the capability of temporal modeling to aggregate features along the temporal dimension, such as temporal pooling [11, 29, 34], recurrent networks [5, 21, 42], efficient temporal modules [20, 22–24], and exploiting explicit temporal information like optical flow [11, 29]. For 3D-CNN approaches [31], most of the works learn spatial and temporal representation by adopting 3D convolution on stacked adjacent frames. Some of them [28, 33] also decompose 3D convolution into a 2D spatial convolution and a 1D temporal convolution or integrate 2D CNN into 3D CNN [45]. However, existing sampling strategies applied to 2D-CNN approaches and 3D-CNN approaches have some shortcomings. Frames uniformly sampled along temporal dimension are sent to 2D-CNN models, which takes fewer frames to represent the whole video but may miss the key information when actions occur in a moment. 3D-CNN models need to aggregate predictions of multiple clips to get a reasonably good result, consuming vast amounts of computation (especially for untrimmed videos). In contrast, our idea is to exploit an effective way to condense a video using a single short clip, which is agnostic to different models.

Sequential sampling. To reduce theoretical computation costs, these approaches consider the frame selection prob-

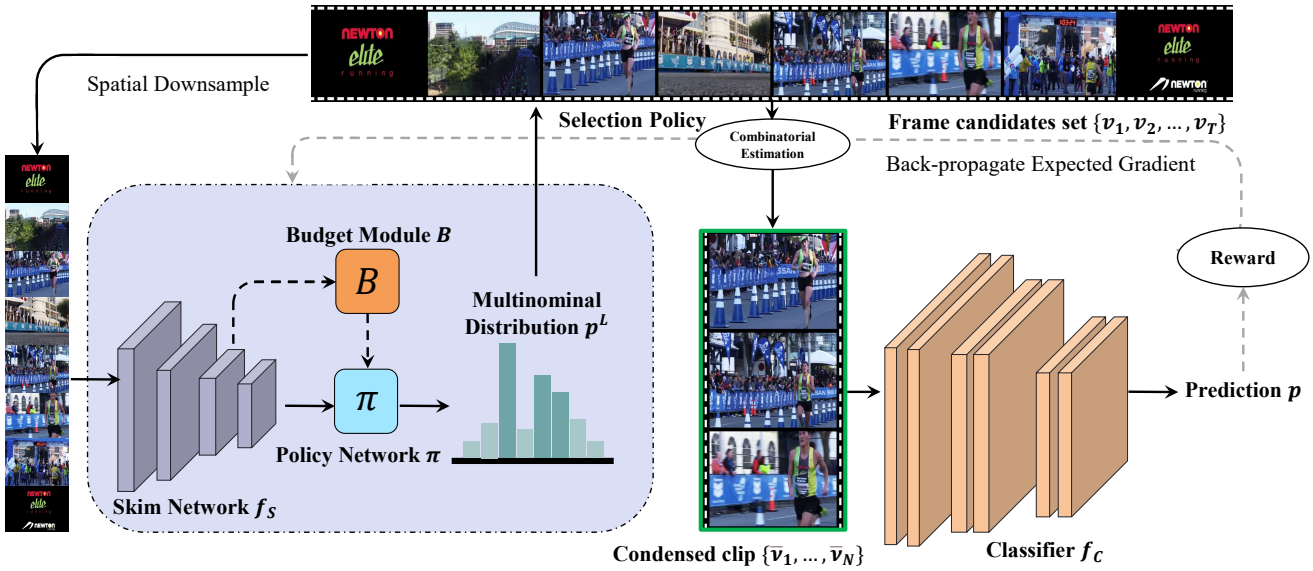

Figure 2. **The overview of our approach.** Given a video, our framework sparsely samples T candidate frames and feeds them into the skim network f_S to take a quick look through the video and extract spatio-temporal features. Then a simple policy network is followed to derive a frame selection policy based on the output multi-nominal distribution of p^L, which activates a subset of N frames to form a single clip as the product of video condensation. By involving an additional budget module B to determine how many frames should be taken on each video, we can further reduce the redundant computation spent on less important frames. Afterwards, an arbitrary classifier is used to obtain the recognition result. Conditioned on the prediction, we back-propagate the expected gradient with the reward of the integrated clip and the corresponding combinational estimation. See texts for more details.

lem as a sequential decision task and require to wait for previous information to indicate which frame to observe next or whether to exit the selection procedure. AdaFrame [39] proposed a Memory-augmented LSTM that provides context information for searching which one to observe next over time. ListenToLook [12] proposed to estimate clip information with a single frame and its accompanying audio using a distillation framework. However, using audio as preview information to seek the next frame cannot avoid irrelevant frames and still takes more than one step to get the final prediction of the entire video. FrameExit [14] formulated the problem in an early-exiting framework with a simple sampling strategy. For each video, FrameExit followed a preset policy to check each frame sequentially and threw out an exiting signal to quit the procedure. Although this simple policy function avoids complex calculations, its deterministic sampling pattern is sub-optimal in terms of exploitation and exploration. In practice, these sequential sampling methods [8, 12, 14, 39, 41] still consume plenty of inference time due to their complex decision process.

Parallel sampling. To mitigate the above issues, some works adopt parallel sampling, which usually chooses what action should be taken on each frame/clip independently and obtains the final selection in parallel. SCSampler [18] used a light-weighted network to estimate a saliency score for each fixed-length clip, while DSN [43] advanced TSN [34] framework by dynamically sampling a discriminative frame within each segment. They both performed the sampling procedure in a non-sequential manner at the cost of limited decision space, leading to sub-optimal selection due to the holistic information vacancy. MARL [37] utilized multi-agents to pick frames in parallel and had to go through a heavy CNN in many iterations to yield STOP actions for all agents. Other works reduced computational costs by selecting input resolution [25], choosing image patches [35], or assigning different bits [30].

In contrast, our method relies on a simple one-step reinforcement learning optimization and does not require multiple steps to determine the final frame selection. Besides, we do not use any RNN-based module but directly aggregate a more holistic feature for video-level modeling. We formulate the problem in a video-to-one-clip condensation framework and show that a reasonable reward function, together with an adaptive frame number budget, can lead to significant performance in both theory and practice.

Video summarization. Video summarization [1, 13, 44, 46] targets selecting a set of video clips or frames to generate a short synopsis that summarizes the video content. DSNet [46] used a temporal interest proposals strategy to solve the temporal consistency problem of video summaries. PGL-SUM [1] tried to overcome drawbacks of RNN-based summarization architectures by using a number of multi-head attention mechanisms. Rather than video summarization, our method focuses on efficient video

recognition, which aims at utilizing as little computation cost as possible to obtain good recognition performance.

3. Method

Unlike most existing works aiming at promoting efficient video recognition by selecting a few frames or clips progressively, our goal is to compress a trimmed/untrimmed video into one single clip with as few frames as possible, while preserving sufficient spatio-temporal cues for video recognition. To this end, we introduce OCSampler, an efficient and effective framework to condense a video into an integrated clip. With OCSampler, the computation overhead can be significantly reduced without sacrificing accuracy. We first describe the components of OCSampler. Then we introduce the training algorithm for each component. Finally, we extend our framework by considering an adaptive frame number budget, which allocates different amounts of computation for each video.

3.1. Network Architecture

Overview. Figure 2 illustrates an overview of our approach. Given an input video, we first uniformly sample T frames along the temporal dimension as frame candidates. OCSampler first skims the frame candidates at a lower resolution using a light-weighted skim network f_{S}, to obtain coarse frame-level features. Then, the features are fed into the policy network π to encode spatio-temporal information across frames and determine the optimal frame set to form an integrated clip, which maximizes a reward function parameterized by the output from the classifier f_{C}. The classifier f_{C} takes the single clip as inputs and predicts the action category. It is worth noting that OCSampler obtains an integrated clip only in one step. In the following sections, we describe these components in details.

Skim network f_{S} is a light-weighted network to extract deep features for frame candidates. It is designed to provide global views across different time for determining which frames should be selected to form a clip for classifier f_{C}. Components like TSM [22] can be inserted to equip Skim network with the capability of fusing information among frame candidates. Note that the additional computation cost incurred by f_{S} is negligible compared with the classifier f_{C}.

Formally, given a candidate set $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_T\}$ uniformly sampled along the temporal dimension with spatial size $H \times W$, they are first resized to lower resolution $\tilde{H} \times \tilde{W}$ and then sent to f_{S} to generate a global video descriptor $\boldsymbol{z}_t^{\mathrm{S}}$:

$$\boldsymbol{z}^{\mathrm{S}} = \{\boldsymbol{z}_1^{\mathrm{S}}, \boldsymbol{z}_2^{\mathrm{S}}, \ldots, \boldsymbol{z}_T^{\mathrm{S}}\} = f_{\mathrm{S}}(\{\tilde{\boldsymbol{v}}_1, \tilde{\boldsymbol{v}}_2, \ldots, \tilde{\boldsymbol{v}}_T\}), \quad (1)$$

where t is the frame index and $\boldsymbol{z}_t^{\mathrm{S}}$ encodes context information for each frame on a per-video basis.

Policy network π receives the global context feature $\boldsymbol{z}^{\mathrm{S}}$ from Skim network f_{S}, and localizes which frames can be used to form a salient clip for each video. Note that this procedure is performed only in one iteration and uses no complicated CNN-based or RNN-based modules but one linear projection f_{L} followed by Softmax function ϕ with an effective clip-relevant policy function:

$$\boldsymbol{p}^{\mathrm{L}} = \{p_1^{\mathrm{L}}, p_2^{\mathrm{L}}, \ldots, p_T^{\mathrm{L}}\} = \phi(f_{\mathrm{L}}(\{\boldsymbol{z}_1^{\mathrm{S}}, \boldsymbol{z}_2^{\mathrm{S}}, \ldots, \boldsymbol{z}_T^{\mathrm{S}}\})), \quad (2)$$

where $\boldsymbol{p}_t^{\mathrm{L}}$ refers to the softmax probability for each frame. Formally, as shown in Figure 3, π determines the chosen N frames from candidates $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_T\}$ to be sent to classifier f_{C}. Since the target is to determine a representative clip rather than several salient frames, it involves making set-level decisions that are non-differentiable and harder than making binary ones due to larger search space. Given that, we still formalize π as a one-step Markov Decision Process (MDP) and train it with reinforcement learning. Specifically, the selection of the clip $\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}$ is drawn from the distribution $\pi(\cdot|\boldsymbol{z}^{\mathrm{S}}, \theta_L)$.

where θ_L denotes learnable parameters of the linear projection f_{L}. In our implementation, we establish a multinomial distribution on them, parameterized by the output probability of π. During training, $\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}$ are produced by sampling from the policy based on corresponding multinomial distribution. During testing, candidates with maximum probabilities are adopted in a deterministic inference procedure.

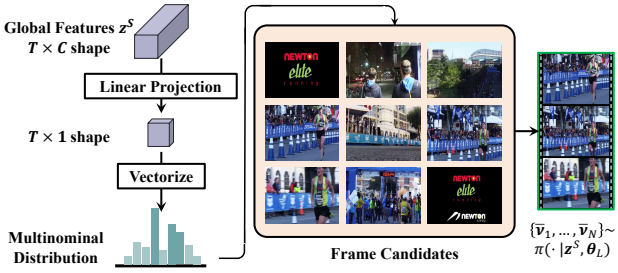

Figure 3. **The architecture of the policy network.** The global context feature z^S is fed into a linear projection layer followed by a vectorization operation, the output of which establish a multinomial distribution $\pi(\cdot|\boldsymbol{z}^{\mathrm{S}}, \theta_L)$ on frame candidates (here we take 9 as an example). During training, we sample frames $\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}$ from $\pi(\cdot|\boldsymbol{z}^{\mathrm{S}}, \theta_L)$, while at the test time, we directly select frames with the largest N softmax probability.

Classifier f_{C} can be any classification network used in video recognition. It receives a clip of temporal length N from policy network π and outputs the recognition result of the video. To be specific, Classifier f_{C} directly processes a clip of N frames $\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}$ with original resolution $H \times W$, *i.e.*,

$$\boldsymbol{p} = f_{\mathrm{C}}(\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}), \quad (3)$$

where $\boldsymbol{p}$ indicates the probability scores for each class. Notably, Classifier f_{C} accounts for most of the computational overhead in our framework and yields the prediction at a

time, instead of sequentially processing each frame. Such a design reduces both computational complexity in theory and inference time in practice.

3.2. Training Algorithm

There are two stages in our training algorithm to optimize OCSampler framework.

Stage I: Initialization. In this stage, we warm up f_{S} and f_{C} by video recognition tasks on target datasets. We train f_{S} by randomly sampling T frames with size $\tilde{H} \times \tilde{W}$ to minimize the cross-entropy loss $L_{\mathrm{CE}}(\cdot)$ over the training set $\mathcal{D}_{\text{train}}$:

$$\underset{f_{\mathrm{S}}}{\text{minimize}} \quad \mathbb{E}_{\{\tilde{\boldsymbol{v}}_1, \tilde{\boldsymbol{v}}_2, \ldots, \tilde{\boldsymbol{v}}_T\} \in \mathcal{D}_{\text{train}}} \left[L_{\mathrm{CE}}(\tilde{\boldsymbol{p}}, y) \right]. \tag{4}$$

Similarly, we pretrain f_{C} by using randomly sampled N frames with $H \times W$ resolution:

$$\underset{f_{\mathrm{C}}}{\text{minimize}} \quad \mathbb{E}_{\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N\} \in \mathcal{D}_{\text{train}}} \left[L_{\mathrm{CE}}(\boldsymbol{p}, y) \right]. \tag{5}$$

Here, y refers to the corresponding label of the sample. Given the good recognition performance, f_{S} and f_{C} are equipped with the ability to extract spatio-temporal features from an arbitrary sample on target datasets and provide good quality reward signals with less noise, leaving the basis for policy network π.

Stage II: Optimizing policy network. In this stage, we freeze the parameters of classifier f_{C} learned in stage I and train policy network π with reinforcement learning by solving one-step Markov Decision Process problem. Based on the probability $\boldsymbol{p}^{\mathrm{L}}$ predicted by f_{L} with global context feature $\boldsymbol{z}^{\mathrm{S}}$ (see Eq. 2), π receives a reward r indicating how beneficial this combination is to construct a clip for recognition. We optimize π by maximizing the sum of the rewards:

$$\underset{\pi}{\text{maximize}} \quad \mathbb{E}_{\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\} \sim \pi(\cdot | \boldsymbol{z}^{\mathrm{S}}, \theta_L)} \left[r \right]. \tag{6}$$

In our implementation, we adopt the off-the-shelf policy gradient algorithm [36] to solve Eq. 6. Note that there are $\binom{T}{N}$ different cases to choose N frames from T candidates, which makes it hard to precisely calculate the combinatorial probability and intractable to handle directly. Formally, we define $q(i_1, \ldots, i_N | \boldsymbol{p}^{\mathrm{L}})$ as the probability of sampling frames sequentially with the order $(i_1, \ldots, i_N)$:

$$q(i_1, \ldots, i_N | \boldsymbol{p}^{\mathrm{L}}) = p^L_{i_1} \times \frac{p^L_{i_2}}{1 - p^L_{i_1}} \times \ldots \times \frac{p^L_{i_N}}{1 - \sum_{j=1}^{N-1} p^L_{i_j}}, \tag{7}$$

There are $N!$ different permutations for N elements, we denote the set of all $N!$ as $\mathcal{P}$. Then the probability of sampling these N frames can be precisely calculated by summing q for all $N!$ different permutations:

$$Prob_{\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}} = \sum_{\sigma \in \mathcal{P}} q(\sigma(i_1), \sigma(i_2), ..., \sigma(i_N) | \boldsymbol{p}^{\mathrm{L}}). \tag{8}$$

However, Eq. 8 is only tractable for a small N (*e.g.*, $N < 10$). In experiments, we estimate this term with the probability of a subset of all permutations (*e.g.*, subset with $\binom{T}{8}$ items) and find that the policy network can be optimized well either with the precise or the estimated probability.

In our case, where policy network aims at figuring out how to condense a video with one clip rather than pick up several frames separately, the reward r is expected to evaluate the integrated clip $\overline{V}$, *i.e.*, $\{\overline{\boldsymbol{v}}_1, \overline{\boldsymbol{v}}_2, \ldots, \overline{\boldsymbol{v}}_N\}$, in terms of video recognition. To this end, we define r as:

$$\begin{aligned} & r(\{\overline{\boldsymbol{v}}_1, \ldots, \overline{\boldsymbol{v}}_N\}) \\ = & \boldsymbol{p}_y(\{\overline{\boldsymbol{v}}_1, \ldots, \overline{\boldsymbol{v}}_N\}) \\ & - \mathbb{E}_{\overline{V} \sim \text{UniformSample}(\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T\})} \left[\boldsymbol{p}_y(\overline{V}) \right], \end{aligned} \tag{9}$$

where $\boldsymbol{p}_y$ refers to the softmax prediction on y (*i.e.*, confidence on the ground-truth label, see Eq. 3). When computing r, we take all of the N frames $\{\overline{\boldsymbol{v}}_1, \ldots, \overline{\boldsymbol{v}}_N\}$ into consideration to avoid information redundancy and short-sighted mistakes raised by single frame judgement. The second term in Eq. 9 refers to the expected value obtained by uniformly sampling N frames from candidates. Since reinforcement learning may be of high variance and converge slowly, we introduce another policy, which does not depend on the policy network, to affect the variance and stabilize the training process significantly.

3.3. Adaptive Frame Number Budget

Processing videos of different complexity equivalently with the same amount of computation is still sub-optimal. To overcome this, we extend our OCSampler to OCSampler+, which automatically learns to select fewer frames for easier videos and more frames for harder ones.

Budget module. We add an additional Budget module f_{B} that takes global context feature $\boldsymbol{z}^{\mathrm{S}}$ as input between Skim network f_{S} and policy network π. Each of these features is first passed to one layer of MLP with 64 neurons independently (shared weights among all streams). The resulting features are then averaged and linearly projected, followed by a softmax function to estimate the frame budgets.

Training with Self-Supervision. We construct a budget label y^B indicating the probability of how many frames should be used by analyzing the statistics obtained from considering all of the combinations. Formally, given a video, we define $\mathcal{G}^m = \{g^m_1, g^m_2, \ldots, g^m_c\}$ (where $1 \leq m \leq T$ and $c = \binom{T}{m}$) as the list containing combinations of m frames from the frame candidate set $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_T\}$. We send each item $g^m_i \in \mathcal{G}^m$ to classifier f_{C} to obtain a boolean value $a^m_i \in \{0, 1\}$, which specifies whether this combination can be predicted correctly. After that, we obtain the ratio of prediction correction r^m with the estimation:

$$r^m = \sum_i a^m_i / \binom{T}{m}. \tag{10}$$

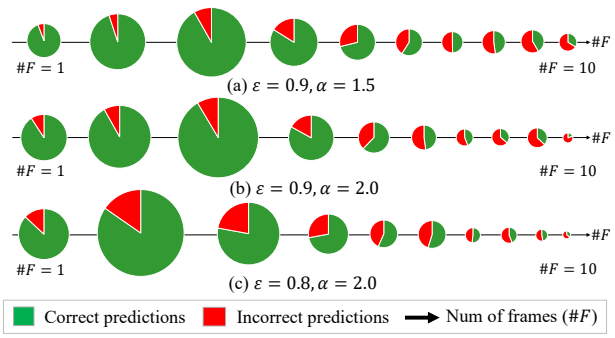

Figure 4. **Trade-off between frame number budgets and prediction accuracy.** The statistics of our method equipped with a budget module for different ϵ and α on the validation set of ActivityNet. The circle area at a certain number of $\#F$ represents the percentage of samples using $\#F$ frames for prediction. Easier examples use fewer frames with higher accuracy, while harder examples use more frames leading to increased miss-classifications.

Based on r^m, we use ϵ to determine the minimum budget required to predict a video correctly with classifier f_{C}:

$$y_k^B = 1, \quad \text{where } k = \arg\min_i (\epsilon \leq r^i). \tag{11}$$

Provided that single-label is more likely to lead to bias on accuracy, we leverage other options with a smooth function to balance the accuracy and efficiency:

$$y_i^B = \begin{cases} 0 & \text{if } i < k, \\ \frac{1}{\alpha^{(i-k)}} & \text{if } i > k, \end{cases} \tag{12}$$

where $\alpha > 1$ and is the hyper-parameter that controls the trade-off between accuracy and computational cost. An example is shown in Figure 4. Then, we learn parameters of the budget network by minimizing the cross-entropy loss between the predicted probability and the pseudo label y^B:

$$L_{\text{Budget}} = L_{\text{CE}}(\boldsymbol{z}^{\text{S}}, y^B). \tag{13}$$

Notably, this procedure of estimating frame budgets also applies for one step. Similar to Eq. 8, we use Monte-Carlo sampling to estimate r^m for Eq. 10. Moreover, to overcome the long-tail issue owing to sample imbalance, we assign class weight based on the sample distribution for Eq. 13. During training, we first optimize the Budget module f_{B} with skim network f_{S} to get the frame budget estimation, and then learn the policy network π as mentioned in Stage II. During inference, we choose the maximum probability in f_{B} as the number of used frames.

4. Experiment

In this section, we conduct comprehensive experiments on widely used datasets to verify our method. We first briefly describe our experimental setup. Then, we compare OCSampler with state-of-the-art approaches , showing

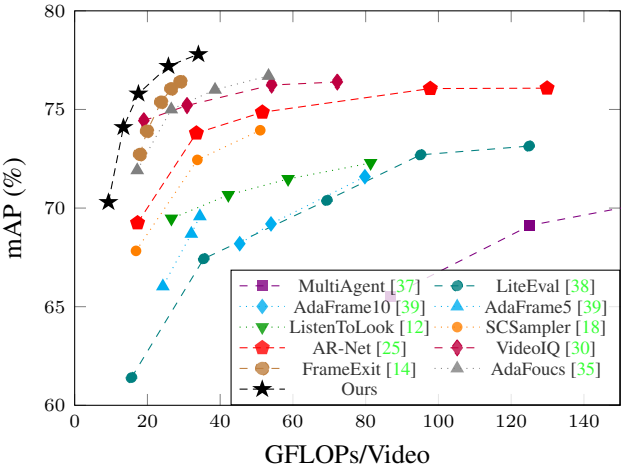

Figure 5. **Accuracy vs. efficiency curves on ActivityNet.** Our proposed OCSampler obtains the best recognition accuracy with fewer GFLOPs than state-of-the-art methods. We directly quote the numbers reported in published papers.

that OCSampler boosts the performance of existing methods. Finally, we provide ablation results to provide additional insights into our policy learning.

4.1. Experimental Setup

Datasets. We report the performance of our approach on four datasets: (1) ActivityNet-v1.3 [2] consists of 200 classes and contains 10,024 training videos and 4,926 validation videos with an average duration of 117 seconds; (2) FCVID [15] is labeled with 239 action categories and includes 45,611 training videos and 45,612 validation videos with an average duration of 167 seconds; (3) Mini-Kinetics has 200 classes from Kinetics [17] assembled by [25, 26], including 121,215 training videos and 9,867 validation videos with an average duration of 10 seconds; (4) Mini-Sports1M is a subset of full Sports1M [16] introduced by [12], containing 30 training videos per class and 10 validation videos per class with a total of 487 action classes.

Evaluation metrics. To evaluate the accuracy, We use top-1 accuracy for multi-class (Mini-Kinetics) classification and mean average precision (mAP) for multi-label classification (ActivityNet, FCVID, and Mini-Sports1M), respectively. To measure the computational cost, we use giga floating-point operation (GFLOPs) as efficiency reflection, which is a hardware-independent metric. We report per video GFLOPs for all experiments since some methods use different numbers of frames per video for recognition.

Implementation details. Experiments are conducted on MMAction2 [4]. If not specified, we uniformly sample 10 frames from each video as frame candidates on all the datasets. Following [14, 25], during training, we adopt random scaling to all frames followed by 224×224 random cropping and random flipping. For inputs to light-weighted CNN, we further lower the resolution of video frames to

Table 1. **Comparison to state of the art on ActivityNet-v1.3 and Mini-Kinetics.** OCSampler outperforms exiting methods in terms of accuracy and efficiency using ResNet, SlowOnly, and X3D-S backbones with ImageNet/Kinetics pretraining. The column of Backbones is for classifier, and best results are **bold-faced**.

Methods	Backbones	ActivityNet		Mini-Kinetics	
		mAP	GFLOPs	Top-1	GFLOPs
ImageNet					
LiteEval [38]	ResNet	72.7%	95.1	61.0%	99.0
SCSampler [18]	ResNet	72.9%	42.0	70.8%	41.9
AR-Net [25]	ResNet	73.8%	33.5	71.7%	32.0
videoIQ [30]	ResNet	74.8%	28.1	72.3%	20.4
AdaFocus [35]	ResNet	75.0%	26.6	72.9%	38.6
FrameExit [14]	ResNet	76.1%	26.1	72.8%	19.7
OCSampler	ResNet	**77.2%**	25.8	**73.7%**	21.6
OCSampler	ResNet	76.9%	**21.7**	72.9%	**17.5**
OCSampler+	ResNet	75.4%	17.9	72.2%	15.8
Kinetics					
Ada2D [19]	SlowOnly-50	84.0%	701	79.2%	738
ListenToLook [12]	R(2+1)D-152	89.9%	2640	–	–
MARL [37]	SEResNeXt-152	90.0%	7540	–	–
OCSampler	SlowOnly-50	87.3%	**68.2**	**82.6%**	**27.3**
OCSampler	SlowOnly-101	**90.1%**	593	-	-
Kinetics					
FrameExit [14]	X3D-S	86.0%	9.8	–	–
OCSampler	X3D-S	**86.6%**	**7.9**	–	–

128×128. During inference, we still feed light-weighted CNN with 128×128 resolution frames and average prediction of 224×224 center-cropped patches for all sampled frames. If not mentioned, we adopt MobileNetV2-TSM and ResNet50 as skim network f_S and classifier f_C respectively. A one-layer fully-connected network with a hidden size of 1280 is used in policy network π. T is set to 10 by default.

4.2. Main Results and Analysis

Comparison with the state-of-the-art methods. The result for ActivityNet and Mini-Kinetics are shown in Table 1. For ImageNet-pretrained cases, we use the ResNet-50 model provided by [14] as the classifier backbone and use $T = 10$ to keep the same with [14]. OCSampler outperforms all other approaches by obtaining an enhanced accuracy with up to $5\times$ GFLOPs reduction for both ActivityNet and Mini-Kinetics. Particularly, we outperform all previous methods with more than 4.4 GFLOPs on ActivityNet, and achieve the same Top-1 accuracy with AdaFocus [35] using less GFLOPs than half of its on Mini-Kinetics. For Kinetics-pretrained cases, we use SlowOnly models as classifier backbones, and it can be observed that our method outperforms alternative baselines by large margins in terms of efficiency. In particular, on ActivityNet, we outperform MARL [37], the leading method among competitors, with $11.7\times$ less computational overhead. And for Mini-Kinetics, we also surpass Ada2D [19] with 3.4% higher accuracy and $26.0\times$ less GFLOPs. The gain in accuracy is mainly attributed to the larger search space without limitation in our framework, while the gain in efficiency is attributed to the reasonable reward function for video condensation (see Section 4.3 for detailed analysis). To verify that the performance of our framework is not limited to the type of classifiers, we conduct experiments with the X3D-S backbone following [14]. With the same light-weight X3D-S as our backbone, OCSampler achieves higher accuracy with 1.9% less GFLOPs, saving 13 frames for inference. This demonstrates the superiority of our framework for efficient video recognition with any classifiers.

Table 2. **Practical efficiency performance of OCSampler and other currently proposed methods on ActivityNet.** The throughtput are evaluated on a NVIDIA TITAN Xp GPU. Here we use MN, MN-T, RN and SLOW to denote MobileNetV2, MobileNetV2-TSM, ResNet and SlowOnly respectively. The best results are **bold-faced**.

Methods	Backbones	mAP	GFLOPs	Throughput (Videos/s)
ImageNet				
AdaFrame [39]	MN+R50	71.5%	79.0	6.4
FrameExit [14]	ResNet-50	76.1%	26.1	19.1
AR-Net [25]	MN+RN	73.8%	33.4	23.1
AdaFocus [35]	MN+RN	75.0%	26.6	44.9
OCSampler	MN-T+R50	**76.9%**	**21.7**	**123.9** (↑2.8x)
Kinetics				
MARL [37]	SEResNeXt-152	90.0%	7715	0.5
ListenToLook [12]	(R2+1)D-152	89.9%	2640	0.8
OCSampler	MN-T+SLOW101	**90.1%**	**593**	**4.4** (↑5.5x)

Results of varying number of used frames are presented in Figure 5. We change the number of used frames within $N \in \{2, 3, 4, 6, 8\}$, and plot the corresponding mAP v.s. GFLOPs trade-off curves on ActivityNet. We also present current state-of-the-art with various computational costs. One can observe that OCSampler leads to a considerably better trade-off between efficiency and accuracy.

Adaptive frame number budget. We investigate the effectiveness of extended OCSampler with frame number budgets by altering the amount of computational overhead per video. Figure 4 illustrates accuracy and the number of processed frames with different values of α and ϵ. According to Eq. 11 and Eq. 12, a higher α encourages more videos to use fewer frames for recognition (the first row) compared to a lower α (the second row), while a higher ϵ serves as a more strict threshold to depress using fewer frames for recognition (the second row) compared to a lower ϵ (the third row). It can also be seen that the fewer number of frames are used, the more correct the result becomes. This trend is desirable since easier samples require less computational cost while harder ones take more overhead.

Practical efficiency. To gain a better understanding of the efficiency achieved by OCSampler, we also test the real inference speed of different methods on a single NVIDIA TITAN Xp GPU. Table 2 shows that our practical acceleration is significant compared to other approaches, which is attributed to the one-step decision procedure for all frames without multiple iterations in our framework.

Results on FCVID and Mini-Sports1M. As shown in Table 3, our approach shows excellent efficacy and efficiency. Without additional modalities, OCSampler outper-

Table 3. **Comparison with state of the art methods on Mini-Sports1M and FCVID**. OCSampler achieves the best mAP while offering significant savings in GFLOPs.

Methods	Mini-Sports1M		FCVID	
	mAP	GFLOPs	mAP	GFLOPs
LiteEval [38]	44.7%	66.2	80.0%	94.3
SCSampler [18]	44.3%	42.0	81.0%	42.0
AR-Net [25]	45.0%	37.6	81.3%	35.1
AdaFuse [26]	44.1%	60.3	81.6%	45.0
OCSampler	**46.7%**	**25.7**	**82.7%**	**26.8**

Table 4. **Comparisons of frame selection policies.** We report the results on different number of N. All of the policies use the same classifier and frame candidates, where T is set to 10.

Policy		mAP			
		$N=1$	$N=2$	$N=4$	$N=6$
Deterministic Policy	Random	50.1%	62.2%	71.2%	73.8%
	Uniform	54.2%	65.5%	72.6%	73.8%
	FrameExit	54.2%	62.2%	70.4%	74.0%
Learned Policy	Frame Reward	61.5%	68.8%	74.2%	76.2%
	Vanilla Reward	60.5%	69.7%	75.2%	76.6%
	Ours	**61.5%**	**70.6%**	**75.8%**	**77.2%**

forms SCSampler by a margin of 2.4% in mAP while using 38.8% less computation on Mini-Sports1M and achieves 1.4% improvement in mAP alleviating 23.6% computational overhead over AR-Net.

4.3. Ablation Studies

Effectiveness of the learned selection policy. Table 4 summarizes the effect of different selection policies. For deterministic policy, we investigate three alternatives: (1) *randomly* sampling frames, (2) *uniformly* sampling frames, and (3) A deterministic policy proposed by *FrameExit*, which can be seen as decoding videos from sparsely to densely. Besides, we also consider using different reward functions for reinforcement learning: (1) *frame reward* considers the confidence of each frame rather than the integrated clip as rewards, (2) *vanilla reward* removes the second item in Eq. 9 as rewards. One can observe that the learned policies have better performance and the best results are obtained by our designed reward function. Notably, uniform policy appears stronger than FrameExit policy when N is set to 2 or 4. This is a reasonable observation, as in these cases, FrameExit policy collects more frames from the first half of videos but omits the second half while uniform policy leverages temporal information with evenly sampled frames.

Effectiveness of decision space. We investigate the effectiveness of decision space by using different numbers of frame candidates. As shown in Table 5, only adopting $T = 16$ frame candidates leads to an mAP increase of 4.0% with only 1.7 GFLOPs additional computation overhead. An interesting phenomenon is that expanding frame candidates leads to a significant rise in accuracy performance at the beginning, but the growth gradually becomes stabilized as the candidate set becomes large, which may be attributed to the saturation of video information. In this sense, the candidate set includes salient frames to represent certain content of the video. As the expansion of candidate set, more salient frames are involved in condensing the entire video, while duplicate information might also pollute the recognition performance owing to introduced temporal redundancy.

Table 5. **Effectiveness of Decision space**. The number of frame candidates N is set to 6 for all settings. For $T = 6$, we directly send frames to classifier without sampling.

No. frame candidates	6	8	10	16	24
mAP	74.0%	76.2%	77.2%	78.0%	78.3%
GFLOPs	24.7	25.6	25.8	26.4	27.2

Table 6. **Generality of selected frames from OCSampler**. Here we set N to 4 for all classifiers. RN, MN-T and SLOW denote ResNet, MobileNetV2-TSM and SlowOnly respectively.

Ablation	mAP(%)				
	RN	X3D-S	R(2+1)D	MN-T	SLOW
Baseline	67.5	62.1	61.1	57.2	77.1
OCSampler	75.8 (↑8.3)	68.3 (↑6.2)	67.2 (↑6.1)	62.0 (↑4.8)	81.9 (↑4.8)

Generality of selected frames. These selected frames are of good generality to improve other classifiers' performance without an extra training scheduler. As shown in Table 6, we directly apply the frames selected by OCSampler with ResNet-50 to other backbones, which also leads to significant improvements in recognition performance.

5. Conclusion

In this paper, we have presented a both accurate and efficient sampling framework by condensing a video into a clip within one step, which we refer to as OCSampler. Our OCSampler avoids heavy computational overhead and addresses the problem of multiple inference times existing in most sampling methods. Moreover, our work designs a simple but reasonable reward function to consider all frames in one clip collectively rather than individually, and strikes an excellent performance on accuracy without sacrificing efficiency. We further extend our method to select adaptive numbers of frames by adopting a frame number budget module. Experiments on four widely used benchmarks verify the effectiveness of our method over existing works in terms of recognition accuracy, selection transferring, computational cost, and practical speed.

Acknowledgements. This work is supported by National Natural Science Foundation of China (No.62076119, No.61921006), Program for Innovative Talents and Entrepreneur in Jiangsu Province, and Collaborative Innovation Center of Novel Software Technology and Industrialization. It is also supported by the Shanghai Committee of Science and Technology, China (Grant No. 20DZ1100800).

References

[1] Evlampios Apostolidis, Georgios Balaouras, Vasileios Mezaris, and Ioannis Patras. Combining global and local attention with positional encoding for video summarization. In *2021 IEEE International Symposium on Multimedia (ISM)*, pages 226–234. IEEE, 2021. 3

[2] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015. 2, 6

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 1, 2

[4] MMAction2 Contributors. Openmmlab's next generation video understanding toolbox and benchmark. https://github.com/open-mmlab/mmaction2, 2020. 6

[5] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 2

[6] Haodong Duan, Yue Zhao, Kai Chen, Dian Shao, Dahua Lin, and Bo Dai. Revisiting skeleton-based action recognition. *arXiv preprint arXiv:2104.13586*, 2021. 1

[7] Haodong Duan, Yue Zhao, Yuanjun Xiong, Wentao Liu, and Dahua Lin. Omni-sourced webly-supervised learning for video recognition. In *European Conference on Computer Vision*, pages 670–688. Springer, 2020. 1

[8] Hehe Fan, Zhongwen Xu, Linchao Zhu, Chenggang Yan, Jianjun Ge, and Yi Yang. Watching a small portion could be as good as watching all: Towards efficient video classification. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 705–711, 2018. 2, 3

[9] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 203–213, 2020. 1

[10] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211, 2019. 1, 2

[11] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016. 1, 2

[12] Ruohan Gao, Tae-Hyun Oh, Kristen Grauman, and Lorenzo Torresani. Listen to look: Action recognition by previewing audio. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10457–10467, 2020. 2, 3, 6, 7

[13] Junaid Ahmed Ghauri, Sherzod Hakimov, and Ralph Ewerth. Supervised video summarization via multiple feature sets with parallel attention. 2021. 3

[14] Amir Ghodrati, Babak Ehteshami Bejnordi, and Amirhossein Habibian. Frameexit: Conditional early exiting for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15608–15618, 2021. 2, 3, 6, 7

[15] Yu-Gang Jiang, Zuxuan Wu, Jun Wang, Xiangyang Xue, and Shih-Fu Chang. Exploiting feature and class relationships in video categorization with regularized deep neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):352–364, 2017. 2, 6

[16] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. 2, 6

[17] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 6

[18] Bruno Korbar, Du Tran, and Lorenzo Torresani. Scsampler: Sampling salient clips from video for efficient action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6232–6242, 2019. 2, 3, 6, 7, 8

[19] Hengduo Li, Zuxuan Wu, Abhinav Shrivastava, and Larry S Davis. 2d or not 2d? adaptive 3d convolution selection for efficient video recognition. *arXiv preprint arXiv:2012.14950*, 2020. 2, 7

[20] Yan Li, Bin Ji, Xintian Shi, Jianguo Zhang, Bin Kang, and Limin Wang. Tea: Temporal excitation and aggregation for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 909–918, 2020. 1, 2

[21] Zhenyang Li, Kirill Gavrilyuk, Efstratios Gavves, Mihir Jain, and Cees GM Snoek. Videolstm convolves, attends and flows for action recognition. *Computer Vision and Image Understanding*, 166:41–50, 2018. 2

[22] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7083–7093, 2019. 1, 2, 4

[23] Zhaoyang Liu, Donghao Luo, Yabiao Wang, Limin Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Tong Lu. Teinet: Towards an efficient architecture for video recognition. In *AAAI*, pages 11669–11676, 2020. 2

[24] Zhaoyang Liu, Limin Wang, Wayne Wu, Chen Qian, and Tong Lu. Tam: Temporal adaptive module for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13708–13718, 2021. 2

[25] Yue Meng, Chung-Ching Lin, Rameswar Panda, Prasanna Sattigeri, Leonid Karlinsky, Aude Oliva, Kate Saenko, and Rogerio Feris. Ar-net: Adaptive frame resolution for effi-

cient action recognition. In *European Conference on Computer Vision*, pages 86–104. Springer, 2020. 2, 3, 6, 7, 8

[26] Yue Meng, Rameswar Panda, Chung-Ching Lin, Prasanna Sattigeri, Leonid Karlinsky, Kate Saenko, Aude Oliva, and Rogerio Feris. Adafuse: Adaptive temporal fusion network for efficient action recognition. In *International Conference on Learning Representations*, 2020. 2, 6, 8

[27] AJ Piergiovanni, Anelia Angelova, and Michael S Ryoo. Tiny video networks. *Applied AI Letters*, page e38, 2019. 1

[28] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017. 1, 2

[29] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *arXiv preprint arXiv:1406.2199*, 2014. 1, 2

[30] Ximeng Sun, Rameswar Panda, Chun-Fu Richard Chen, Aude Oliva, Rogerio Feris, and Kate Saenko. Dynamic network quantization for efficient video inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7375–7385, 2021. 2, 3, 6, 7

[31] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 1, 2

[32] Du Tran, Heng Wang, Lorenzo Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5552–5561, 2019. 1, 2

[33] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 1, 2

[34] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016. 1, 2, 3

[35] Yulin Wang, Zhaoxi Chen, Haojun Jiang, Shiji Song, Yizeng Han, and Gao Huang. Adaptive focus for efficient video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16249–16258, October 2021. 2, 3, 6, 7

[36] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 5

[37] Wenhao Wu, Dongliang He, Xiao Tan, Shifeng Chen, and Shilei Wen. Multi-agent reinforcement learning based frame sampling for effective untrimmed video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6222–6231, 2019. 3, 6, 7

[38] Zuxuan Wu, Caiming Xiong, Yu-Gang Jiang, and Larry S Davis. Liteeval: A coarse-to-fine framework for resource efficient video recognition. *arXiv preprint arXiv:1912.01601*, 2019. 2, 6, 7, 8

[39] Zuxuan Wu, Caiming Xiong, Chih-Yao Ma, Richard Socher, and Larry S Davis. Adaframe: Adaptive frame selection for fast video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1278–1287, 2019. 2, 3, 6, 7

[40] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 1(2):5, 2017. 1

[41] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2678–2687, 2016. 2, 3

[42] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015. 1, 2

[43] Yin-Dong Zheng, Zhaoyang Liu, Tong Lu, and Limin Wang. Dynamic sampling networks for efficient action recognition in videos. *IEEE Transactions on Image Processing*, 29:7970–7983, 2020. 3

[44] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. *arXiv:1801.00054*, 2017. 3

[45] Yizhou Zhou, Xiaoyan Sun, Zheng-Jun Zha, and Wenjun Zeng. Mict: Mixed 3d/2d convolutional tube for human action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[46] Wencheng Zhu, Jiwen Lu, Jiahao Li, and Jie Zhou. Dsnet: A flexible detect-to-summarize network for video summarization. *IEEE Transactions on Image Processing*, 30:948–962, 2020. 3

[47] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 695–712, 2018. 1