Delving into Details: Synopsis-to-Detail Networks for Video Recognition

Shuxian Liang[1,2⋆], Xu Shen[2], Jianqiang Huang[2], and Xian-Sheng Hua[1⋆⋆]

[1] Zhejiang University
shuxian.lsx@zju.edu.cn, huaxiansheng@gmail.com,
[2] Alibaba Cloud Computing Ltd.
{shenxuustc,jianqiang.jqh}@gmail.com

Abstract. In this paper, we explore the details in video recognition with the aim to improve the accuracy. It is observed that most failure cases in recent works fall on the mis-classifications among very similar actions (such as high kick *vs.* side kick) that need a capturing of fine-grained discriminative details. To solve this problem, we propose synopsis-to-detail networks for video action recognition. Firstly, a synopsis network is introduced to predict the top-k likely actions and generate the synopsis (location & scale of details and contextual features). Secondly, according to the synopsis, a detail network is applied to extract the discriminative details in the input and infer the final action prediction. The proposed synopsis-to-detail networks enable us to train models directly from scratch in an end-to-end manner and to investigate various architectures for synopsis/detail recognition. Extensive experiments on benchmark datasets, including Kinetics-400, Mini-Kinetics and Something-Something V1 & V2, show that our method is more effective and efficient than the competitive baselines. Code is available at: https://github.com/liang4sx/S2DNet.

Keywords: Video recognition, action recognition

1 Introduction

The explosive growth of online videos requires for an automatic recognition of large-scale videos, including human actions, events or other contents within them. For example, there are more than 10^5 hours of fresh video contents uploaded to YouTube every day to be processed for ranking and recommendation. In this situation, both high accuracy and high efficiency are required for large-scale online video analysis.

Because of the strong expressive power, in recent years, deep networks become the mainstream solutions for video recognition [5, 6, 12, 24, 41, 45, 46, 51]. To model spatial and temporal patterns in videos, 2D CNNs based methods first extract spatial features of each frame with 2D CNNs, then model the temporal

⋆ This work was done when the author was visiting Alibaba as a research intern.
⋆⋆ Corresponding author.

patterns based on a temporal fusion of spatial features. The temporal fusion strategies consist of post-hoc fusion [6, 12, 15, 60] or mid-level fusion [62, 47, 56]. These methods sacrifice the fine-grained temporal modeling at the low level for efficiency, resulting in a moderate accuracy. 3D CNNs [5, 10, 11, 46] handles spatial and temporal dimensions jointly, which achieve higher accuracy, but the computational costs are large, making them difficult to process large-scale real-time online videos. Recently, a feasible way for high accuracy and high efficiency is to investigate better temporal modeling based on 2D CNNs [32, 42, 53]. In [32], parts of the 2D CNN channels are shifted along temporal dimension to merge information among neighboring frames efficiently. [42] introduces a coarse-to-fine framework, where features of multi-scale input are combined for final prediction and early exits of easy cases are adopted for higher inference speed. In [53], a policy network is used to select relevant regions and frames for better efficiency. However, there are still a number of failed cases in their results.

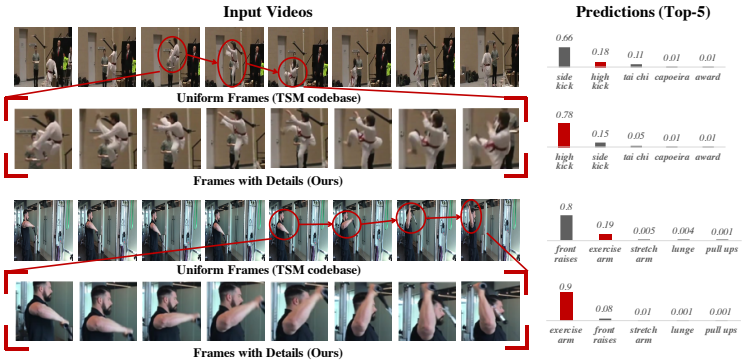

Fig. 1: Two failure cases of recent works [32, 42, 53]. The left shows both the input clips of uniform frames used by these works (in first/third rows), and the input clips of frames with detail sampled by our method (in second/fourth rows). The right part shows the predictions generated by [32] using these clips. The bars in red denote the probabilities of ground-truth actions. Without the details, the first case is mis-classified since the high kick moment (with the leg in front of the hip) is missing ($3^{\text{rd}}/4^{\text{th}}$ frames in the second row) and the pose ($2^{\text{nd}}/3^{\text{rd}}/4^{\text{th}}/5^{\text{th}}$ frames in the first row) of the person seems like the pose of side kick (with the leg extending out from the hip). Also, the second case is mis-classified because the motion of exercising arms (lifting the arms upward) is indistinctive and the pose of the person in earlier frames ($1^{\text{st}}/2^{\text{nd}}/3^{\text{rd}}/4^{\text{th}}$ frames in the third row) is similar to the pose of front raises (lifting the arms to the shoulder height)

We investigate the failure cases in recent works and find most of these failure cases fall on the mis-classifications among very similar actions (such as high kick *vs.* side kick) that need a deep understanding of fine-grained discriminative

details. Two failure cases on Kinetics-400 [5] are shown in Fig. 1. With the conventional uniform frames, we can observe that the top-5 predictions of recent works are rather reasonable and share the same scene ("martial art" for case 1 and "indoor exercise" for case 2). But without the details in the second/fourth rows, even for our human eyes, it's difficult to rank the ground-truth actions "high kick" and "exercise arm" in the first place. Intuitively, for action recognition of a video, the observation process of humans consists of two stages [2]: 1) Go through the video and identify the key frames/regions that are related to the most likely actions in consideration. 2) Slow the playback rate of key frames and zoom in the related region of each frame to obtain extra *details* for a more precise prediction. Inspired by this process, to further improve the accuracy of video recognition, we need to introduce the zoom in/out effects in both spatial and temporal dimensions for better inspection of details in the video.

Following the aforementioned analysis and motivation, we present a new perspective for accurate video recognition by proposing a novel synopsis-to-detail network (S2DNet). Firstly, a synopsis network goes through the input video, predicts the top-k most likely actions and generates the synopsis (location & scale of details and contextual features). Secondly, given the synopsis, a detail network is introduced to dynamically extract the details from the input and infer the final prediction based on the details. As all modules (including samplers, backbones and classifiers) of the synopsis/detail networks are differentiable, the synopsis-to-detail network can be trained directly from scratch in an end-to-end manner. Note that both recognizing top-k likely actions and discriminating one out of the top-k actions using details are much easier tasks than direct action recognition from the input video. As a result, despite adopting light-weighted models for both synopsis/detail networks, our method achieves both high accuracy and high efficiency.

There is another recent architecture for video recognition which has a coarse-to-fine (C2F) design [33, 42], but provides conceptually different perspectives. The C2F methods have not explored the potential of sampling details (zoom in detail regions and slow the playback rate of detail frames), which is a key concept in our method. Moreover, in C2F methods, each network shares the same task of classifying among all actions in parallel. The proposed S2DNet, however, formulates the task of action recognition as a progressive process, where the final output is generated by two simpler tasks (firstly recognizing top-k likely actions and then discriminating one out of the top-k actions using extra details).

Extensive experiments on benchmark datasets, including Kinetics-400, Mini-Kinetics and Something-Something V1 & V2, show that our method is more effective and efficient than the competitive baselines.

2 Related Works

Action Recognition. One prevalent approach to capture spatio-temporal patterns of actions adopts Convolutional Neural Networks (CNNs). Early works extract frame-level features with 2D CNNs and empower them with temporal

dynamics via two-stream networks with optical flow [45], temporal averaging [51], long short-term memory [6], *etc.* The other line of works develops 3D CNNs [12, 19, 41, 46], handling both spatial and temporal dimensions jointly. Later works on 3D CNNs leverage self-attention mechanisms [52], pathways of different framerates [11], learnable correlation operators [49], *etc.* More recently, there have been attempts to apply vision transformers [7] for action recognition [1, 3, 8, 39], matching or exceeding state-of-the-arts on multiple datasets.

In consideration of both accuracy and efficiency, recent efforts of efficient action recognition methods lie in two folds. The first focuses on designing lightweighted architectures, such as 2D CNNs with cost-efficient temporal modeling modules [9, 30, 32, 34, 50, 54, 62] and optimized 3D CNNs [10, 28, 35, 47]. The second attempts to dynamically allocate computation along time axis [14, 25, 26, 36, 37, 55, 59] or in both space and time [42, 53].

Coarse-to-fine Architectures. The coarse-to-fine architectures have a long history in computer vision [13, 27, 48, 58, 61]. On the one hand, the architectures alleviate computational costs by tackling easy cases in the coarsest level [27, 29]. On the other, they also help improve accuracy by fusing features of different levels [27], eliminating redundancy of input [4, 58] and reducing searching space [31, 44, 61]. In the context of action recognition, prior works adopt coarse-to-fine architectures for early exits of easy cases [42] and multi-level feature fusion [33, 42]. Our design of two-stage architecture is inspired by these prior works.

S2DNet is also similar in form to some other lines of works, *e.g.*, space-time attention and ensembling. Due to page limit, we defer discussions about the differences between these works and S2DNet to the supplementary material.

3 Synopsis-to-Detail Network

The proposed Synopsis-to-Detail Networks (S2DNet) (Fig. 2) consists of a Synopsis Network (SNet) and a Detail Network (DNet). The **synopsis** in this paper refers to a brief summary of the key factors of the input video, which is highly related to the corresponding top-k likely actions. The **details** in this paper refer to the discriminative factors that assist in differentiating the true action out of the aforementioned top-k actions.

The process of S2DNet is like: first, SNet goes through the video, predicts the top-k actions and extracts the synopsis; then, given the synopsis, DNet delves into related details of the video and infers the final precise action.

3.1 Synopsis Network (SNet)

The synopsis includes *location & scale parameters of details* (θ) and *contextual features* (m) related to the top-k likely actions. The former specifies the key frames and regions of the input video. The latter acts as task information for the extraction of discriminative features among the top-k actions. SNet learns the above information from the raw frames. It is based on a classic action recognition architecture, including a sampler, a backbone and a classifier.

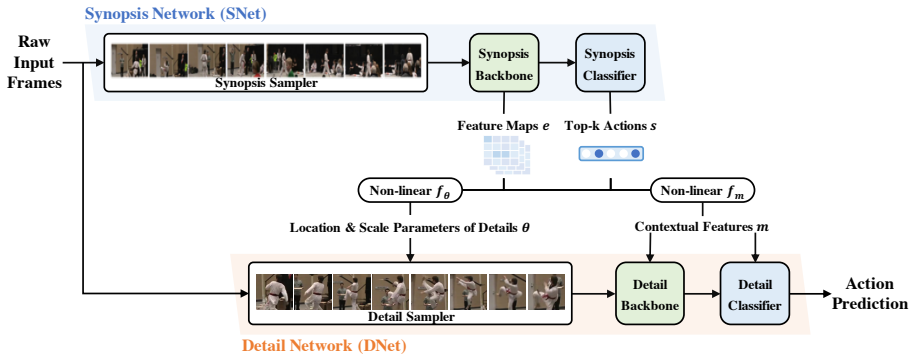

Fig. 2: Model Overview. S2DNet consists of a Synopsis Network and a Detail Network. First, the Synopsis Network goes through the video, predicts the top-k likely actions and extracts video synopsis accordingly (*location & scale of details* θ and *contextual features* m). Then, given the synopsis, the Detail Network delves into related details of the video and infers the final precise action

Synopsis Sampler. Consider a video with $T \times \tau$ frames totally, where T is the number of raw input frames and τ is the temporal stride of these frames. The raw input frames are denoted by $\mathcal{I} \in \mathbb{R}^{T \times H \times W}$, where H and W are height and width (the dimension of RGB channels are omitted for brevity). The synopsis sampler uniformly samples T_S frames out of the raw input frames $\mathcal{I}$, then spatially resizes them to $H_S \times W_S$. The output frames of the synopsis sampler are denoted by $\mathcal{V} \in \mathbb{R}^{T_S \times H_S \times W_S}$. Importantly, the sizes of $\mathcal{V}$ are much smaller than the sizes of $\mathcal{I}$ (*e.g.*, 16×144×144 *vs.* 150×224×224). This ensures a low computational cost of the following feature extraction.

Synopsis Backbone. The synopsis backbone f_s can be any 2D/3D CNNs for video recognition (*e.g.*, [10, 11, 32]). It takes the output frames of the synopsis sampler as input and generates spatio-temporal feature maps of the video:

$$e = f_s(\mathcal{V}), \tag{1}$$

where $e \in \mathbb{R}^{C_S \times T_S \times h_S \times w_S}$. C_S, h_S and w_S are channel size, height and width.

Synopsis Classifier. The synopsis classifier is designed to recognize the top-k likely actions of the input video. A fully-connected (FC) layer with softmax is adopted as the synopsis classifier h_s, which takes the feature maps e as input:

$$p = h_s(\text{GAP}(e)), \tag{2}$$

where GAP is global average pooling and p is the softmax predictions over all N actions. A k-hot vector s (with 1s for the top-k actions and 0s for other actions) is introduced to represent the top-k likely actions. Specifically, s is the top-k binarization of p, which takes only a single line of code[3] in PyTorch [40].

[3] `torch.nn.functional.one_hot(torch.topk(p,k).indices,N).sum(dim=-1)`

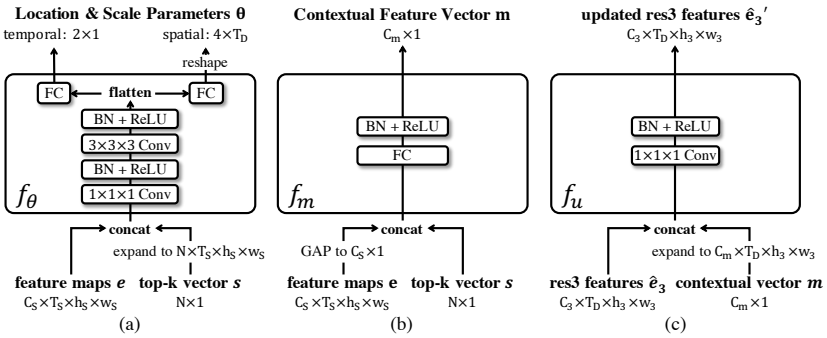

Fig. 3: Illustration of (a) generating location & scale parameters of details θ, (b) generating contextual features m, and (c) fusing contextual features m to intermediate feature maps of DNet

To ensure that the ground-truth class falls in the top-k predictions, a cross entropy loss is applied to the output of SNet:

$$\mathcal{L}_s = -\sum_{n=1}^{N} \mathbb{I}\left(n = y\right) \log p\left(n\right), \tag{3}$$

where y denotes the ground-truth class and $\mathbb{I}\left(\cdot\right)$ is an indicator function.

Synopsis I: Generating location & scale parameters of details θ (Fig. 3a). Inspired by the differentiable attention [17, 18, 21, 57], location and scale are formulated as 6 individual parameters: $\theta = (\mu_t, \mu_y, \mu_x, \delta_t, \delta_y, \delta_x)$. These parameters will be used in the sampler of DNet and their explanations are deferred to Sec. 3.2. Taking the feature maps e and the top-k vector s as input, the parameters are generated by the module f_θ:

$$\theta = f_\theta\left(\left[e, E(s)\right]\right), \tag{4}$$

where $[\cdot, \cdot]$ denotes concatenation and $E(\cdot)$ expands the vector s to the shape of e ($N{\times}1{\rightarrow}N{\times}T_S{\times}h_S{\times}w_S$). f_θ consists of one 1×1×1 convolutional layer, one 3×3×3 convolutional layer (both with BN [22] and ReLU [38]) and two parallel fully-connected (FC) layers. The two FC layers generate $(\mu_t, \delta_t) \in \mathbb{R}^{2\times1}$ for temporal sampling and $(\mu_y, \mu_x, \delta_y, \delta_x) \in \mathbb{R}^{4\times T_D}$ for spatial sampling, respectively.

Synopsis II: Generating contextual features m (Fig. 3b). Given the feature maps e and the top-k vector s, m is generated by the module f_m:

$$m = f_m\left(\left[\mathrm{GAP}(e), s\right]\right), \tag{5}$$

where $m \in \mathbb{R}^{C_m \times 1}$. The module f_m is a FC layer with ReLU and BN.

3.2 Detail Network (DNet)

To delve into the details, DNet first samples a space-time volume with details from the raw frames $\mathcal{I}$, based on *location & scale parameters of details* θ. Then, it extracts discriminative features over the volume with the help of *contextual features* m. DNet has a sampler, a backbone and a classifier.

Detail Sampler. Given the raw input $\mathcal{I} \in \mathbb{R}^{T\times H\times W}$ and the real-valued parameters $\theta = (\mu_t, \mu_y, \mu_x, \delta_t, \delta_y, \delta_x)$, the detail sampler generates a volume $\hat{\mathcal{V}} \in \mathbb{R}^{T_D\times H_D\times W_D}$ that contains rich details of the top-k actions. Following the differentiable attention [17, 18, 21, 57], an array of 3-dimensional filters is applied to the raw input $\mathcal{I}$, yielding a sequence of local patches with smoothly varying location and scale.

Specifically, given the expected output size $T_D\times H_D\times W_D$, a $T_D\times H_D\times W_D$ grid of sampling filters is applied to $\mathcal{I}$. The center of the grid is determined by the real-valued parameters (μ_t, μ_y, μ_x), which are formally time/height/width offsets to the center of $\mathcal{I}$. The time/height/width strides of the grid are controlled by the real-valued parameters $(\delta_t, \delta_y, \delta_x)$. Consequently, the grid location (p_t, p_y, p_x) at frame z, row j and column i is:

$$\begin{aligned} p_t(z) &= \mu_t T + (z - T_D/2 - 0.5)\delta_t, \\ p_y(j) &= \mu_y H + (j - H_D/2 - 0.5)\delta_y, \\ p_x(i) &= \mu_x W + (i - W_D/2 - 0.5)\delta_x. \end{aligned} \tag{6}$$

Note that the smaller δ_t is, the slower the playback rate of detail frames will be. The smaller (δ_y, δ_x) are, the larger the resolution ratio of detail regions will be.

Previous works use Gaussian filters [18, 21, 57] or linear filters [21] for sampling. We have experimented with both filters and find Gaussian filters suit better for our case (discussed in the supplementary material). The coordinates specified by the grid are the mean locations of Gaussian filters. Given the variance as 1, the temporal, horizontal and vertical filtering weights $\mathcal{G}_T$ (dimensions $T_D\times T$), $\mathcal{G}_Y$ (dimensions $H_D\times H$) and $\mathcal{G}_X$ (dimensions $W_D\times W$) are defined as:

$$\begin{aligned} \mathcal{G}_T[z, r] &= \frac{1}{Z_T} exp(-\frac{(p_t(z) - r)^2}{2}), \\ \mathcal{G}_Y[j, v] &= \frac{1}{Z_Y} exp(-\frac{(p_y(j) - v)^2}{2}), \\ \mathcal{G}_X[i, u] &= \frac{1}{Z_X} exp(-\frac{(p_x(i) - u)^2}{2}), \end{aligned} \tag{7}$$

where (z, j, i) are coordinates of a point in the output $\hat{\mathcal{V}}$ and (r, v, u) are coordinates of a point in the input $\mathcal{I}$. Z_T, Z_Y and Z_X are the normalization constants that ensure $\sum_r \mathcal{G}_T[z, r]$=1, $\sum_v \mathcal{G}_Y[j, v]$=1 and $\sum_u \mathcal{G}_X[i, u]$=1.

Finally, the overall sampling operation is formulated as three 1-dimensional Gaussian filtering. The output volume $\hat{\mathcal{V}}$ from the raw input $\mathcal{I}$ is sampled via $\hat{\mathcal{V}} = \mathcal{G}_X\mathcal{G}_Y\mathcal{G}_T\mathcal{I}$, where the dimension transposition is omitted for brevity.

Detail Backbone. Similar to SNet, the detail backbone f_d can be instantiated with various video backbones. Taking the output volume of the detail sampler $\hat{\mathcal{V}}$ and the contextual features m as input, it generates detail feature maps $\hat{e} \in \mathbb{R}^{C_D \times T_S \times h_D \times w_D}$ of details by:

$$\hat{e} = f_d\left(\hat{\mathcal{V}}, m\right). \tag{8}$$

Fusing the contextual features m to detail features (Fig. 3c). Following [11], m is laterally fused into detail features via convolution. For ResNets-like architectures [10, 20], we apply the lateral fusion after *res3* and *res4* (more discussions are in our supplementary material), for example,

$$\hat{e}'_3 = f_u([\hat{e}_3, E(m)]), \tag{9}$$

where $\hat{e}_3/\hat{e}'_3 \in \mathbb{R}^{C_3 \times T_D \times h_3 \times w_3}$ is the original/updated *res3* features and $E(\cdot)$ expands m to the shape of $\hat{e}_3$ ($C_m \times 1 \rightarrow C_m \times T_D \times h_3 \times w_3$). f_u is a $1 \times 1 \times 1$ convolutional layer with BN and ReLU.

Detail Classifier. The detail classifier h_d outputs the final precise predictions by generating a vector $\hat{p}$ of softmax probabilities over all classes. Taking the detail feature maps $\hat{e}$ and the contextual features m as input, a FC layer is adopted for the classification:

$$\hat{p} = h_d\left([\mathrm{GAP}(\hat{e}), m]\right). \tag{10}$$

Notably, m conveys contextual information (the global features to refer to and the top-k actions to consider). Since DNet focuses on differentiating top-k actions, we adopt the top-k vector s as an "attention" for all output actions by multiplying $\hat{p}$ with s. Therefore, the cross entropy loss for DNet is refined as:

$$\mathcal{L}_d = -\sum_{n=1}^{N} \mathbb{I}(n = y) s(n) \log \hat{p}(n). \tag{11}$$

Finally, the overall loss of S2DNet is computed by:

$$\mathcal{L} = \alpha \mathcal{L}_s + \mathcal{L}_d, \tag{12}$$

where α weighs the task of top-k action recognition and top-1 action recognition.

3.3 Instantiations

The proposed S2DNet is generic and can be instantiated using various video backbones and implementation specifics. In this paper, we focus on action recognition for trimmed videos, and opt to instantiate S2DNet with the classic 2D/3D CNNs-based methods (*e.g.*, [10, 11, 32]). However, note that our method can also apply with recent Vision Transformers [1, 3, 7, 8, 39].

Specifically, to compare with state-of-the-art 2D CNNs, a TSM model with MobileNetV2 [43] is used in SNet, and a TSM model with ResNet-50 [20] is used in DNet. To compare with state-of-the-art 3D CNNs, a X3D-S [10] model is used in SNet and a X3D-M/X3D-XL model is used in DNet. The results using TSM/X3D backbones are reported in Sec. 4.2.

4 Experiments

In this section, we empirically validate the proposed method on four action recognition datasets using the standard evaluation protocols. First, in Sec. 4.2, the comparison with state-of-the-art (SOTA) methods for S2DNet is presented. Second, in Sec. 4.3, ablation results on different configurations of S2DNet are discussed. Third, in Sec. 4.4, S2DNet is also compared with efficient action recognition methods, demonstrating its merit of high efficiency. At last, we present the efficiency analysis (Sec. 4.5) and some visualization results (Sec. 4.6), to provide additional insights of S2DNet.

4.1 Setups

Datasets. Our experiments are based on four widely-used action recognition datasets. Kinetics-400 [5] is a large-scale action recognition dataset with $240k$ training and $20k$ validation videos in 400 action classes. Mini-Kinetics (assembled by [36]) is a subset of the Kinetics-400 and it contains $121k$ training and $10k$ validation videos in 200 action classes. Something-Something V1 & V2 [16] are two action recognition datasets with $98k/194k$ videos in the shared 174 classes. When compared with the SOTA methods, S2DNet is evaluated on Kinetics-400 and Something-Something V2. When compared with the efficient methods, S2DNet is evaluated on Mini-Kinetics and Something-Something V1 & V2. Following previous works, we report top-1 and/or top-5 classification accuracy (%).

Training & Inference. We use τ=2, k=5, α=0.1 and C_m=256 for all experiments in this work. During training, following [32, 36, 53], random scaling, 224×224 random cropping and random horizontal flipping (except Something-Something V1&V2) are adopted for data augmentation. During inference, for the spatial domain, the raw frames are first resized to 256×256. On Kinetics-400, three 224×224 patches are cropped from the resized frames, following common practice in [10, 11, 52]. On other datasets, only one center 224×224 cropping is used. For the temporal domain, on all datasets, we use only one clip per video. More details (*e.g.*, training hyper-parameters) are in the supplementary material.

4.2 Comparison with State-of-the-Arts

This section provides the comparison with SOTA methods on Kinetics-400 and Something-Something V2, considering both accuracy and efficiency. Kinetics-400 includes the daily actions that are highly relevant to interacting objects or scene context, thereby requiring strong spatio-temporal modeling capacity. Recent methods on this dataset are mostly based on 3D CNNs to learn space-time features jointly. Something-Something V2 pays more attention to modeling temporal relationships as pointed in [32, 56, 60]. Recent methods on this dataset include both 2D CNNs and 3D CNNs.

Implementation details. To compare with SOTA methods on Kinetics-400, S2DNet is instantiated using efficient 3D CNNs (*i.e.*, X3D [10]). To compare with SOTA methods on Something-Something V2, S2DNet is instantiated using

Table 1: Comparison with the state-of-the-arts on Kinetics-400 and Something-Something V2. "FLOPs" denotes multiply-add operations. "R50" and "R101" denote ResNet-50 and ResNet-101 [20]. "MN2" denotes MobileNetV2 [43]. "-" indicate that the numbers are not available for us. For S2DNet, "X3D-S/X3D-M" indicates that X3D-S is used in SNet and X3D-M is used in DNet

Methods	**Backbone**	**Frames**	**top-1**	**top-5**	**FLOPs×Views**
Kinetics400					
TSM [32]	R50	16	74.7	91.4	65G×30
C2F [42]	R50	16	76.0	-	18G×9
TANet [34]	R50	16	76.9	92.9	86G×12
SlowFast [11]	R50	8/32	75.6	92.1	36G×30
SmallBig [28]	R50	8	76.3	92.5	57G×6
CorrNet [49]	R50	32	77.2	-	115G×30
TDN [50]	R50	16	77.5	93.2	72G×30
SmallBig [28]	R101	16	77.4	93.3	418G×12
TDN [50]	R101	16	78.5	93.9	132G×30
CorrNet [49]	R101	32	79.2	-	224G×30
SlowFast [11]	R101-NL	16/64	79.8	93.9	234G×30
X3D [10]	X3D-M	16	76.0	92.3	6.2G×30
X3D [10]	X3D-XL	16	79.1	93.9	48G×30
S2DNet (ours)	X3D-S/X3D-M	16/16	78.0	93.6	**5.4G×3**
S2DNet (ours)	X3D-S/X3D-XL	16/16	**80.6**	**94.2**	39G×3
Something-Something V2					
GST [35]	R50	16	62.6	87.9	59G×6
SlowFast [11]	R50	128	63.0	88.5	-
TSM [32]	R50	16	63.4	88.5	65G×6
C2F [42]	R50	16	64.1	-	85G×6
SmallBig [28]	R50	16	63.8	88.9	105G×6
STM [23]	R50	16	64.2	89.8	67G×30
TANet [34]	R50	16	64.6	89.5	72G×6
TDN [50]	R50	16	65.3	89.5	72G×1
S2DNet (ours)	MN2/R50	16/16	**66.4**	**89.9**	**35G×1**

efficient 2D CNNs (*i.e.*, TSM [32]). For both datasets, the input parameters are: T_S=16, T_D=16, H_S=W_S=224 and H_D=W_D=144.

Kinetics-400. The upper part of Tab. 1 presents the comparison with SOTA results for two S2DNet instantiations using X3D. Note that X3D-M and X3D-XL achieve similar performances to the SOTA works with ResNet-50 and ResNet-101, respectively. To make a fair comparison, we compare **S2DNet (X3D-S/X3D-M)** with ResNet-50 based works (*e.g.*, [11, 28, 30, 32, 34, 49]), and compare **S2DNet (X3D-S/X3D-XL)** with ResNet-101 based works (*e.g.*, [11, 28, 49, 50, 52]). Firstly, it is observed that S2DNet improves the X3D counterparts by a margin of 2.0%/1.5% on top-1 accuracy, while requiring fewer multiply-add operations (FLOPs). Secondly, **S2DNet (X3D-S/X3D-M)** achieves +0.5% top-1 accuracy at 13.3× fewer FLOPs than the best reported result of TDN

(R50) [50]. Thirdly, **S2DNet (X3D-S/X3D-XL)** is +0.8% more accurate on top-1 accuracy and requires 6.0× fewer FLOPs than the best reported result of SlowFast (R101) [11]. The above results verify that our top-k action recognition contributes to a better performance of the final precise action prediction.

Moreover, although the top-5 results are generated by the light-weighted SNet (X3D-S), the best top-5 performance of S2DNet is still better than X3D-XL and SlowFast. This phenomenon indicates that a high-quality detail recognition help foster better top-k action recognition in return.

Something-Something V2. The lower part of Tab. 1 presents the comparison with SOTA results for one S2DNet instantiation using TSM. The results show that S2DNet achieves a significant improvement of 3.0%/1.4% on top-1/top-5 accuracy for TSM. Meanwhile, compared with the best reported result of TDN, our method achieves +1.1% top-1 performance at 2.0× fewer FLOPs. These results demonstrate that, by formulating action recognition into first recognizing top-k actions and then discriminating one out of the top-k actions, the task is easier to handle than the direct action recognition from input videos.

4.3 Ablation Experiments

This subsection provides ablation studies on Mini-Kinetics comparing accuracy and efficiency.

Implementation details. In this subsection, we use TSM with MobileNetV2 in SNet and TSM with ResNet-50 in DNet. The input parameters are : T_S=T_D=16, H_S=W_S=144 and H_D=W_D=112.

Architecture configurations. Tab. 2a shows results of different architecture configurations of S2DNet. These configurations determine whether the two sub-networks (SNet and DNet) are used together or separately. Notably, to measure the standalone performance of DNet, all sampling parameters for the detail sampler are frozen, all lateral fusions in the detail backbone are canceled and the detail classifier takes as input only the detail features $\hat{e}$.

Compared with the full network ($a1$ *vs.* $a0$), SNet presents more performance drop on top-1 (−6.1%) than top-5 (−1.6%). This is because the top-5 classfication is much simpler than the direct top-1 classification. Despite using a more sophisticated backbone, DNet is inferior to SNet with a performance drop of 0.5% on top-1 and 1.8% on top-5 ($a2$ *vs.* $a1$). This indicates that location & scale information provided by SNet is better than a fixed pre-defined one. By combining SNet and DNet, S2DNet achieves a significant gain of 6.1%/6.7% on top-1 accuracy ($a0$ *vs.* $a1$/$a2$). This result indicates that synopsis information and detail information complement each other.

Detail network configurations. Results of different configurations of DNet are shown in Tab. 2b. For $b1$/$b3$, location & scale parameters θ for the detail sampler are frozen. For $b1$/$b2$, contextual features m (Eq. 5) are not laterally fused to the intermediate features of the detail backbone. By incorporating location & scale parameters θ provided by SNet, DNet achieves a significant gain of 2.6% on top-1 accuracy and a gain of 0.8% on top-5 accuracy ($b2$ *vs.* $b1$). This result reveals that the detail sampler extracts more discriminative details from

Table 2: Ablations on Mini-Kinetics. In all subtables, the bottom rows ($a0$, $b0$ and $c0$) represent the default settings of S2DNet

(a) Architecture configuration

	SNet	DNet	top-1	top-5	FLOPs
$a1$	✓		68.6	89.5	**2.2G**
$a2$		✓	68.1	87.7	15.3G
$a0$	✓	✓	**74.7**	**91.1**	18.0G

(b) Detail network configuration

	Sample	Fuse	top-1	top-5	FLOPs
$b1$			70.7	90.1	**17.5G**
$b2$	✓		73.3	90.9	17.9G
$b3$		✓	71.7	90.9	17.6G
$b0$	✓	✓	**74.7**	**91.1**	18.0G

(c) Detail sampler configuration

	Space	Time	top-1	top-5	FLOPs
$c1$			71.7	89.6	**17.6G**
$c2$	✓		72.7	89.5	17.9G
$c3$		✓	74.0	89.4	17.9G
$c0$	✓	✓	**74.7**	**91.1**	18.0G

input video with the proposed detail sampling. The lateral fusion of contextual features in the detail backbone brings in a 1.0% improvement ($b3$ *vs.* $b1$), which indicates that contextual features boost the final prediction of actions. Finally, using all components in DNet as proposed ($b0$) leads to a further improvement, indicating that these components boost each other.

Detail sampler configurations. As shown in Tab. 2c, S2DNet is evaluated under different space-time sampling strategies. For $c1$, location & scale parameters θ are frozen, where the center croppings of uniformly sampled frames are used as the input volume $\hat{\mathcal{V}}$ for DNet. This setting leads to a performance drop of 2.8% on top-1 accuracy ($c1$ *vs.* $c0$). Only introducing spatial sampling brings in a 1.0% performance gain ($c2$ *vs.* $c1$). Only introducing temporal sampling brings in a performance gain of 2.3% ($c3$ *vs.* $c1$). These results show that both spatial sampling and temporal sampling play an important role in delving into fine-grained discriminative details from input videos.

4.4 Comparison with Efficient Action Recognition Methods

Although S2DNet is proposed to improve the accuracy, it also enjoys the merit of high efficiency. This subsection provides the comparison with efficient action recognition methods for S2DNet on three datasets: Mini-Kinetics, Something-Something V1 and Something-Something V2.

Implementation details. For a fair comparison, we use the same video backbones (MobileNetV2 and ResNet-50) as [26, 36, 53]. To balance the trade-off between efficiency and accuracy, we adopt T_S=T_D=16, H_S=W_S=144 and H_D=W_D=112 on Mini-Kinetics. On Something-Something V1 & V2, we adopt T_S=8, T_D=12, H_S=W_S=144 and H_D=W_D=144.

Table 3: Comparison with efficient action recognition methods. "BN-I" denotes to BN-Inception [22] and "R18" denotes ResNet-18. Only top-1 accuracy is reported since the top-5 results of many of these works are not available

(a) Mini-Kinetics

Method	Backbone	top-1	FLOPs
AR-Net [36]	MN2/R50	71.7	32G
AdaFocus [53]	MN2/R50	72.2	27G
AdaFuse [37]	R50	72.3	23G
DKP [25]	R18/R50	72.7	18G
S2DNet (ours)	MN2/R50	**74.7**	**18G**

(b) Something-Something V1/V2

Method	Backbone	top-1	FLOPs
ECO [62]	BN-I/R18	39.6/-	32G
TSM [32]	R50	45.6/59.1	33G
AdaFuse [37]	R50	46.8/59.8	31G
AdaFocus [53]	MN2/R50	48.1/60.7	34G
S2DNet (ours)	MN2/R50	**49.7**/**62.5**	**22G**

Results. Tab. 3 illustrates 2.0%/1.6%/1.8% better accuracy of S2DNet on the three datasets. While the gains are significant, it is also observed that S2DNet requires fewer FLOPs. This indicates that S2DNet is good at handling the trade-off between accuracy and efficiency, making it flexible to deploy in real world.

4.5 Efficiency Analysis

As in Tab. 4, compared with the efficient/SOTA methods [32, 50], S2DNet enjoys higher accuracy (+3%/+1.1%), lower latency (↓1.49×/↓2.50×), lower FLOPs (↓1.88×/↓2.05×) with minor/less extra parameters (+15%/-8.2%). The FLOPs and latency of S2DNet are lower for two reasons. First, SNet is light-weighted and brings a small amount of extra parameters and FLOPs. Second, DNet has a smaller spatial input size (*e.g.*, 224→144) by using local regions with details, which reduces the FLOPs significantly.

Table 4: Efficiency comparison on Something-Something V2. All measures use a Tesla V100 with batch size as 16. † denotes the reproduced results

Method	Backbone	Frames	FLOPs	Param.	Latency (ms)	top-1
TSM [32]	R50	16	66G	**24.3M**	15.7	63.4
TDN [50]	R50	16	72G	30.5M†	26.3†	65.3
S2DNet (ours)	MN2/R50	16/16	**35G**	28.0M	**10.5**	**66.4**

4.6 Visualization

In S2DNet, fine-grained details are sampled by a detail sampler, using *location & scale of details* from SNet. This process is visualized in Fig. 4, using the $a0$ instantiation of S2DNet in Tab. 2a. The upper half of the figure shows the

uniform frames sampled by SNet (and also by some previous works), and the lower half shows the contents sampled by the detail sampler.

Fig. 4: Visualization of the sampled contents of S2DNet on Kinetics-400. The labels in red denote ground-truth actions

In the first case, one can observe that the moment of headbutting is missing in the uniform frames while the pose of the man on the right is similar to punching person. This leads to a mis-classification of the previous works (*e.g.*, TSM). As for S2DNet, by leveraging the synopsis from SNet, it samples space-time details that are most related to the top-k actions (actions about hands/heads). Specifically, it slows the playback rate of the frames in which the two persons get close, and zooms in the regions where their moving hands/heads are conspicuous. By doing so, the action headbutting is recognized.

In the second case, one can observe that there are a lot of redundant frames in the uniform frames and the true action sniffing is indistinctive. This results in a mis-classification of the previous works. S2DNet samples space-time details that are most related to the top-k actions (actions about hands/faces). Specifically, it gets rid of the redundant frames and zooms in the regions where motions of hands/faces are conspicuous. By doing so, the action sniffing is recognized.

5 Conclusion

This paper proposes a novel network for video recognition, namely Synopsis-to-Detail Network. Inspired by the observation that recent works fail to differentiate very similar actions (such as high kick *vs.* side kick), S2DNet first predicts the top-k actions and generates the synopsis by coarsely going through the video. Then, according to the synopsis, it extracts discriminative details in the input and infers the final precise action. Extensive experiments demonstrate that our method outperforms existing works in terms of both accuracy and efficiency.

Acknowledgements. This work was partially supported by the National Key R&D Program of China under Grant 2020AAA0103901.

References

1. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. arXiv preprint arXiv:2103.15691 (2021)
2. Bear, M., Connors, B., Paradiso, M.A.: Neuroscience: Exploring the Brain, Enhanced Edition: Exploring the Brain. Jones & Bartlett Learning (2020)
3. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? arXiv preprint arXiv:2102.05095 (2021)
4. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: CVPR (2018)
5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
6. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR (2015)
7. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
8. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. arXiv preprint arXiv:2104.11227 (2021)
9. Fan, Q., Chen, C.F.R., Kuehne, H., Pistoia, M., Cox, D.: More is less: Learning efficient video representations by big-little network and depthwise temporal aggregation. NeurIPS (2019)
10. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: CVPR (2020)
11. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: ICCV (2019)
12. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: CVPR (2016)
13. Fleuret, F., Geman, D.: Coarse-to-fine face detection. IJCV **41**(1), 85–107 (2001)
14. Gao, R., Oh, T.H., Grauman, K., Torresani, L.: Listen to look: Action recognition by previewing audio. In: CVPR (2020)
15. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: Actionvlad: Learning spatio-temporal aggregation for action classification. In: CVPR (2017)
16. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The" something something" video database for learning and evaluating visual common sense. In: ICCV (2017)
17. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines (2014)
18. Gregor, K., Danihelka, I., Graves, A., Rezende, D., Wierstra, D.: Draw: A recurrent neural network for image generation. In: ICML (2015)
19. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: CVPR (2018)
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
21. Huang, Z., Xue, D., Shen, X., Tian, X., Li, H., Huang, J., Hua, X.S.: 3d local convolutional neural networks for gait recognition. In: ICCV (2021)
22. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)

23. Jiang, B., Wang, M., Gan, W., Wu, W., Yan, J.: Stm: Spatiotemporal and motion encoding for action recognition. In: ICCV (2019)
24. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)
25. Kim, H., Jain, M., Lee, J.T., Yun, S., Porikli, F.: Efficient action recognition via dynamic knowledge propagation. In: ICCV (2021)
26. Korbar, B., Tran, D., Torresani, L.: Scsampler: Sampling salient clips from video for efficient action recognition. In: ICCV (2019)
27. Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G.: A convolutional neural network cascade for face detection. In: CVPR (2015)
28. Li, X., Wang, Y., Zhou, Z., Qiao, Y.: Smallbignet: Integrating core and contextual views for video classification. In: CVPR (2020)
29. Li, X., Liu, Z., Luo, P., Change Loy, C., Tang, X.: Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In: CVPR (2017)
30. Li, Y., Ji, B., Shi, X., Zhang, J., Kang, B., Wang, L.: Tea: Temporal excitation and aggregation for action recognition. In: CVPR (2020)
31. Li, Y., Lin, W., See, J., Xu, N., Xu, S., Yan, K., Yang, C.: Cfad: Coarse-to-fine action detector for spatiotemporal action localization. In: ECCV (2020)
32. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: ICCV (2019)
33. Lin, W., Zhang, C., Lu, K., Sheng, B., Wu, J., Ni, B., Liu, X., Xiong, H.: Action recognition with coarse-to-fine deep feature integration and asynchronous fusion. In: AAAI (2018)
34. Liu, Z., Wang, L., Wu, W., Qian, C., Lu, T.: Tam: Temporal adaptive module for video recognition. In: ICCV (2021)
35. Luo, C., Yuille, A.L.: Grouped spatial-temporal aggregation for efficient action recognition. In: ICCV (2019)
36. Meng, Y., Lin, C.C., Panda, R., Sattigeri, P., Karlinsky, L., Oliva, A., Saenko, K., Feris, R.: Ar-net: Adaptive frame resolution for efficient action recognition. In: ECCV (2020)
37. Meng, Y., Panda, R., Lin, C.C., Sattigeri, P., Karlinsky, L., Saenko, K., Oliva, A., Feris, R.: Adafuse: Adaptive temporal fusion network for efficient action recognition. In: ICLR (2020)
38. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
39. Neimark, D., Bar, O., Zohar, M., Asselmann, D.: Video transformer network. arXiv preprint arXiv:2102.00719 (2021)
40. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. NeurIPS **32** (2019)
41. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: ICCV (2017)
42. Quader, N., Lu, J., Dai, P., Li, W.: Towards efficient coarse-to-fine networks for action and gesture recognition. In: ECCV (2020)
43. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR (2018)
44. Sarlin, P.E., Cadena, C., Siegwart, R., Dymczyk, M.: From coarse to fine: Robust hierarchical localization at large scale. In: CVPR (2019)
45. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NeurIPS (2014)

46. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015)
47. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: CVPR (2018)
48. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV **57**(2), 137–154 (2004)
49. Wang, H., Tran, D., Torresani, L., Feiszli, M.: Video modeling with correlation networks. In: CVPR (2020)
50. Wang, L., Tong, Z., Ji, B., Wu, G.: Tdn: Temporal difference networks for efficient action recognition. In: CVPR (2021)
51. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV (2016)
52. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018)
53. Wang, Y., Chen, Z., Jiang, H., Song, S., Han, Y., Huang, G.: Adaptive focus for efficient video recognition. In: ICCV (2021)
54. Weng, J., Luo, D., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Jiang, X., Yuan, J.: Temporal distinct representation learning for action recognition. In: ECCV (2020)
55. Wu, Z., Xiong, C., Jiang, Y.G., Davis, L.S.: Liteeval: A coarse-to-fine framework for resource efficient video recognition. NeurIPS (2019)
56. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: ECCV (2018)
57. Yang, J., Shen, X., Tian, X., Li, H., Huang, J., Hua, X.S.: Local convolutional neural networks for person re-identification. In: ACM MM (2018)
58. Zhang, J., Shan, S., Kan, M., Chen, X.: Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In: ECCV (2014)
59. Zhi, Y., Tong, Z., Wang, L., Wu, G.: Mgsampler: An explainable sampling strategy for video action recognition. In: ICCV (2021)
60. Zhou, B., Andonian, A., Oliva, A., Torralba, A.: Temporal relational reasoning in videos. In: ECCV (2018)
61. Zhu, S., Li, C., Change Loy, C., Tang, X.: Face alignment by coarse-to-fine shape searching. In: CVPR (2015)
62. Zolfaghari, M., Singh, K., Brox, T.: Eco: Efficient convolutional network for online video understanding. In: ECCV (2018)